Unique Signature Schemes with Tight Security in the Random Oracle Model via Hash-and-Subset-Sign

Thomas Attema [1,3], Lisa Kohl [1], and Joost van der Laan [2]

[1]Cryptology Group, CWI Amsterdam, The Netherlands
lisa.kohl@cwi.nl
[2]TCS Group, University of Amsterdam, The Netherlands
j.vanderlaan3@uva.nl
[3]Applied Cryptography & Quantum Applications, TNO, The Hague, The Netherlands
thomas.attema@tno.nl

Abstract

Unique signatures are digital signature schemes, where each message has exactly one valid signature. The uniqueness property typically comes at a price: Most unique signature schemes are known to have a security loss at least linear in the number of signature queries when aiming for existential unforgeability under chosen message attacks (EUF-CMA) security, which was shown to be inherent in the standard model by Coron (EUROCRYPT 2002) for signature schemes where public keys are verifiable (i.e., where it can be efficiently checked whether a public key has unique signatures). The only known tight unique signature schemes in the literature that support verifiable public keys are variants of chain-based signatures (CRYPTO 2017, FC 2018, EUROCRYPT 2022), which achieve tight security in the random oracle model. To achieve constant security loss, however, these schemes require $\log q$ calls of the random oracle to sign and verify, where q is an upper bound on the number of signatures generated using the scheme.

In this work, we provide the first tight unique signature scheme with verifiable public keys which requires only a constant number (namely, six) random oracle queries to sign and verify. Specifically, we provide a general transformation akin to hash-and-sign which transforms a signature scheme with much weaker security (a variant of random unforgeability) into an EUF-CMA secure scheme. Signatures of our resulting scheme consist of at most 3 sub-signatures in parallel.

Furthermore, we consider tightness in the multi-user setting, where the adversary may query signatures relative to many public keys, before deciding for which to forge. In this setting, we provide two constructions which achieve (1) a tight signature scheme, secure against semi-adaptive corruptions (i.e., where the adversary has to decide whether to corrupt a public key *before* making any signature queries relative to this key), and (2) a tight few-time signature scheme, secure against adaptive corruptions, where the scheme's setup is allowed to depend on an a-priori polynomial bound on the number of queries that the adversary has access to. These are the first unique signature schemes which are tightly secure in the multi-user model.

1 Introduction

Unique signatures, a special class of digital signature schemes [RSA78;Rab79], have gained significant attention since their formal definition by Goldwasser and Ostrovsky [GO93]. On top of authenticity, integrity, and non-repudiation, unique signatures ensure that for each message there is exactly *one* valid signature. The one-to-one correspondence between messages and signatures makes unique signatures particularly suitable for applications such as e-cash systems [Cha81;ASM07], e-lotteries [CHY+05], and in distributed leader election protocols [MRV99b;CCM+24].

EUF-CMA security. The security notion considered for digital signature schemes is typically existential unforgeability against chosen-message attacks (EUF-CMA), which requires that it is hard for an attacker to forge a signature on a fresh message after having seen signatures on q messages. Here, the adversary can choose both the message to provide a forgery on and the messages to obtain signatures on. Note that for the case of unique signatures, EUF-CMA security is equivalent to the (typically stronger) notion of sEUF-CMA, where the adversary wins even if it provides a *fresh* signature on a previously queried message.

Tight security reductions. To show that a digital signature scheme is secure, it is typically shown that it is at least as difficult to break EUF-CMA security as it is to solve some well-studied computational problem. Such a proof is constructed using a reduction: An efficient algorithm which can transform any efficient attacker $\mathcal{A}$ against the signature scheme into an efficient solver $\mathcal{R}^{\mathcal{A}}$ for the computational problem. Given such a reduction, if the underlying problem is indeed difficult to solve, and there exists no efficient solver, then it follows that there exists no efficient attacker against the signature scheme.

The efficiency of a reduction is measured in its *loss* or *tightness*, which captures the difference of success probability and resources needed between $\mathcal{A}$ and $\mathcal{R}^{\mathcal{A}}$. For any security parameter $\lambda \in \mathbb{N}$, let $\epsilon_{\mathcal{A}}(\lambda)$ and $t_{\mathcal{A}}(\lambda)$ denote the success probability and running time of $\mathcal{A}$, respectively. The security loss of $\mathcal{R}$ is defined as

$$\ell_{\mathcal{R}}(\lambda) = \frac{t_{\mathcal{R}^{\mathcal{A}}}(\lambda)/\epsilon_{\mathcal{R}^{\mathcal{A}}}(\lambda)}{t_{\mathcal{A}}(\lambda)/\epsilon_{\mathcal{A}}(\lambda)} = \frac{t_{\mathcal{R}^{\mathcal{A}}}(\lambda) \cdot \epsilon_{\mathcal{A}}(\lambda)}{t_{\mathcal{A}}(\lambda) \cdot \epsilon_{\mathcal{R}^{\mathcal{A}}}(\lambda)}\,.$$

If the reduction loss is constant, the reduction is called *tight*. While a general reduction simply tells us that both the scheme and the computational problem are in the same complexity class of problems, a tight reduction ensures that both the scheme and problem take the same amount of time to break or solve, up to some constant factor. We say a digital signature scheme is *tightly secure* if there exists a tight reduction from EUF-CMA security of the scheme to a hard non-interactive computational problem (also called *non-interactive complexity assumption* (NICA)).

Quantifying the tightness of a scheme is especially important in practice. If the security loss of a reduction depends on the resources available, such as the amount of signatures generated for a digital signature scheme, then this will lead to larger concrete parameters and requires anyone who instantiates the scheme to estimate the number of generated signatures a priori. For example, consider a scheme with a security loss linear in the number of signature queries, say, loss $\ell = q$ for $q = 2^{30}$. In this case, to achieve provable 128-bit security for the underlying assumption, the scheme must be instantiated with a security parameter of $\lambda = 128 + 30 = 158$, resulting in larger concrete parameters. Because of this, the study of tight signature schemes [Lub96;BR96], and deriving lower bounds on reduction losses has been a long-standing research goal.

Lower bounds on tightly unique signatures in the standard model. Tightly secure unique signature schemes turn out challenging to construct. All constructions that have been proven secure in the standard model have a security loss at least linear in the number of signature queries q [Lys02; MRV99a]. In fact, this linear loss was proven inherent in 2002 by Coron [Cor02]. Specifically, Coron proved that any reduction from EUF-CMA security to a non-interactive complexity assumption which uses the adversary as a black-box and only employs simple rewinding (i.e., rewinding the adversary to the start) must have a security loss of at least q. Morgan and Pass [MP18] later proved that even black-box reductions which can additionally rewind the adversary in a nested fashion must have a security loss of at least $\sqrt{q}$. This therefore rules out tight reductions for unique signature schemes in the standard model.

The random oracle model. The random oracle model (ROM) was introduced by Bellare and Rogaway in 1993 [BR93] as a bridge between cryptographic theory and practice. In the ROM, one assumes that all parties have query-access to a deterministic random oracle. Bellare and Rogaway argue that this more closely resembles the type of assumptions and primitives available in practice. An attacker in the ROM should be successful no matter how the oracle is instantiated. This gives the reduction additional power: A reduction can read an adversary's inputs and program the outputs dynamically, the adversary's success probability then remains unchanged as long as the random oracle distributions remain the same. The ROM is not covered by the lower bounds of Coron [Cor02] and Morgan and Pass [MP18], therefore schemes proven secure in this model may circumvent these bounds. However, creating a tight reduction in the ROM is still non-trivial. Consider the RSA-based full-domain hash scheme (RSA-FDH) introduced in the same paper by Bellare and Rogaway [BR93], which is a unique signature scheme. The authors prove RSA-FDH scheme secure in the ROM, but the reduction still has a linear loss in the number of random oracle queries (which was later improved to linear in the number of signature queries by Coron [Cor00]).

Tightly unique signatures in the ROM. A reduction with security loss sublinear in the number of signature queries was first achieved with "chain-based signatures", a construction devised by Guo et al. [GCS+17] and subsequently improved to short signatures by Shacham [Sha18]. As the name suggests, chain-based signatures essentially consist of a chain of signatures which each sign a hash of the previous

signature. Their construction incurs a security loss of $k \cdot q^{1/k}$, where k is the length of the chain. Hence, the scheme offers a trade-off between efficiency and tightness. The security loss was later improved by Guo and Susilo [GS22] to $4 \cdot q^{1/k}$ using an improved analysis of the same scheme. Therefore, to achieve a constant security loss for any polynomial $q = \lambda^c = 2^{c \log \lambda}$ number of queries, one has to choose $k \geq \log q \in \omega(\log \lambda)$.

The cost in efficiency of the chain-based signature scheme is undesirable. While the paper [GS22] provides a matching lower bound of $q^{1/k}$, where k is the number of hash-and-sign operations, this bound only applies to the special class of chain-based schemes. We therefore pose the following research question:

Does there exist a unique signature scheme with tight security which only performs a constant number of random oracle queries during signing and verifying?

Tight security in the multi-user setting. Most results on tightly secure signatures focus on the single-user setting, where the adversary tries to forge a signature relative to a single fixed public key. However, in realistic settings there might be millions or billions of users. Hence, a more realistic security notion is multi-user EUF-CMA security with corruptions (MU-EUF-CMA-C)[1], where the adversary has signature-query access with respect to several public keys. The adversary is further allowed to adaptively corrupt public keys to obtain the respective secret key, and finally has to provide a valid forgery with respect to one of the uncorrupted public keys. While EUF-CMA security and MU-EUF-CMA-C security are equivalent in an asymptotic sense, the reduction introduces a security loss that is linear in the number of users. Hence, it is an interesting question whether one can obtain a signature scheme for which the security loss is independent of the number of users (and signatures) in this setting.

It turns out that this notion is non-trivial to satisfy tightly, even without unique signatures. Bader et al. [BJL+16] proved that even one-time signature (OTS) schemes cannot be constructed tightly secure if the secret keys used by the scheme are unique or efficiently rerandomizable. Although there are several tight signature schemes in the multi-user model [Bad14;BHJ+15;GJ18;WLG+19;DGJ+21; HJK+21;HOS25], there currently is no *unique* signature construction in the literature which achieves tight security in the multi-user model with corruptions. This leads us to our second research question below.

Does there exist a unique signature scheme with tight security in the multi-user model?

1.1 Our Results

We answer the first question affirmatively, providing a signature scheme with tight security, which only performs a constant number of group operations and random oracle queries during signing and verifying. We provide a generic transformation which transforms any unique signature scheme that satisfies a much weaker security notion – random unforgeability against random-message attacks (RUF-RMA) [FHN+12] security – tightly into a unique signature scheme which satisfies EUF-CMA security. The RUF-RMA notion is weaker, as it only gives an attacker access to a signing oracle on *random message* and the attacker must forge a signature on a *randomly* selected message, rather than on a message of its choice.

Theorem 1.1 *Let* S^{in} *be a signature scheme which tightly achieves RUF-RMA security, using the* Hash-and-Subset-Sign (HaSS) *construction (Section 3), we construct a signature scheme* S *which is tightly EUF-CMA secure. This construction uses the same keys as* S^{in}*, each* S*-signature consists of at most* 3 S^{in}*-signatures and each signature is constructed and verified using at most* 6 *additional random oracle queries. Moreover, if* S^{in} *has unique signatures, then so does* S*.*

To construct a tight unique signature scheme, one therefore only has to construct a unique signature scheme which tightly satisfies RUF-RMA security. The notion is easily satisfied by well-known constructions based on RSA or CDH on bilinear groups.

Corollary 1.2 *If either the RSA assumption holds (Definition A.2), or the CDH assumption holds (Definition 5.2) with respect to their respective generators, then there exists a tightly secure unique signature scheme with EUF-CMA security. This scheme can be constructed by combining the construction from respectively Definition A.3 or Definition 5.3 with the HaSS transformation (Section 3).*

[1] In the literature, this security notion is also referred to as MU-EUF-CMA^{Corr} [BHJ+15;GJ18]

For the above scheme, both signing and verifying take at most 6 random oracle queries, signing takes 3 exponentiations for RSA and 3 pairing evaluations when based on bilinear groups. Verifying may take 3 exponentiations or up to 6 pairing evaluations plus 3 equality tests. Moreover, when using certified RSA parameters or certified bilinear groups, the signature scheme satisfies public-key verifiability Definition 2.2, i.e., it can be publicly verified that a (potentially maliciously generated) public key has unique signatures.

To achieve our second goal, we observe that when the same transformation is instantiated using bilinear pairings, we can prove multi-user security *without corruptions* for the resulting scheme. We note though that this does not carry over for RSA, as we need to rely on re-randomizability of public keys.

Corollary 1.3 *If the CDH assumption in Definition 5.2 holds with respect to some bilinear group generator, then there exists a tightly secure unique signature scheme with MU-EUF-CMA security.*

Tightly secure unique few-time signatures. We give several partial answers towards our second research question. Our results include two new constructions. Our first construction is a unique few-time signature (FTS) scheme, which is tightly secure in the multi-user model with corruptions. An FTS is a signature scheme where the scheme's parameters may depend polynomially on the amount of signatures computed. For our scheme, we have signatures consisting of $O(q \log q)$ group elements, public keys consisting of a single group element and secret keys of $O(q^2 \log q)$ group elements, where q is the number of signature queries and at least λ. This trade-off for tightness is far from ideal, but a first step towards tightly secure signatures with compact public keys and signatures.

Our technical result is that we can transform any unique signature scheme with MU-EUF-CMA security (i.e., without corruptions) into a unique FTS scheme with MU-EUF-CMA-C security. Together with Corollary 1.3 mentioned above, this gives us the first unique FTS scheme to achieve tight MU-EUF-CMA-C security.

Theorem 1.4 *Let* S^{in} *be a unique signature scheme which tightly achieves MU-EUF-CMA security, then using the CHORSS*[2] *transformation (Section 4) we can construct a unique FTS scheme* S *which is tightly MU-EUF-CMA-C secure.*

Our construction takes inspiration from existing hash-based FTS and one-time signature schemes in literature [Lam79;Per01;DSS05;RR02;BHK+19;YAG20].

Tightly secure against semi-adaptive corruptions. Our second multi-user construction is a unique signature scheme which achieves tight security in the multi-user model against semi-adaptive corruptions (MU-EUF-CMA-SAC). Our version of semi-adaptive corruptions describes a weaker security notion where an adversary can no longer corrupt parties for which it has already made signature queries.

Theorem 1.5 *Let* S^{in} *be a unique signature scheme which tightly achieves MU-EUF-CMA security, then we can construct a unique signature scheme* S *which is tightly MU-EUF-CMA-SAC secure (Appendix C).*

In contrast to our other multi-user construction, the semi-adaptive signature scheme has compact keys and signatures. Public keys are twice as large, signatures are three times as large[3] while secret keys are the size of one base secret key plus two base signatures.

1.2 Related Work

In Table 1 and Table 2, we provide a comparison of our scheme with the state of the art, focusing on signature schemes that support public-key verifiability. We note that while the list is not exhaustive, it covers the most relevant schemes in the literature in terms of the highlighted properties.

Tight non-unique constructions. In the context of digital signatures, uniqueness is often a strictly required functional property. However, if one considers relaxing the uniqueness property, tight schemes exist even in the standard model [HJ16]. In the ROM, tightly secure non-unique signature schemes were presented by Bellare et al. [BIN97] and subsequently by Bernstein [Ber08] and Abdalla et al. [AFL+12]. The multi-user model for digital signatures was first defined by Menezes and Smart [MS04], the first tight construction was found by Bader [Bad14]. Subsequent tight constructions include [BHJ+15;GJ18; WLG+19;DGJ+21;HJK+21;HOS25].

[2] The 'Chained Hash to Obtain Random Signature Subset' transformation

[3] Although strictly speaking signatures consist of three base signatures, two of the three base signatures are the same for every message and therefore only have to be sent once.

Scheme	Unique	Tight	Constant Complexity
Hash-and-Sign (FDH) [BR93]	✓	✗	✓
Salted Hash-and-Sign (PSS) [KW03;BR96]	✗	✓	✓
Chain-based [GCS+17;Sha18]	✓	✓	✗
HaSS [This work]	✓	✓	✓

Table 1: Comparison of signature schemes in the single-user setting in the ROM. We consider a scheme to have constant complexity if it has constant signature and key sizes and takes a constant number of group operations (such as exponentiation) and random oracle queries to compute and verify a signature (i.e., independent of the security parameter).

Scheme	Unique	Tight	Many-time security
[Bad14;BHJ+15;GJ18;WLG+19;DGJ+21;HJK+21;HOS25]	✗	✓	✓
CHORSS [This work]	✓	✓	✗

Table 2: Comparison of signature schemes in the multi-user setting. Here, by many-time security we denote that the scheme supports an a-priori unbounded number of signature queries (with adaptive corruption of parties).

Constructions related to HaSS. Recall that we presented our hash-and-subset-sign construction as a general transformation from a RUF-RMA secure scheme to standard EUF-CMA security. We observe that the well-known hash-and-sign transformation [BR93] can also be viewed as a transformation from RUF-RMA security to EUF-CMA, with the drawback that this transformation incurs a linear security loss due to a guessing argument. If we use RSA-based signatures as the underlying RUF-RMA secure scheme then the hash-and-sign transformation yields FDH [BR93]. If we consider a seeded variant instead (where the hash function takes a random seed as additional input), this transformation yields the PSS [BR96] scheme. We note that, interestingly, adding even a one-bit random seed allows one to obtain a tight security reduction [KW03] (as we recall in the technical overview), although both PSS variants lose the uniqueness property.

Weakening uniqueness. Our constructions are quite strong because they have perfect uniqueness and even allow for verifiable public keys to ensure uniqueness holds even with respect to maliciously generated keys. One may consider whether weakening this strong notion of uniqueness could allow one to weaken the oracle model used.

If one were to remove the public key verifiability property and construct a scheme which is only unique under honest key generation then a tight reduction is actually possible for the RSA-FDH signature scheme: The FDH signature scheme has been proven tightly secure under a weaker notion of uniqueness in the work by Kakvi and Kiltz [KK12]. In this work the authors prove FDH tightly secure based on the ϕ-hiding assumption [CMS99]. Central to their proof is the ability to initialize corrupt public keys for which signatures are not unique. Under these chosen parameters, the FDH scheme is only unique if the public keys are honestly generated. This construction is therefore unfit for any applications where uniqueness of the signatures is a required security property.

A slightly stronger notion is computational uniqueness, which states that a polynomial-time attacker should not be able to find a public key, message and a pair of distinct signatures which both verify w.r.t. said message and public key. Computational uniqueness is obtained by hash-based signature schemes (discussed in more detail below). However, this weakening of the uniqueness property does not seem to immediately lead to a tight reduction, the difficulty for hash-based signature schemes in particular is that one cannot easily re-randomize the underlying problem, a technique used for our tight security proof and many others.

Constructions related to CHORSS. Our CHORSS construction takes inspiration from existing hash-based few-time and one-time signature schemes in literature [Lam79;DSS05;Per01;RR02;BHK+19; YAG20]. These listed schemes all have public keys that consist of a list of hashes, where a signature is a selective revealing of the pre-images of these hashes. Our construction bears the closest resemblance to HORS [RR02] and DFORS [YAG20], but instead of revealing hash pre-images, we output signatures of the underlying scheme. This allows our signatures to be truly unique as long as the underlying scheme

is unique, while hash-based signatures are only computationally unique. Our construction is tightly secure, achieving a constant security loss independent of the number of users, signatures and the security parameter (although the latter two do influence its parameter choices).

Tightly secure VRFs. Verifiable random functions (VRFs) [MRV99b] are the public-key analogue of PRFs: each input has a unique output that can be proven correct with respect to the public key, and outputs remain pseudorandom as long as their proofs are not revealed. Similarly to unique signatures, VRFs with tight security are hard to achieve. Namely, Niehues [Nie21] showed that in the standard model, any reduction from a non-interactive assumption to VRF security loses a factor of roughly Q, where Q is the number of evaluation queries, extending the meta-reduction of Bader et al. [BJL+16]. As noted in [Nie21], the lower bound does not apply in the ROM, where Guo et al. [GCS+17] gave a unique signature scheme with a tight reduction, which in turn can be transformed into a tight VRF in the ROM.

Our scheme also yields a tightly secure VRF in the ROM. Namely, any unique signature scheme that satisfies uniqueness even against malicious public keys can be transformed into a VRF as follows: the proof for input x is a signature σ on x, and the output is $\mathcal{H}(\sigma)$. This transformation is folklore and underlies, e.g., the RSA-FDH-based VRF of [GNP+15]. This transformation is tight: pseudorandomness reduces to unforgeability without loss. The reduction answers evaluation queries using its signing oracle and returns a uniformly random value as the challenge output for x^*. As $\mathcal{H}$ is a random oracle, this is indistinguishable from $\mathcal{H}(\sigma^*)$ unless the adversary queries $\mathcal{H}$ on σ^*. In this case, the reduction finds σ^* by running signature verification on all of the adversary's $\mathcal{H}$ queries, which yields a forgery on x^*, as x^* was never queried to the evaluation oracle. Applied to our scheme, we obtain a VRF with unique proofs and tight security that requires only a constant number of random oracle calls for evaluation and verification. In comparison, the same transformation applied to the scheme of Guo et al. [GCS+17] gives a loss of $k \cdot q^{1/k}$, which is only small if the number n of chained random oracle calls (and proof elements) grows logarithmically in q.

We note that the standard notion of VRFs only requires the output to be unique, not the proof. If one is fine with non-unique proofs, it might be possible to obtain tightly secure VRFs via a different route, although we are not aware of any explicit construction in literature for which a tight security proof in the ROM was given.

1.3 Technical Overview

Problem setting. Recall the EUF-CMA security game with respect to some signature scheme $\mathsf{S} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$. The adversary knows the algorithm definition of each algorithm $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$, and initially receives a public key pk. The only secret information is the corresponding secret key sk. The adversary is given a signing oracle, which it can query on any message m to receive a signature using the secret key sk. It can query this oracle q times. Finally, the adversary wins the EUF-CMA security game, if it can produce a signature on a message which it did not query to the signing oracle with. Let $\varepsilon_{\mathcal{A}}$ be the success probability of such an adversary $\mathcal{A}$.

Because we are in the random oracle model, the signature scheme is allowed to depend on some hash function $\mathcal{H}$. However, during the security game, the adversary does not get a description of this hash function, instead it only has oracle access to it. Furthermore, the hash function is instantiated using lazy sampling: namely we only define the result of a hash $\mathcal{H}(x)$ when it is queried for the first time, we define its result by sampling an output uniformly at random and storing this result in a table. A consequence of the random oracle model is that the adversary is no longer able to compute each of the algorithms $\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver}$ locally. Instead, it must query the random oracle wherever it is used in the respective algorithms.

Tight security reductions. Our goal when constructing a signature scheme is to prove its security using a reduction. A reduction is an algorithm which uses EUF-CMA adversaries against our signature scheme to break an underlying problem itself. The reduction is able to run the EUF-CMA adversary $\mathcal{A}$ using any input it chooses, during this execution the reduction controls the adversary's oracles. It can read the oracle inputs and choose the respective outputs.

This definition has the implicit restriction that the adversary should not be able to distinguish the real and simulated oracle outputs. If the adversary notices that the reduction gives incorrect signatures for example, then it could choose to not produce a forgery in the end. Due to this restriction, the powers

of the reductions we create are restricted to choosing the public key strategically and controlling the random oracle. Where both must preserve their respective distribution from the adversary's perspective.

Typically, we only consider a scheme tightly secure if there exists a tight reduction to a non-interactive complexity assumption. A non-interactive complexity assumption means that the underlying problem does not require interaction between the solver and the problem, such as access to oracles. This rules out trivial tight reductions, as any scheme can be tightly reduced to its own security. Recall that we further only consider a reduction tight if the success probability of the reduction is at least $\varepsilon_{\mathcal{A}}/c$ for some constant $c > 0$.

Unique signatures via hash-and-sign [BR93]. We first recall the hash-and-sign method to create a unique but non-tight signature scheme. Here, a message is first hashed, and the hashed message then signed using the secret key. Such a signature can be created using for example the RSA trapdoor permutation [RSA78]. An RSA public key consists an RSA modulus $N = p \cdot q$ for p, q prime and an exponent e. This gives a permutation $f(x) = x^e \mod N$. To compute the inverse of this permutation, one needs the corresponding secret key: the inverse exponent $d = e^{-1} \mod (p-1)(q-1)$. Using RSA-based hash-and-sign yields us the full-domain hash (FDH) unique signature scheme [BR93]. Here, signatures are of the form $\sigma = f^{-1}(\mathcal{H}(m))$ and can be verified by checking if $f(\sigma) = \mathcal{H}(m)$.

A reduction is able to prove hash-and-sign based signature schemes secure by "programming" the random oracle. In our FDH example the reduction would receive a problem (N, e, y) from the non-interactive RSA game, where (N, e) is an RSA public key and y is a group element $\mod N$. The reduction wins this game if it can find a group element z such that $z^e = y$. The reduction will send (N, e) as public key to the adversary. For each query to the random oracle, the reduction considers two options (1) the reduction could program the oracle as $\mathcal{H}(m) = r^e$ for some random group element r. This would allow the reduction to answer respective signature queries, or (2) the reduction could program the oracle as $\mathcal{H}(m) = y \cdot r^e$, if the reduction receives a signature on this message, then it can use this to find an element z such that $z^e = y$ and solve its own underlying problem. The most efficient choice here is to use method (2) only for a fraction $1/q$ of all oracle queries and (1) for everything else. This gives the optimal probability that the reduction is able to answer all signature queries, and solve its own problem using the resulting forgery. This method was proven optimal for FDH by Coron [Cor00][Cor02]. However, the success probability of this reduction is $\Theta(\varepsilon_{\mathcal{A}}/q)$ and therefore not tight.

This example demonstrated the main technique that reductions of unique signature schemes in the ROM use: The reduction can program the random oracle to either choose a hash with a known solution/signature using method (1), or it can embed its underlying problem (2). The reduction fails if either the adversary succeeds without solving an embedded problem, or if the adversary queries its oracle on an input which would force the reduction to solve its own embedded problem to answer.

Tight and unique signatures via chain-based signatures [GCS+17;GS22]. Next we recall the chain-based signature, which achieves tight unique signature schemes, albeit at higher verification and signing costs. Each signature in this construction can be viewed as a chain where each link in said chain is a signature of a hash of the message and all previous signatures in the chain. Immediately we see that the reduction has a lot more flexibility in which hash it wishes to embed its underlying problem. This allows the reduction to perform a much more aggressive embedding strategy without too much of an increase in the probability that it must solve its own embedded problem. This leads the reduction to its loss of $q^{1/k}$, which is tight for $k = \omega(\log \lambda)$, where k is the chain length.

Tight signatures via salted hash-and-sign [BR96;KW03]. An alternative method to give the reduction additional power is to sacrifice uniqueness. In the following construction, each message has exactly two signatures. The reduction can then hide a challenge in exactly one of the two options, while always giving out the other signature in signature queries. Moreover, with probability 1/2 the forgery provided by the adversary will solve the reduction's challenge, allowing it to solve its underlying computational problem.

We describe the signature scheme by Katz and Wang [KW03] in the following. This scheme is a variant of the probabilistic signature scheme (PSS) [BR96], which instead uses a random seed of length 1. As such we will also refer to it as 1-PSS. The signing algorithm for this scheme first computes a hash $\mathcal{H}_2(\mathsf{sk}, m)$ to get a single bit b as output. The resulting signature is simply an RSA signature on the hash $\mathcal{H}_1(m||b)$. To verify such a signature, σ one checks if $\sigma^e = \mathcal{H}_1(m||b)$ for either $b = 0$ or $b = 1$. Each message now has two possible signatures, but an adversary cannot detect which one will be the result of a signing query without computing $\mathcal{H}_2(\mathsf{sk}, m)$ using the secret key.

The reduction defined by Katz and Wang [KW03] will now embed its problem in either the hash $\mathcal{H}_1(m||0)$ or $\mathcal{H}_1(m||1)$, while it defines the other hash to have a known solution it created itself. This enables the reduction to answer any signature query by outputting the prepared solution. Finally, the forgery provided by the adversary has a probability of $1/2$ to solve the hidden challenge and therefore the underlying RSA problem.

Another way to view the gained strength of the reduction is that we have weakened the power of signature queries. For the 1-PSS scheme the adversary is not certain which of two hashes it will receive a signature on.

Our solution: tight, unique, and constant complexity. We now present our solution, which circumvents the shortcomings of the above constructions. Our construction achieves the same property as 1-PSS, where the adversary is not certain which hashes it will receive a signature on from a signature query. However, our construction is able to achieve this property *without losing uniqueness*. We present an RSA-based variant of our scheme.

To sign a message m in our RSA-based Hash-and-Subset-Sign (HaSS) scheme, the signing algorithm computes $\mathcal{H}_1(\mathsf{pk}, m) \in \mathbb{Z}_N^3$ (i.e., we choose a hash function with larger output)[4] and splits the output into three hashes (h_1, h_2, h_3). Each of these hashes is then exponentiated using the RSA secret key, thereby creating three sub-signatures $(\sigma_1, \sigma_2, \sigma_3) = (h_1^d, h_2^d, h_3^d)$. Next, the signing algorithm queries a different hash function $\mathcal{H}_2$ on each of these signatures. However, this hash function only outputs a single bit $b_i = \mathcal{H}_2(\mathsf{pk}, i, m, \sigma_i)$. Let $b = (b_1, b_2, b_3)$ be a tuple of these three bits, the following rules decide which subset of $(\sigma_1, \sigma_2, \sigma_3)$ will be the resulting signature.

- The signature $(\sigma_1, \sigma_2, \sigma_3)$ is valid if $b = (0, 0, 0)$ or $b = (1, 1, 1)$,
- the signature $(\perp, \sigma_2, \sigma_3)$ is valid if $b = (?, 0, 1)$,
- the signature $(\sigma_1, \perp, \sigma_3)$ is valid if $b = (1, ?, 0)$, and
- the signature $(\sigma_1, \sigma_2, \perp)$ is valid if $b = (0, 1, ?)$.

We use a question mark to denote that the case holds regardless of whether that bit is 0 or 1. The verification algorithm can verify a signature by checking for each i that $\sigma_i^e = h_i$ (if σ_i is included in the subset). Furthermore, for all cases above it is easy to verify that a signature is in said case using only the sub-signatures and respective bits which are present in the signature. Indeed, the only bit which the verification algorithm cannot compute in each case is the one marked with "?". Uniqueness follows from the uniqueness of the RSA signatures and the fact that each of the cases above are mutually exclusive, each pair of cases has at least one bit on which they disagree.

The essential property needed for our reduction to work is that an attacker cannot predict which of the RSA exponentiations are needed for a signature before computing them. Our reduction will then always embed its problem in exactly the hashes which we do not need to exponentiate for a signature.

In the main body of this paper we will prove tightness for the general transformation, and we will provide a concrete instantiation using bilinear pairings.

1.4 Technical Overview: Multi-User Security

We now consider a stronger security notion: EUF-CMA security in the multi-user model with adaptive corruptions (MU-EUF-CMA-C). In this multi-user security game the adversary will instead receive N different public keys, and it may ask signature queries with respect to any public key. The adversary wins the security game if it forges a new signature with respect to any public key. Additionally, the adversary is able to perform "corruption queries": when it corrupts a public key the adversary will receive the corresponding secret key. The eventual forgery must be for an uncorrupted user.

Because this security notion is stronger, it is also more difficult to instantiate tightly. The naive solution would be to simply use a single-user signature scheme and during the reductions guess which keys will be corrupted by the adversary. Due to this guessing, however, such a reduction would have loss N, where N is the number of users (i.e., public keys). In the standard model this loss of N is inherent [BJL+16]. The difficulty is similar to achieving tight single-user signatures in the standard model: Either the reduction knows or does not know a secret key which verifies. Similarly, in the single-user setting, either the reduction knows or does not know a signature which verifies. This leads us to

[4] Note that if accounting for this larger output of the hash function by three calls to a single hash function, we get to a total of 6 RO calls in our construction.

conclude that our only hope for a tight reduction is to make the correctness of a secret key depend on some random oracle.

A tight multi-user FTS. For an a-priori fixed q we can design a few-time-signature (FTS) scheme, which can sign q messages and has security loss 4 no matter the number of users N. We present our scheme as a general construction over some inner unique signature scheme, which in particular must be tightly secure in the multi-user setting without corruptions. We can in fact construct this inner scheme using our single-user construction above. Such a generalization from single user to multi-user security without corruptions is generally possible whenever the underlying challenge can be re-randomized (which is the case for CDH but not RSA). The reduction can then initialize each public key based on a different rerandomized CDH challenge, while forgery for any of these public keys allows us to solve the original.

Each secret key in the outer scheme consists of L inner signatures of the numbers 1 to L, where L is polynomial in q. An outer signature now consists of a selection of these inner signatures in the outer secret key. On a high level, this construction ensures that revealing a secret key is equivalent to revealing several inner signatures, and forging an outer signature is also equivalent to forging an inner signature. This allows us to construct a tight reduction to the security of the inner scheme. Because the security of the scheme relies on each signature revealing a new part of the secret key, this construction immediately forces us to grow the secret key with q.

In the proof itself we show that it is hard to find a message m for which one does not need any new inner signatures (i.e. where all inner signatures needed have already been revealed in previous signature queries). This implies that a forger which forges an outer signature, needs to forge at least one inner signature. This, in turn, allows us to reduce to the unforgeability of the inner scheme. Essential to this construction is that we never have to give out the inner secret key, only inner signatures.

1.5 Paper Structure

In Section 2 we will go over some definitions and our security notions. Next we prove our main result: A generic tight signature construction in Section 3. In Section 4 we also provide a unique signature scheme which is tightly secure in the multi-user setting. This result relies on the construction of the previous section. In Section 5 we provide an instantiation of our schemes using bilinear groups. In the appendix, we provide an RSA instantiation in Appendix A, a tight MU-RUF-RMA proof for the scheme based on CDH on bilinear groups in Appendix B, and finally an additional multi-user secure scheme in Appendix C.

2 Definitions and Security Notions

We mostly consider probabilistic polynomial time (PPT) algorithms. A probabilistic algorithm A has access to a string of random coins, for an input x we may write $y \leftarrow\!\$\ A(x)$ to denote that y is sampled as a random output of A by running A on x and a string of uniform random coins. For a set U we may also write $x \leftarrow\!\$\ U$ for sampling an element of the set uniformly at random. Algorithms may have oracle access to other algorithms, we write $A^{\mathcal{O}(\cdot)}$ to denote that A may query $\mathcal{O}$ for any input, during its execution. We write $[n]$ for the set $\{1, 2, \ldots, n\}$.

2.1 The Random Oracle Model

A truly random function must have the form of a random table and therefore requires exponential storage space. There also exist (compact) functions for which we believe that the output is close to random. The random oracle model is an idealized version of such random functions and assumes that everyone has access to a deterministic random function whose outputs are indistinguishable from random. In security reductions, this random oracle is often emulated using lazy sampling, where each oracle output is only sampled uniformly at random on its first query. This output is then stored in a table such that the same result can be output on subsequent queries.

For algorithm definitions, we simply consider the random oracle as a deterministic function which maps bit strings of arbitrary length to some specified range, we write $\mathsf{RO} : \{0,1\}^* \to \mathcal{Y}$. Algorithms which use a random oracle in their definition should be correct for any choice of the RO function and should be secure for any oracle chosen uniformly at random from the set of functions $\{0,1\}^* \to \mathcal{Y}$.

We restrict any algorithm to use only a single random oracle RO, an algorithm which queries two RO's $\mathcal{H}_1(x)$ and $\mathcal{H}_2(x)$ can be transformed into an algorithm which using a single RO with an additional

input specifying which RO to query, i.e. the example queries would become $\mathcal{H}(1,x)$ and $\mathcal{H}(2,x)$. For correctness, it is important that we then use an unambiguous encoding to map these inputs to different bit strings. If we need two different output ranges $\mathcal{Y}_1, \mathcal{Y}_2$ then we can define our RO to have range $\mathcal{Y}_1 \times \mathcal{Y}_2$ and use the left or right output depending on which we need.

2.2 Signature Schemes

Definition 2.1 *A signature scheme* $(\mathsf{Setup}, \mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ *consists of the following polynomial time algorithms:*

- *A setup algorithm for public parameters* $\mathcal{PP} \leftarrow\!\!\$\ \mathsf{Setup}(1^\lambda)$,
- *a key generation algorithm* $(\mathsf{pk}, \mathsf{sk}) \leftarrow\!\!\$\ \mathsf{Gen}(1^\lambda, \mathcal{PP})$,
- *a signing algorithm* $\sigma \leftarrow \mathsf{Sign}(\mathcal{PP}, \mathsf{sk}, m)$, *and*
- *a deterministic verification algorithm* $b \leftarrow \mathsf{Ver}(\mathcal{PP}, \mathsf{pk}, m, \sigma)$.

We will often omit the public parameters $\mathcal{PP}$ *from the input of* Sign *and* Ver*, as which public parameters are used is often clear from context.*

We only consider signature schemes which are *perfectly correct* which means that for all λ, for all $\mathcal{PP} \leftarrow\!\!\$\ \mathsf{Setup}(1^\lambda)$, for all $(\mathsf{pk}, \mathsf{sk})$ in the image of $\mathsf{Gen}(1^\lambda, \mathcal{PP})$ and for all messages $m \in \{0,1\}^*$, we have that

$$1 \leftarrow \mathsf{Ver}(\mathsf{pk}, m, \mathsf{Sign}(\mathsf{sk}, m))$$

Definition 2.2 (Uniqueness (and public key verifiability)) *A signature scheme* $(\mathsf{Setup}, \mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ *has unique signatures if for all* $\mathcal{PP}, \mathsf{pk}, m$ *we have that there exists at most one* σ *such that*

$$1 \leftarrow \mathsf{Ver}(\mathcal{PP}, \mathsf{pk}, m, \sigma).$$

Implicit in this definition is that it is impossible to maliciously construct a combination of $\mathcal{PP}, \mathsf{pk}, m$ which allows for two correct signatures. We sometimes mention this second property explicitly as *Public key verifiability*. This property holds for all of our constructed signature schemes

We also allow all of these algorithms to have access to a random oracle function. However, we require all signature schemes to be perfectly correct and unique for any fixed choice of random oracle. We may write $(\mathsf{Setup}^{\mathsf{RO}(\cdot)}, \mathsf{Gen}^{\mathsf{RO}(\cdot)}, \mathsf{Sign}^{\mathsf{RO}(\cdot)}, \mathsf{Ver}^{\mathsf{RO}(\cdot)})$, to bring attention to this dependency.

We now introduce our main security notion: existential unforgeability under chosen message attacks (EUF-CMA) and its multi-user (MU) variants.

Definition 2.3 *For* $xxx \in \{$*EUF-CMA, MU-EUF-CMA, MU-EUF-CMA-C*$\}$ *we say a signature scheme is xxx-secure for some* $q \geq 1, N \geq 1$ *if any adversary has at most negligible advantage against its corresponding experiment given in Figure 1. More precisely there exists some negligible function* negl *such that for all PPT adversaries* $\mathcal{A}$ *we have that*

$$\mathsf{Adv}^{xxx}_{\mathcal{A}}(\lambda, q, N) = \Pr[\mathsf{Exp}^{xxx}_{\mathcal{A}}(1^\lambda, q, N) = 1] \leq \mathsf{negl}(\lambda),$$

where the probability is taken over the random choice of RO*, the random coins of the experiment and the random coins of* $\mathcal{A}$*. For* $xxx =$ *EUF-CMA we may omit* N *and write* $\mathsf{Exp}^{EUF\text{-}CMA}_{\mathcal{A}}(1^\lambda, q)$.

In this security definition we have explicitly defined the random oracle using lazy sampling. As this definition is the same for all of our security experiments we will omit it and the corresponding table T in the future. Furthermore, we have written the range of the random oracle as a generic $\mathcal{Y}$ the specific range is defined by the scheme itself. If a scheme uses more than one random oracle we can still model it by combining the oracle domains and ranges and by using an additional input to specify which oracle one intends to query.

Next we introduce a weaker variant of EUF-CMA security named random unforgeability under random message attacks (RUF-RMA). Our definition differs slightly from the standard notion in literature as we allow the adversary a challenge oracle, as opposed to giving the adversary only one challenge message.

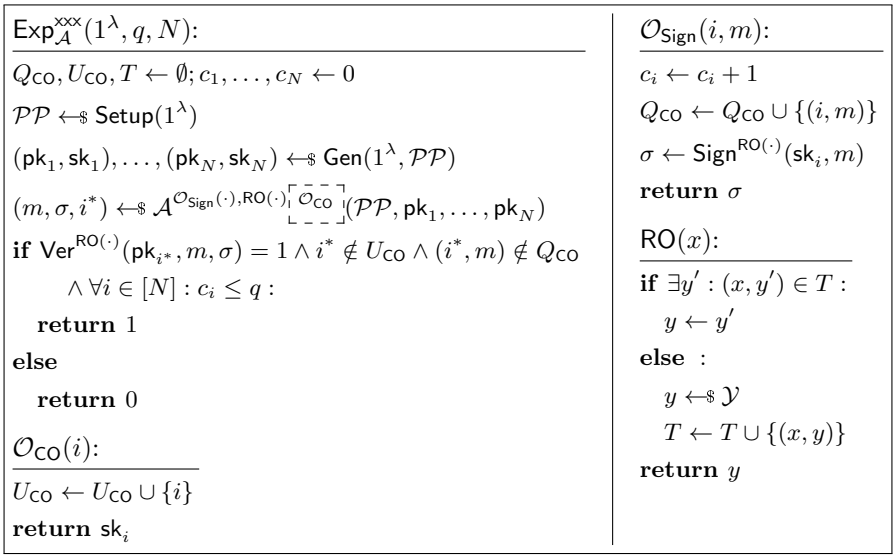

Figure 1: The MU-EUF-CMA security experiment, if the code in the dashed box is included (the corruption oracle) we call it MU-EUF-CMA-C instead. The EUF-CMA security experiment is a special case of the MU-EUF-CMA experiment where $N = 1$. For the EUF-CMA experiment we may omit the number i from the signing oracle.

Definition 2.4 (Random Unforgeability under Random Message Attacks) *A signature scheme is xxx-secure, for xxx* $\in \{$*MU-RUF-RMA, RUF-RMA*$\}$*, if there exists no polynomial-time adversary* $\mathcal{A}$ *which achieves non-negligible advantage against the corresponding security game depicted in Figure 2. More precisely there exists some negligible function* negl *such that for all* $\mathcal{A}$ *we have that*

$$\mathsf{Adv}^{\mathsf{xxx}}_{\mathcal{A}}(\lambda, q, N) = \Pr[\mathsf{Exp}^{\mathsf{xxx}}_{\mathcal{A}}(1^\lambda, q, N) = 1] \leq \mathsf{negl}(\lambda),$$

where the probability is taken over the random choice of RO*, the random coins of the experiment and the random coins of* $\mathcal{A}$*.*

3 A Tight and Efficient Unique Signature Scheme in the ROM

The technical result of this paper is that we can transform any RUF-RMA secure signature scheme into an EUF-CMA secure signature scheme, while preserving uniqueness. We call the original scheme and the resulting transformed scheme the *inner* and *outer* signature scheme respectively from now on. This transformation can be done in such a way that the EUF-CMA security of the outer scheme can be tightly reduced to the RUF-RMA security of the inner scheme. It is much easier to tightly instantiate the RUF-RMA security notion as it considers much weaker adversaries. We will provide such a tight instantiation in Section 5. With this instantiation, we have constructed the first unique signature scheme with a tight reduction to a non-interactive complexity assumption, where the scheme uses a constant number of group operations and in particular a constant number of random-oracle queries.

To achieve a tight scheme, our corresponding reduction needs to be able to answer any signature query, but we must also be able to use the adversary's forgery to gain some new information (which most likely will lead to solving an underlying problem) with constant probability. This leads to an apparent contradiction where we are able to produce signatures, but we still gain new information when similar signatures are forged. The core of our scheme is how it achieves both of these properties.

The intuition behind the scheme depicted in Figure 3 is that we want there to be a constant probability that the signer performs more work than is strictly necessary to compute a signature. Specifically, our (*outer*) scheme is a wrapper around some *inner* signature scheme, and each outer signature consists of three inner signatures. However, with some probability only two out of three inner signatures are needed for the outer signature. With this setup, a forger has a small probability to compute an inner signature which is not strictly necessary. Using the random oracle model we will then ensure that this information solves some underlying problem. Below, we will give a more precise description of our contribution: the Hash-and-Subset-Sign (HaSS) scheme.

$\mathsf{Exp}^{\text{(MU)-RUF-RMA}}_{\mathcal{A}}(1^\lambda, q, N)$:

$Q_{\mathsf{CH}} \leftarrow \emptyset; T \leftarrow \emptyset; c \leftarrow 0$

$\mathcal{PP} \leftarrow\!\$\ \mathsf{Setup}(1^\lambda)$

$(\mathsf{pk}_1, \mathsf{sk}_1), \ldots, (\mathsf{pk}_N, \mathsf{sk}_N) \leftarrow\!\$\ \mathsf{Gen}(1^\lambda, \mathcal{PP})$

$(m, \sigma, i^*) \leftarrow\!\$\ \mathcal{A}^{\mathcal{O}_{\mathsf{CH}}(\cdot), \mathcal{O}_{\mathsf{CO}}(\cdot)}(\mathcal{PP}, \mathsf{pk}_1, \ldots, \mathsf{pk}_N)$

if $(i^*, m) \in Q_{\mathsf{CH}} \wedge \mathsf{Ver}(\mathsf{pk}_{i^*}, m, \sigma) = 1$
$\quad \wedge\, c \leq q$:
output 1
else
output 0

$\mathcal{O}_{\mathsf{CH}}(i)$:

$c \leftarrow c + 1$
$x \leftarrow\!\$\ \mathcal{X}$
$Q_{\mathsf{CH}} \leftarrow Q_{\mathsf{CH}} \cup \{(i, x)\}$
return x

$\mathcal{O}_{\mathsf{CO}}(i)$:

$c \leftarrow c + 1$
$x \leftarrow\!\$\ \mathcal{X}$
$\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}_i, x)$
return (x, σ)

Figure 2: The MU-RUF-RMA experiment, the special case where $N = 1$ is the RUF-RMA experiment. For the RUF-RMA experiment we may often omit the input/output i when it refers to a party, as this value would always equal one. The definition of the random oracle and any dependency on it is allowed but is omitted for simplicity. In this experiment, the set $\mathcal{X}$ is the message space of the signature scheme.

The HaSS Transformation. Let $\mathsf{S}^{\mathsf{in}} = (\mathsf{Setup}^{\mathsf{in}}, \mathsf{Gen}^{\mathsf{in}}, \mathsf{Sign}^{\mathsf{in}}, \mathsf{Ver}^{\mathsf{in}})$ be a unique signature scheme, with finite message space X and signatures in some set Y. We name the scheme S^{in} the *inner* signature scheme. Let $\mathcal{H}_1 : \{0,1\}^* \to X^3$ and $\mathcal{H}_2 : \{0,1\}^* \to \{0,1\}$ be two hash functions modelled as random oracles. Our *outer* signature scheme S consists of the algorithms $(\mathsf{Setup}, \mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ which we will each describe below:

- The key generation algorithm Gen and setup algorithm Setup are identical to $\mathsf{Gen}^{\mathsf{in}}$ and $\mathsf{Setup}^{\mathsf{in}}$ respectively.
- The signing algorithm $\mathsf{Sign}(\mathsf{sk}, m)$, first computes $(h_1, h_2, h_3) \leftarrow \mathcal{H}_1(\mathsf{pk}, m)$ and then computes both $\sigma_i \leftarrow \mathsf{Sign}^{\mathsf{in}}(\mathsf{sk}, h_i)$ and $b_i \leftarrow \mathcal{H}_2(\mathsf{pk}, i, m, \sigma_i)$ for each $i \in \{1, 2, 3\}$. If $b_1 = b_2 = b_3$ then we output $(\sigma_1, \sigma_2, \sigma_3)$. Otherwise, there exists exactly one $i \in \{1, 2, 3\}$ such that both $b_i = 0$ and $b_j = 1$ for $j = i + 1 \bmod 3$. We now output the same tuple but replace σ_k with $\perp$ for $k = i + 2 \bmod 3$.
- The verification algorithm $\mathsf{Ver}(\mathsf{pk}, m, \sigma)$ simply confirms that each σ_i is a signature of h_i unless $\sigma_i = \perp$, and that $\sigma_i = \perp$ if and only if the other signatures are not $\perp$, and we have $\mathcal{H}_2(\mathsf{pk}, i + 1, m, \sigma_{i+1}) = 0$ and $\mathcal{H}_2(\mathsf{pk}, i + 2, m, \sigma_{i+2}) = 1$ (where again we perform addition modulo 3).

Lemma 3.1 *The above unique signature scheme is perfectly correct, for any choice of deterministic hash functions $\mathcal{H}_1, \mathcal{H}_2$ we have that for any $\mathcal{PP} \leftarrow\!\$\ \mathsf{Setup}(1^\lambda)$, any $(\mathsf{pk}, \mathsf{sk}) \leftarrow\!\$\ \mathsf{Gen}(1^\lambda, \mathcal{PP})$ and for any m we have that*

$$\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, m) \iff 1 \leftarrow \mathsf{Ver}(\mathsf{pk}, m, \sigma).$$

Proof. This follows directly from checking each of the cases for all choices of bits b_i. □

The reason our scheme allows for tight reductions in the ROM, is because a reduction can program the random oracle such that it precisely knows the inner signatures needed for the outer signature, while it can hide its underlying challenge in the inner signature which is not needed. Recall the RUF-RMA security experiment 2.4. An RUF-RMA adversary has access to two oracles: a corruption oracle $\mathcal{O}_{\mathsf{CO}}()$ which outputs message-signature pairs for randomly sampled messages and a challenge oracle $\mathcal{O}_{\mathsf{CH}}()$ which samples messages uniformly at random from a domain $\mathcal{X}$. An adversary beats the security game if it can forge a signature for a message output by the challenge oracle. This security notion formalizes the ability to hide challenges and corruptions in hash outputs. When the reduction outputs a uniformly random hash, it can either be the result from the corruption oracle (for which we know its corresponding inner signature) or the result from the challenge oracle (where providing a signature is equivalent to beating the security game).

An EUF-CMA adversary to the outer scheme now only has oracle access to the hash function and can therefore not determine which inner signature is necessary without querying $\mathcal{H}_2$ using the correct inner signature. With constant probability, the first hash query it makes for an inner signature it computed itself, may be one of our planted challenges. This therefore allows a reduction to achieve a constant success probability.

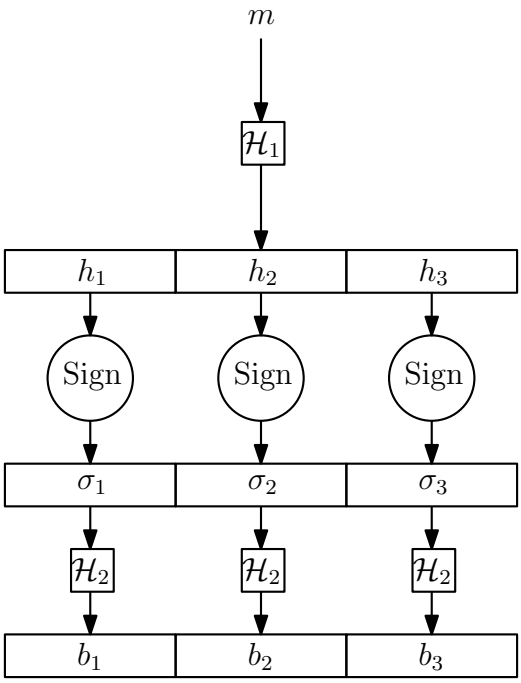

Figure 3: The Computation of h_i, σ_i and b_i, for $i \in \{1, 2, 3\}$ in the outer signature algorithm.

Lemma 3.2 *Let* $(\mathsf{Setup}^{\mathsf{in}}, \mathsf{Gen}^{\mathsf{in}}, \mathsf{Sign}^{\mathsf{in}}, \mathsf{Ver}^{\mathsf{in}})$ *be a unique signature scheme and let* $(\mathsf{Setup}, \mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ *be the outer signature scheme constructed using* S^{in} *as an inner signature scheme, following the HaSS (Section 3) transformation. If the inner signature scheme* S^{in} *is RUF-RMA secure, then* S *is EUF-CMA secure, where for both schemes we need to perform approximately the same amount of work to beat their respective security games. Specifically, for any PPT EUF-CMA adversary* $\mathcal{A}$ *against* S *there exists a PPT RUF-RMA adversary* $\mathcal{R}^{\mathcal{A}}$ *such that*

$$\Pr[\mathsf{Exp}^{EUF\text{-}CMA}_{\mathcal{A}}(1^\lambda, q) = 1] \leq 4 \cdot \Pr[\mathsf{Exp}^{RUF\text{-}RMA}_{\mathcal{R}^{\mathcal{A}}}(1^\lambda, 3q_{\mathsf{RO}}) = 1]$$

where the randomness is taken over the random choice of RO *and the random coins of* $\mathcal{A}$, $\mathcal{R}^{\mathcal{A}}$ *and the security games. The variables* q *and* q_{RO} *denote the amount of signature queries and random oracle queries made by* $\mathcal{A}$ *respectively. Additionally, the algorithm* $\mathcal{R}^{\mathcal{A}}$ *is PPT and only runs* $\mathcal{A}$ *once.*

Proof. Our goal is to transform the EUF-CMA adversary $\mathcal{A}$ into a RUF-RMA adversary against the scheme S^{in}. For this we create a reduction $\mathcal{R}$ which is an adversary against the RUF-RMA game and therefore has access to its oracles. This reduction then pretends to be the EUF-CMA security game and runs $\mathcal{A}$ using oracles controlled by $\mathcal{R}$. After recalling the relevant oracles, we will describe the reduction, then we argue that $\mathcal{A}$ cannot detect that it is not in the true security game, and finally we will argue about the success probability of the reduction.

Recall. The reduction must provide the adversary with a public key pk after which it must simulate a signing oracle $\mathsf{Sign}(\mathsf{sk}, \cdot)$ and the random oracle RO. The reduction itself will receive a public key from the underlying RUF-RMA security game, it has access to a corruption $\mathcal{O}_{\mathsf{CO}}$ and a challenge oracle $\mathcal{O}_{\mathsf{CH}}$. Both oracles sample their output uniformly at random, only the former oracle additionally provides a corresponding signature.

The reduction beats the RUF-RMA security game specifically if it outputs a signature forgery on a message $x \in X$ which resulted from a query to the challenge oracle. Throughout this proof, for most oracles we will write Q_X where X is an oracle name to denote the set of queries made to the oracle X. For the corruption and challenge oracle we will instead write Q_{CO} and Q_{CH} to denote the set of randomly sampled group elements.

Reduction. First, assume that $\mathcal{A}$ will always query $\mathcal{H}_1(\mathsf{pk}, m)$ before invoking $\mathcal{O}_{\mathsf{Sign}}(m)$, querying $\mathcal{H}_2(\mathsf{pk}, i, m, \sigma_i)$ or outputting a forgery on m. We can make this assumption without loss of generality as we could simply make a wrapper around $\mathcal{A}$ to make these additional queries. Upon a query of $\mathcal{H}_1(\mathsf{pk}, m)$ the reduction first samples bits $b_i \in \{0, 1\}$ for $i \in \{1, 2, 3\}$ uniformly at random. These bits will become the results of the $\mathcal{H}_2(\mathsf{pk}, i, m, \sigma_i)$ queries regardless of the eventual values of the σ_i, this

allows the reduction to already determine which inner signatures σ_i are needed for the outer signature on the message m. For each i, if σ_i would be needed for the final signature the reduction queries $(x, y) \leftarrow\!\$\ \mathcal{O}_{\mathsf{CO}}()$, this is used to define $h_i := x$, $\sigma_i := y$, and $\mathcal{H}_2(\mathsf{pk}, i, m, \sigma_i) := b_i$. If on the other hand, σ_i would not be needed in the outer signature, then the reduction will instead query its challenge oracle $x \leftarrow\!\$\ \mathcal{O}_{\mathsf{CH}}()$ and defines $h_i := x$. Finally, the reduction outputs the RO response as (h_1, h_2, h_3). If the adversary later queries $\mathcal{O}_{\mathsf{Sign}}(m)$, then the reduction can answer this signature query correctly as it knows exactly the σ_i needed.

We briefly argue that all random variables involved are distributed identically to how they would be in the EUF security experiment. Clearly the key generation algorithm is identical. Furthermore, to sample the outputs of $\mathcal{H}_1$ we query $\mathcal{O}_{\mathsf{CH}}$ and $\mathcal{O}_{\mathsf{CO}}$. In the definitions of these respective oracles the output group element x is sampled independently and uniformly at random. Therefore the output of the hash function remains independent and uniformly at random.

Whenever $\mathcal{A}$ makes a query of the form $\mathcal{H}_2(\mathsf{pk}, i, m, \sigma_i)$ the reduction finds $\mathcal{H}_1(\mathsf{pk}, m) = (h_1, h_2, h_3)$ and checks whether σ_i is a signature for h_i. If this holds then there are two possible cases (1) h_i was output by $\mathcal{O}_{\mathsf{CH}}()$ or (2) by $\mathcal{O}_{\mathsf{CO}}()$. Which case we are in is determined by the value of each $\mathcal{H}_2(\mathsf{pk}, j, m, \sigma_j)$ for $j \in \{1, 2, 3\}$. In the former case (1), the reduction can use this information to beat the RUF-RMA security game, and at the end of the computation it will output (h_i, σ_i). We call this event $\mathbf{W}$, and conclude that the event $\mathbf{W}$ causes $\mathcal{R}^{\mathcal{A}}$ to beat the RUF-RMA security game.

Analysis. Our goal is to argue that the probability of $\mathbf{W}$ occurring is high. First we assume without loss of generality that the adversary has queried $\mathcal{H}_2(\mathsf{pk}, i, m, \sigma_i)$ for all σ_i in the eventual forgery. Let (m^*, σ^*) be the forgery produced by $\mathcal{A}$ and $Q_{\mathcal{H}_2}$ be the set of random oracle queries to $\mathcal{H}_2$. We find that

$$\begin{aligned}
&\mathsf{Ver}(\mathsf{pk}, m^*, \sigma^*) = 1 \wedge m^* \notin Q_{\mathsf{Sign}} \\
\implies &\exists \sigma_i \in \sigma^* \text{ s.t. } \mathsf{Ver}^{\mathsf{in}}(\mathsf{pk}, \mathcal{H}_1(\mathsf{pk}, m^*)_i, \sigma_i) = 1 \wedge m^* \notin Q_{\mathsf{Sign}} \\
\implies &\exists (\mathsf{pk}, i, m^*, \sigma_i) \in Q_{\mathcal{H}_2} \text{ s.t. } \mathsf{Ver}^{\mathsf{in}}(\mathsf{pk}, \mathcal{H}_1(\mathsf{pk}, m^*)_i, \sigma_i) = 1 \wedge m^* \notin Q_{\mathsf{Sign}},
\end{aligned}$$

which implies

$$\Pr[\exists (\mathsf{pk}, i, m, \sigma_i) \in Q_{\mathcal{H}_2} \text{ s.t. } \mathsf{Ver}^{\mathsf{in}}(\mathsf{pk}, \mathcal{H}_1(\mathsf{pk}, m)_i, \sigma_i)_i = 1 \wedge m \notin Q_{\mathsf{Sign}}] \geq \varepsilon_{\mathcal{A}},$$

where $\varepsilon_{\mathcal{A}} = \Pr[\mathsf{Ver}(\mathsf{pk}, m^*, \sigma^*) = 1 \wedge m^* \notin Q_{\mathsf{Sign}}]$ is the success probability of $\mathcal{A}$. Note that we have replaced m^* with a general m, which may only weaken the statement.

We now consider the first query $(\mathsf{pk}, i, \hat{m}, \hat{\sigma}_i)$ to $\mathcal{H}_2$, such that $\mathsf{Ver}^{\mathsf{in}}(\mathsf{pk}, \mathcal{H}_1(\mathsf{pk}, \hat{m})_i, \hat{\sigma}_i) = 1$ and $\hat{m} \notin Q_{\mathsf{Sign}}$ at the time of the query to $\mathcal{H}_2$. If such a first query does not exist then we say $(\mathsf{pk}, i, \hat{m}, \hat{\sigma}_i) = \perp$, from the above derivation we know that $\mathcal{A}$ being successful implies the existence of such an $\hat{m}$, i.e., we may write

$$\Pr[(\mathsf{pk}, i, \hat{m}, \hat{\sigma}_i) \neq \perp] \geq \varepsilon_{\mathcal{A}}.$$

We will now consider the probability that the query $(\mathsf{pk}, i, \hat{m}, \hat{\sigma}_i)$ causes the event $\mathbf{W}$, this probability is a lower bound on $\Pr[\mathbf{W}]$. Recall that the event $\mathbf{W}$ occurs if the adversary forges a sub-signature which is not needed for the corresponding outer signature, we can write the event that $\hat{\sigma}_i$ is such an unnecessary sub-signature as $\hat{\sigma}_i \notin \sigma$ for $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, \hat{m})$, this leads to the following probability derivation:

$$\begin{aligned}
\Pr[\mathbf{W}] &\geq \Pr[(\mathsf{pk}, i, \hat{m}, \hat{\sigma}_i) \neq \perp \wedge \hat{\sigma}_i \notin \sigma, \sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, \hat{m})] \\
&= \Pr[(\mathsf{pk}, i, \hat{m}, \hat{\sigma}_i) \neq \perp \wedge \hat{b}_{i+1} = 0 \wedge \hat{b}_{i+2} = 1] \\
&= \Pr[(\mathsf{pk}, i, \hat{m}, \hat{\sigma}_i) \neq \perp] \Pr[\hat{b}_{i+1} = 0 \wedge \hat{b}_{i+2} = 1] \\
&\geq \varepsilon_{\mathcal{A}}/4.
\end{aligned}$$

Where $\hat{b}_i$ corresponds to the bits in the image of $\mathcal{H}_2$ which are used to compute the signature of $\hat{m}$. As $(\mathsf{pk}, i, \hat{m}, \hat{\sigma}_i)$ is the first query to $\mathcal{H}_2$ for an inner signature on a message m which has not been the input of a signature query before, it follows that the choice of $(\mathsf{pk}, i, \hat{m}, \hat{\sigma}_i)$ is independent of the choice of $\hat{b}_i$'s. This independence was used in the derivation above. We find that the probability that $\mathcal{R}^{\mathcal{A}}$ beats the RUF-RMA security game is at least $\varepsilon_{\mathcal{A}}/4$, which proves our Lemma. □

Remark 3.3 *One may wonder if we can do better, using fewer random oracle queries and inner signatures. In fact, the current construction is optimal in that at least three random-oracle queries and three inner signatures are necessary for this approach to work. To see why, it is helpful to view the construction*

as a majority-vote mechanism, where the bits b_i 'vote' which inner signature is excluded from the outer signature. A crucial property is that, in some cases, the outcome of the majority vote can be determined by observing only two out of three votes. The observed votes correspond to the inner signatures that the reduction is able to simulate, while the remaining unobserved vote allows the reduction to embed its challenge. Clearly, this idea only works with at least three votes, i.e., at least three inner signatures and RO queries are required in our construction.

The above result only considers single-user security, however it is possible to extend the result to multi-user EUF-CMA security without corruptions (MU-EUF-CMA). In this security model, the adversary will receive multiple public keys and can ask signature queries with respect to each of them. Finally the adversary wins if it forges a new signature w.r.t. one of the public keys.

The above reduction can be extended to base MU-EUF-CMA security of the outer scheme on the MU-RUF-RMA security of the inner scheme, this proof follows the same structure and can be found in Appendix B.

Lemma 3.4 *Let* $(\mathsf{Setup}^{\mathsf{in}}, \mathsf{Gen}^{\mathsf{in}}, \mathsf{Sign}^{\mathsf{in}}, \mathsf{Ver}^{\mathsf{in}})$ *be a unique signature scheme and let* $(\mathsf{Setup}, \mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ *be the outer signature scheme constructed using* S^{in} *as an inner signature scheme, following the HaSS (Section 3)transformation. If the inner signature scheme* S^{in} *is MU-RUF-RMA secure, then* S *is MU-EUF-CMA secure, where for both schemes we need to perform approximately the same amount of work to beat their respective security games. Specifically, for any PPT MU-EUF-CMA adversary* $\mathcal{A}$ *against* S *there exists a PPT MU-RUF-RMA adversary* $\mathcal{R}^{\mathcal{A}}$ *such that*

$$\Pr[\mathsf{Exp}^{\textit{MU-EUF-CMA}}_{\mathcal{A}}(1^\lambda, q, N) = 1] \leq 4 \cdot \Pr[\mathsf{Exp}^{\textit{MU-RUF-RMA}}_{\mathcal{R}^{\mathcal{A}}}(1^\lambda, 3q_{\mathsf{RO}}, N) = 1]$$

where the randomness is taken over the random choice of RO *and the random coins of* $\mathcal{A}$, $\mathcal{R}^{\mathcal{A}}$ *and the security games. The variables* q *and* q_{RO} *denote the amount of signature queries and random oracle queries made by* $\mathcal{A}$ *respectively. Additionally, the algorithm* $\mathcal{R}^{\mathcal{A}}$ *is PPT and only runs* $\mathcal{A}$ *once.*

Remark 3.5 *Our construction is tight and efficient, requiring only* 6 *random oracle queries to achieve a constant loss of* 4*. Literature also distinguishes* fully tight *schemes which have a loss of* 1 *to a non-interactive complexity assumption. In this remark we will briefly explain an extension of our scheme which allows the scheme to get arbitrarily close to fully tight at the cost of much inefficiency in the scheme itself. This extension is therefore not practically relevant but may be of theoretical interest.*

The extension is to use the outer scheme from the HaSS transformation 3 as the inner scheme in another construction of this definition. This creates a sort of layered signature scheme, each time we layer our scheme we triple the amount of innermost (at the lowest level) signatures we need per outer signature in the worst-case. We can view this as a tree graph, where each node has three children nodes and all the leaves are distinct signatures in the original scheme. Each node decides whether only two of its three child nodes are needed for the outer signature, using the rules in the HaSS transformation 3. Each layer ℓ therefore increases the chance that one of the innermost signatures/leaves is deemed unnecessary. For any individual innermost signature this probability is $1 - (3/4)^\ell$. As this probability nears 1*, the success probability of the reduction will also near that of the used adversary as it can hide challenges in a larger fraction of innermost signatures. By adding more layers we can thus approach, (but never reach) a fully tight reduction.*

4 A Tight Unique FTS Scheme in the Multi-User Model

Before we delve into our scheme let us first briefly recall what results we already have from Section 3 and our instantiation which we will present in Section 5. The instantiation gives us a unique signature scheme which is tightly RUF-RMA and MU-RUF-RMA secure based on the CDH assumption on bilinear groups. Using the transformation from Section 3 this gives us a tight EUF-CMA secure unique signature scheme, a simple extension given in the appendix also proves tight MU-EUF-CMA security (without corruptions). There also exists the following well-known transformation:

Lemma 4.1 ([BHJ+15;MS04]) *Any signature scheme* $(\mathsf{Setup}, \mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ *which is EUF-CMA secure is also MU-EUF-CMA-C secure. Specifically, if there exists an adversary* $\mathcal{A}$ *against the MU-EUF-CMA-C security of the scheme then there also exists an adversary* $\mathcal{R}^{\mathcal{A}}$ *against the EUF-CMA security such that the following holds*

$$\mathsf{Adv}^{\textit{MU-EUF-CMA-C}}_{\mathcal{A}}(\lambda, q, N) \leq N \cdot \mathsf{Adv}^{\textit{EUF-CMA}}_{\mathcal{R}^{\mathcal{A}}}(\lambda, q)$$

We therefore also have a scheme which is MU-EUF-CMA-C secure with loss independent of the number of signature queries but linear loss in the number of users. This raises the question whether the security loss N can be avoided. In this section, we indeed present an alternative MU-EUF-CMA-C secure solution admitting a tight reduction, i.e., the loss is independent of the number of users N. As with the previous sections, our result is presented as a general transformation, specifically we can transform any unique MU-EUF-CMA scheme into a tightly secure MU-EUF-CMA-C scheme, while preserving uniqueness.

Specifically, our transformation will allow us to construct a few-time signature scheme which is tightly secure in the multi-user model. A few-time signature scheme can be considered a signature scheme whose parameters are allowed to polynomially depend on the number of signatures q which will be computed per key pair. If we instantiate our scheme using CDH on bilinear groups as in Section 5, for $q \geq \lambda$, then the resulting construction has secret keys consisting of $O(q^2 \log q)$ group elements, signatures of $O(q \log q)$ elements and public keys consisting of a single group element. This construction then achieves a constant security loss of 4.

This construction gives us the first unique signature scheme which is tightly secure in the multi-user model. However, as the parameters of our scheme depend on q, the practical applications of this scheme are therefore more niche. Whether a tight unique MU-EUF-CMA-C construction exists whose parameters do not depend on q remains an open question.

Intuition. Recall that the difference between the two multi-user security notions is that an MU-EUF-CMA-C adversary has access to a corruption oracle, this oracle allows the adversary to "corrupt" users which reveals their secret keys. A reduction from MU-EUF-CMA-C security to MU-EUF-CMA security must therefore be able to answer these corruption queries, while it has only access to signature queries itself. The essence of our construction is therefore that we make revealing secret keys equivalent to revealing signatures. We take inspiration from hash-based FTS schemes, with the closest comparisons being HORS [RR02] and DFORS [YAG20]. In such constructions a signature consists of revealing several hash pre-images. For our construction, the outer signature scheme will create a secret key consisting of a set of inner signatures, after constructing these inner signatures it will delete the inner secret key. Signatures consist of partially revealing this outer secret key, which consists of inner signatures.

The CHORSS transformation. We will now describe our *Chained Hash to Obtain Random Signature Subset* (CHORSS) transformation. Let $(\mathsf{Setup}^{\mathsf{in}}, \mathsf{Gen}^{\mathsf{in}}, \mathsf{Sign}^{\mathsf{in}}, \mathsf{Ver}^{\mathsf{in}})$ be a unique signature scheme, which we call the *inner* signature scheme, and let $q \geq \lambda \geq 6$ be an integer. Let $k = 5q \log q$ (which we assume is integer for simplicity) and $L = 2q \cdot k$. Our *outer* signature scheme $(\mathsf{Setup}, \mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ uses some random oracle $\mathsf{RO} : \{0,1\}^* \to [L]$. The algorithms of the CHORSS scheme are defined as follows:

- The setup algorithm Setup is identical to the inner setup algorithm $\mathsf{Setup}^{\mathsf{in}}$.
- The key generation algorithm Gen takes as input public parameters $\mathcal{PP}$ and runs the inner key generation algorithm $(\mathsf{pk}', \mathsf{sk}') \leftarrow \mathsf{Gen}^{\mathsf{in}}(\lambda, \mathcal{PP})$. While we define the outer public pk key to be the same as the inner public key pk', the outer secret key will instead consist of an array of signatures of the numbers 1 to L using sk'. Specifically, the algorithm outputs $(\mathsf{pk}, \mathsf{sk}) := (\mathsf{pk}', (\mathsf{Sign}^{\mathsf{in}}(\mathsf{sk}', 1), \ldots, \mathsf{Sign}^{\mathsf{in}}(\mathsf{sk}', L)))$.
- The signing algorithm Sign will initialize $p_0 = ()$ (where p stands for partial signature) and perform the following steps for $i = 1...k$: The algorithm queries $j \leftarrow \mathsf{RO}(\mathsf{pk}, m, p_{i-1})$ and then computes $p_i = p_{i-1}||\sigma_j$, where σ_j is the inner signature of the number j, this signature is not computed on demand but instead retrieved from the signature array stored in the outer secret key sk. After this loop, the resulting signature is a tuple of k inner signatures $\sigma := p_k$.
- For the verification algorithm $\mathsf{Ver}(\mathsf{pk}, m, \sigma)$ we unpack σ to find each of the partial signatures p_i and then for each i we confirm that the last entry of p_i is the signature of $\mathsf{RO}(\mathsf{pk}, m, p_{i-1})$, we can easily verify this using $\mathsf{pk} = \mathsf{pk}'$ and the inner verification algorithm.

Uniqueness of the above scheme directly follows from the uniqueness of the underlying scheme. Next we will prove its security in the multi-user security with corruptions. We do so over a constant number of hybrid games. The core of the argument is that it is hard to find a message for which we require no new inner signatures. Specifically, because the adversary only has access to q signature queries, at most half of the inner signatures of the secret key will be revealed. For any new message m, the naive probability that all needed inner signatures are contained in the revealed set is then $(\frac{1}{2})^k$ and is negligible.

Lemma 4.2 *Consider the unique signature scheme* S *resulting from CHORSS transformation for some fixed* $q \geq \lambda \geq 6$*, and the respective inner scheme* S^{in}*. If* S^{in} *is MU-EUF-CMA secure, then* S *is MU-EUF-CMA-C secure. Specifically, if there exists an adversary* $\mathcal{A}$ *against* S *then we can construct an adversary* $\mathcal{R}^{\mathcal{A}}$ *against* S^{in} *such that*

$$\mathsf{Adv}^{\textit{MU-EUF-CMA-C}}_{\mathcal{A}}(\lambda, q, N) \leq \mathsf{Adv}^{\textit{MU-EUF-CMA}}_{\mathcal{R}^{\mathcal{A}}}(\lambda, 10q^2 \log q, N) + q_{\mathsf{RO}} \cdot e \cdot \left(\frac{1}{q}\right)^q. \tag{1}$$

Where $\mathcal{R}^{\mathcal{A}}$ *is efficient and only queries* $\mathcal{A}$ *once.*

Proof. We will prove our Lemma using a reduction. We will first give the reduction itself, which uses its MU-EUF-CMA oracles to perfectly simulate the MU-EUF-CMA-C game of the outer scheme S. The adversary is given access to both a random oracle and an outer signature oracle, each of which must be simulated by the reduction. To perform this simulation, the reduction itself is equipped with access to an inner signature oracle. The core idea of the construction is that, in order to succeed, the adversary is forced to query the random oracle on an inner signature forgery. Such a query provides the reduction with an inner signature forgery, thereby allowing it to win the MU-EUF-CMA security game. More precisely, we will show in the analysis below that the probability that the adversary wins the MU-EUF-CMA-C security game *without* providing an inner signature forgery to the reduction is negligible. We prove this by arguing that the adversary can only win using different methods with negligible probability. From this it follows that our reduction has probability $\varepsilon_{\mathcal{A}} - \mathsf{negl}(\lambda)$ to beat the MU-EUF-CMA game. Where $\varepsilon_{\mathcal{A}}$ is the success probability of the adversary $\mathcal{A}$ in the MU-EUF-CMA-C game.

Reduction. The reduction $\mathcal{R}$ will simply send the public keys from the MU-EUF-CMA game to $\mathcal{A}$. The random oracle it uses is simulated using lazy sampling and the reduction monitors this oracle for when $\mathcal{A}$ queries it using an inner signature forgery. Whenever $\mathcal{A}$ performs a signature query, the reduction will perform the k inner signature queries, for the inner signatures necessary to construct the outer signature. The reduction will store these results in a table such that it does not have to repeat signature queries. If the adversary ever forges an outer signature using a new (forged) inner signature, or if the adversary ever queries the random oracle using a newly forged inner signature, then the reduction will use this inner signature to beat the MU-EUF-CMA security game.

The reduction can additionally simulate corruption queries by simply querying its own signing oracle for the inner signatures of the elements of $[L]$. The secret key is namely a list of these inner signatures.

Note that the reduction perfectly simulates the MU-EUF-CMA-C game as it always correctly produces the outer signatures and secret keys, therefore the success probability of $\mathcal{A}$ remains unchanged. However, since we need to perform at most L inner signature queries per public key (k per outer signature query and potentially L per corruption query), we need to increase the maximum number of queries in the MU-EUF-CMA game to L accordingly, as seen in the Lemma statement.

As our reduction is already a perfect simulator, what remains to be proven is that the reduction is able to use the information gathered from the adversary to beat the MU-EUF-CMA security game.

Analysis and game G_1. We will consider the probability that the adversary $\mathcal{A}$ beats the MU-EUF-CMA-C security game without ever producing a fresh inner forgery. Accordingly, we define G_1 as a variant of the multi-user security game which will immediately abort if $\mathcal{A}$ ever uses a fresh inner forgery in a random oracle query or as part of the outer forgery. We are interested in the probability that $\mathcal{A}$ beats this security game, we denote this with $\Pr[G_1(\mathcal{A})]$. Which corresponds to the probability that $\mathcal{A}$ beats the original game **without** providing a fresh inner forgery.

Recall that whenever $\mathcal{A}$ queries a signature it will receive an ordered list of inner signatures on a subset of the numbers $\{1, \ldots, L\} = [L]$. Every outer-signature is a list of k inner signatures on elements in $[L]$. We refer to the list of elements of $[L]$ whose inner signatures compose an outer-signature as a sub-message list. Note that each entry $i+1$ of the sub-message list is found by hashing all previous i inner signatures. Therefore, as the adversary per definition of G_1 cannot query the random oracle for inner signatures it has not received before, we can find which part of the sub-message list the adversary can and cannot know. Specifically the adversary can only find a prefix of the sub-message list, up to and including the first sub-message for which it does not know the corresponding inner signature.

We name the suffix of the sub-message list which the adversary cannot compute as the 'hidden' part of the sub-message list. Whenever a signature query is performed by the adversary, the reduction will randomly sample what elements are included in the hidden suffix of the sub-message list. Of course, the boundary between the hidden and revealed section of the sub-message list will change over the duration of the security game, specifically in response to signature queries.

An equivalent game G_2. In Game G_2 we change how the game samples the hidden part of the sub-message list during signature queries.

In game G_2 we instead sample q chains per public key of length $k-1$ upon initialization. Upon the i-th signature query, game G_2 will not randomly sample the unknown part of the chain, but will instead define the sequence of RO outputs as a prefix of the i-th pre-sampled chain of the corresponding public key.

Due to a signature query, some part of the hidden sub-message list may be revealed for other messages, besides the subject of the signature query. We do not modify how these entries are sampled.

This definition is equivalent to G_1 as we only change the moment when we sample the hidden lists, but we do not introduce any new dependencies on this information. Therefore, we find that $\Pr[G_2(\mathcal{A})] = \Pr[G_1(\mathcal{A})]$. The main result of our change is that per public key we have fixed a set of $q \cdot (k-1)$ inner signatures which will (partially) be revealed to $\mathcal{A}$, while the adversary can only choose an additional q inner signatures themselves. We briefly re-iterate why this adaptive reprogramming of the random oracle does not cause inconsistencies. The main reason is that we only reprogram oracle queries which require fresh inner signatures to perform. Since per definition of G_1 (and also G_2) we know that the adversary never queries the oracle using fresh signatures, we know that the adversary does not notice this reprogramming.

Proving negligible success probability of G_2. Since per definition of G_2 the adversary can only win if it provides a fresh forgery without using fresh inner forgeries, the adversary is only able to beat G_2 if it finds a message m for which no new inner signatures are needed, i.e., the set of inner signatures needed for m is contained in the set of inner signatures the adversary has received.

As each query reveals k inner signatures, at most half of all inner signatures $L = 2q \cdot k$ contained in the secret key will be revealed throughout the security game.

Therefore, intuitively, for a random message m, each of the $k \geq q$ inner signatures needed has a probability of at most one half to be contained in this set. The probability that all needed inner signatures are contained in the revealed set is then $(\frac{1}{2})^k$ and is negligible.

However, the above intuition assumes that all revealed inner signatures are chosen at random. Due to our argument about the fixed hidden suffix of the sub-message list, we know that this is the case for most inner signatures which are revealed. In fact, per signature query there exists exactly one inner signature which the adversary can choose itself to be revealed. Furthermore, the complete set of q inner signatures which the adversary chooses to be revealed may depend on the set of inner signatures needed for any particular message.

Therefore, to upper-bound the adversary's success probability, we consider a slightly stronger adversary whose behavior is easier to analyze.

Recall that we are interested in the probability that the adversary wins the MU-EUF-CMA-C security game *without* producing an forgery. More precisely, we aim to show that this probability is negligible. The stronger adversary is given, in advance, $L/2$ inner signatures sampled uniformly at random and may additionally request q inner signatures of its choice. This reflects the fact that, for each outer-signature query, the adversary can influence at most one revealed inner signature, whereas the remaining $k-1$ inner signatures are sampled uniformly at random.

By issuing random-oracle queries, the adversary may construct sub-message sequences whose associated inner signatures together form a valid outer signature. As a consequence, for the adversary to succeed in forging an outer signature without producing an inner-signature forgery it must at least find a message whose sub-message sequence contains at most q distinct messages for which it does not already know the corresponding inner signatures.

Let us condition on the event that the adversary queries inner signatures for some fixed set S of messages with $|S| = q$. Later, we will apply a union bound over all possible choices of such sets, of which there are

$$\binom{L/2}{q} = \binom{qk}{q}.$$

Under this condition, the adversary succeeds if it finds a sequence of k messages such that every message either belongs to S or is among the $L/2$ messages for which the adversary initially received an inner signature. Hence, conditioned on a fixed choice of S, the success probability is

$$\left(\frac{L/2+q}{L}\right)^k = \left(\frac{1}{2} + \frac{q}{2qk}\right)^k = \left(\frac{1}{2}\right)^k \left(1 + \frac{1}{k}\right)^k \leq e \cdot \left(\frac{1}{2}\right)^k.$$

Taking a union bound over all possible choices of S bounds the adversary's success probability by

$$\begin{aligned}\binom{qk}{q} \cdot e \cdot \left(\frac{1}{2}\right)^k &\leq (k \cdot e)^q \cdot e \cdot \left(\frac{1}{2}\right)^k \\ &= e \cdot \left(\frac{1}{2}\right)^{k - q\log(k \cdot e)} \\ &= e \cdot \left(\frac{1}{2}\right)^{k - q\log k - q\log e} \\ &\leq e \cdot \left(\frac{1}{2}\right)^{k - q\log k - 2q}.\end{aligned} \tag{2}$$

We will now show that by choosing $k := 5q\log q$, this probability is negligible in q and thus in λ due to our restriction that $q \geq \lambda$. To this end, we observe that

$$\begin{aligned}k - q\log k - 2q &= 5q\log q - q\log(5q\log q) - 2q = 4q\log q - q\log 5 - q\log\log q - 2q \\ &\geq 3q\log q - q\log 5 - 2q \geq q(3\log q - 5) \geq q\log q\end{aligned}$$

for all $q \geq 6$. Hence, we can upper-bound the probability of Equation 2 as follows:

$$\binom{qk}{q} \cdot e \cdot \left(\frac{1}{2}\right)^k \leq e \cdot \left(\frac{1}{2}\right)^{k - q\log k - 2q} \leq e \cdot \left(\frac{1}{2}\right)^{q\log q} = e \cdot \left(\frac{1}{q}\right)^q, \tag{3}$$

which is negligible in q and thus in λ.

To complete the proof, we observe that the set of messages for which the adversary has the above probability to forge a signature is bounded by q_{RO}, as the adversary needs at least one random oracle query to test whether it can forge said message. Therefore, it follows that the probability that $\mathcal{A}$ finds a message it can forge over a polynomial number of random oracle queries is also negligible. This gives us the result that $\Pr[G_2] \leq q_{\mathsf{RO}}/q^q \leq \mathsf{negl}(\lambda)$, when we now bring all of these steps together we find

$$\begin{aligned}\mathsf{Adv}^{\text{MU-EUF-CMA-C}}_{\mathcal{A}}(\lambda, N, q) &= \Pr[G_1(\mathcal{A})] + \mathsf{Adv}^{\text{MU-EUF-CMA}}_{\mathcal{B}}(\lambda, N, 10q^2\log q) \\ &= \Pr[G_2(\mathcal{A})] + \mathsf{Adv}^{\text{MU-EUF-CMA}}_{\mathcal{B}}(\lambda, N, 10q^2\log q) \\ &\leq \mathsf{Adv}^{\text{MU-EUF-CMA}}_{\mathcal{B}}(\lambda, N, 10q^2\log q) + \frac{q_{\mathsf{RO}}}{q^q}\end{aligned}$$

as desired.

□

Remark 4.3 *For practical implementations, it is important to ensure that the inner secret key used during the key generation algorithm is deleted. If this secret key would be revealed during corruptions then the security of the above scheme would degrade, and we would have a loss linear in the number of users again.*

One could also consider a notion of security where the adversary is able to learn this secret key by corrupting parties during the key generation process. However, such an adversary would therefore need to choose which parties to corrupt before even knowing the corresponding public keys, this alternative notion would therefore not degrade the security of our scheme.

5 Instantiation of Our Schemes

In this section we give a tight unique signature scheme which instantiates RUF-RMA and MU-RUF-RMA security. The construction we present is simply BLS signatures [BLS01] without a hash function and is based on CDH on bilinear groups. We remark that RUF-RMA security is satisfied by RSA. Combined with our transformations in Section 3 and 4 we can construct a tight EUF-CMA scheme and a tight MU-EUF-CMA-C FTS respectively.

Definition 5.1 (Bilinear Group Generator) *Let* GGen^2 *be an algorithm that on input* 1^λ *returns a bilinear group description* $\mathcal{G} = (\mathbb{G}, \mathbb{G}_T, p, g, g_T, e)$*, where* $\mathbb{G}$ *and* $\mathbb{G}_T$ *are cyclic groups of order* p *for a*

2λ-bit prime p with respective group generators g and g_T and $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a non-degenerate mapping (that is $e(g, g) \neq 1$) satisfying

$$e(g^a, g^b) = e(g, g)^{ab}.$$

Then we call GGen^2 a bilinear group generator.

We in particular restrict ourselves to bilinear group generators which are PPT and where the group operation, inverse, equality test and the bilinear mapping itself are all efficiently computable. Finally, we also require that each element in the group $\mathbb{G}$ has a unique canonical representation. One example of such a group generator is given by Boneh and Franklin [BF01] whose group generator is in particular easily certifiable and therefore allows us to obtain public key verifiability Definition 2.2. On such a group we consider the following hardness assumption.

Definition 5.2 (CDH on Bilinear Groups) *Let GGen^2 be a PPT bilinear group generator, the CDH assumption for GGen^2, states that for any PPT algorithm $\mathcal{A}$ the probability*

$$\Pr[g^{ab} = z | z \leftarrow\!\!\$\ \mathcal{A}(\mathcal{G}, g^a, g^b), \mathcal{G} = (\mathbb{G}, \mathbb{G}_T, p, g, g_T, e) \leftarrow\!\!\$\ \mathsf{GGen}^2(1^\lambda)]$$

is negligible in λ. We call $(\mathcal{G}, g^a, g^b)$ a CDH problem instance.

This is a stronger assumption than CDH without a bilinear map, in particular we are able to verify any CDH solution, a property we explicitly use to verify signatures in the scheme. Such groups where DDH is easy but CDH is hard are also called 'Gap Diffie-Hellman' groups [BLS01].

Definition 5.3 (Bilinear Signature Scheme) *Given a bilinear group generator $(\mathbb{G}, \mathbb{G}_T, p, g, g_T, e)$ we define a signature scheme $(\mathsf{Setup}, \mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ consisting of the following algorithms:*

- *A Setup algorithm Setup which on input 1^λ computes $\mathcal{G} = (\mathbb{G}, \mathbb{G}_T, p, g, g_T, e) \leftarrow\!\!\$\ \mathsf{GGen}^2(1^\lambda)$ as public parameters.*
- *A key generation algorithm Gen which on input $(\mathcal{PP}, 1^\lambda)$ unpacks $\mathcal{PP} = (\mathbb{G}, \mathbb{G}_T, p, g, g_T, e)$ and samples a secret key $\mathsf{sk} \leftarrow\!\!\$\ [p-1]$. The public key is computed as $\mathsf{pk} = g^{\mathsf{sk}}$.*
- *A signing algorithm signs elements $x \in \mathbb{G}$ by computing $\sigma = x^{\mathsf{sk}}$ as the signature of a message x.*
- *The verification algorithm checks whether $e(\sigma, g) = e(x, \mathsf{pk})$.*

Correctness is easy to verify per definition of the bilinear mapping. In the following Lemma we will explicitly prove that this scheme has unique signatures. This property also follows from the fact that $z \mapsto e(z, g)$ is a bijection.

Lemma 5.4 (Bilinear Uniqueness) *The signature scheme in Definition 5.3 has unique signatures, in particular if e is our bilinear mapping then for any a, b we have that*

$$e(g^a, g^b) = e(z, g) \iff z = g^{ab}$$

Proof. Let $\mathcal{G} := (\mathbb{G}, \mathbb{G}_T, p, g, g_T, e)$ our bilinear group instance. Using the fact that e is non-degenerate and that $\mathbb{G}$ and $\mathbb{G}_T$ both have order p we find that

$$\begin{aligned} e(g^a, g^b) = e(g^c, g) &\iff e(g, g)^{ab} = e(g, g)^c \\ &\iff ab = c \mod p \\ &\iff g^{ab} = g^c. \end{aligned}$$

This in particular shows that there is a unique element z such that $e(z, g) = e(g^a, g^b)$. Because each element has a unique canonical representation we find that the signatures are indeed unique. □

Lemma 5.5 (Weak Unforgeability) *If the CDH assumption holds for the bilinear group generator GGen^2 then the corresponding signature scheme in Definition 5.3 is RUF-RMA secure. Specifically, if there exists an adversary $\mathcal{A}$ against the RUF-RMA security game of the bilinear signature scheme then there exists a bilinear CDH adversary $\mathcal{R}^{\mathcal{A}}$ such that*

$$\mathsf{Adv}^{RUF\text{-}RMA}_{\mathcal{A}}(1^\lambda, q) = \mathsf{Adv}^{CDH}_{\mathcal{R}^{\mathcal{A}}}(1^\lambda).$$

Furthermore, the time it takes to run $\mathcal{R}^{\mathcal{A}}$ is approximately equal to the runtime of $\mathcal{A}$.

Proof. We prove this Lemma by providing a reduction from the weak-unforgeability problem to the bilinear assumption. Our reduction is instantiated on the input $(\mathcal{G}, g^a, g^b)$ and sends $(\mathcal{G}, g^a)$ to the adversary $\mathcal{A}$. It will now simulate the CO and CH oracles as follows. Upon a query to the CO oracle it samples a random $r \leftarrow\!\!\$\ [p-1]$, computes $x := g^r$ and outputs $(x, (g^a)^r)$. Upon a query to the CH oracle it samples $r \in [p-1]$ and outputs $x := g^b \cdot g^r$. If the adversary terminates with output (x, σ) for an $x \in Q_{\mathsf{CH}}$ then we test $e(\sigma, g) = e(x, g^a)$, if this holds then per Lemma 5.4 we know we have $g^{ab} \cdot g^{ar} = \sigma = m^a$, and we can therefore compute $g^{ab} = (\sigma/(g^a)^r)$, which is the answer to the CDH challenge.

So clearly, if $\mathcal{A}$ beats the security experiment we are able to solve the CDH challenge. We will briefly argue that the reduction accurately simulates the RUF-RMA security experiment. As the public and secret key are generated in the same manner, we only need to show that the distribution of m is still uniformly random when we query either of the CO and CH oracles. This follows directly from the fact that we sample the variable r uniformly at random, and because $r \mapsto g^r$ and $r \mapsto g^r \cdot y$ are both bijections for image $\mathbb{G}^\times$. □

For our tight multi-user signature scheme we need an instantiation which tightly achieves the following multi-user variant of weak unforgeability:

For tight multi-user unforgeability we need the additional property that we can re-randomize public keys. We note that this property is not easily satisfied by RSA, which is why we present a scheme based on bilinear groups.

Lemma 5.6 *If the CDH assumption holds for the bilinear group generator* GGen^2 *then the corresponding signature scheme in Definition 5.3 is MU-RUF-RMA secure. Specifically, if there exists an adversary* $\mathcal{A}$ *against the MU-RUF-RMA security game of the bilinear signature scheme then there exists a bilinear CDH adversary* $\mathcal{R}^{\mathcal{A}}$ *such that*

$$\mathsf{Adv}^{MU\text{-}RUF\text{-}RMA}_{\mathcal{A}}(\lambda, q, N) \leq \mathsf{Adv}^{CDH}_{\mathcal{R}^{\mathcal{A}}}(\lambda).$$

Furthermore, the time it takes to run $\mathcal{R}^{\mathcal{A}}$ *is approximately equal to the runtime of* $\mathcal{A}$*.*

Proof. We prove this Lemma by providing a tight reduction from MU-RUF-RMA to RUF-RMA. Then using Lemma 5.5 we will have proven our result. Our reduction is instantiated on the input $(\mathcal{G}, g^a)$ and generates N public keys by generating $r_1, \ldots r_N \leftarrow\!\!\$\ [p-1]$ (where p is the prime order of our group), and chooses $\mathsf{pk}_i = g^{a+r_i}$. For every $CO(i)$ query we query our own CO oracle to receive (x, x^a), we then output (x, x^{a+r_i}). For every CH query we simply query our own CH oracle and return the answer. If the adversary terminates with output (x, i, σ) for an $x \in Q_{\mathsf{CH}}$ then the reduction outputs $(x, \sigma/(g^{r_i}))$ itself. If $\sigma = x^{a+r_i}$ then it follows that $\sigma/g^{r_i} = x^a$ and thus a valid forgery for the RUF-RMA security game. Finally, the adversary does not detect a difference as all signatures we return are correct and all public keys are sampled uniformly at random. □

Our instantiation gives an example of a scheme for which we can apply our transformations in Sections 3 and 4 to find a tight unique EUF-CMA secure signature scheme and a tight unique MU-EUF-CMA-C FTS scheme.

Acknowledgements

The first author was supported by TNO's Early Research Program - Next Generation Cryptography. L. Kohl was supported by a NWO (Dutch Research Council) grant with number VI.Veni.222.348 and by NWO Gravitation project QSC. This work was conducted while J. van der Laan was an intern in the Cryptology Group at CWI Amsterdam, as part of his Master's thesis at Utrecht University.

We have used an AI assistant (Claude, Anthropic) for proofreading. All suggested changes were checked and manually implemented by the authors. Additionally, Claude was used for basic editing of the introduction.

References

[AFL+12] Michel Abdalla et al. "Tightly-Secure Signatures from Lossy Identification Schemes". In: *EUROCRYPT 2012*. Ed. by David Pointcheval and Thomas Johansson. Vol. 7237. LNCS. Springer, Berlin, Heidelberg, Apr. 2012, pp. 572–590. DOI: 10.1007/978-3-642-29011-4_34.

[ASM07] Man Ho Au, Willy Susilo, and Yi Mu. "Practical Compact E-Cash". In: *ACISP 07*. Ed. by Josef Pieprzyk, Hossein Ghodosi, and Ed Dawson. Vol. 4586. LNCS. Springer, Berlin, Heidelberg, July 2007, pp. 431–445. DOI: 10.1007/978-3-540-73458-1_31.

[Bad14] Christoph Bader. "Efficient Signatures with Tight Real World Security in the Random-Oracle Model". In: *CANS 14*. Ed. by Dimitris Gritzalis, Aggelos Kiayias, and Ioannis G. Askoxylakis. Vol. 8813. LNCS. Springer, Cham, Oct. 2014, pp. 370–383. DOI: 10.1007/978-3-319-12280-9_24.

[Ber08] Daniel J. Bernstein. "Proving Tight Security for Rabin-Williams Signatures". In: *EUROCRYPT 2008*. Ed. by Nigel P. Smart. Vol. 4965. LNCS. Springer, Berlin, Heidelberg, Apr. 2008, pp. 70–87. DOI: 10.1007/978-3-540-78967-3_5.

[BF01] Dan Boneh and Matthew K. Franklin. "Identity-Based Encryption from the Weil Pairing". In: *Proceedings of the 21st Annual International Cryptology Conference on Advances in Cryptology*. CRYPTO '01. Berlin, Heidelberg: Springer-Verlag, 2001, pp. 213–229. ISBN: 3540424563.

[BHJ+15] Christoph Bader et al. "Tightly-Secure Authenticated Key Exchange". In: *TCC 2015, Part I*. Ed. by Yevgeniy Dodis and Jesper Buus Nielsen. Vol. 9014. LNCS. Springer, Berlin, Heidelberg, Mar. 2015, pp. 629–658. DOI: 10.1007/978-3-662-46494-6_26.

[BHK+19] Daniel J. Bernstein et al. "The SPHINCS$^+$ Signature Framework". In: *ACM CCS 2019*. Ed. by Lorenzo Cavallaro et al. ACM Press, Nov. 2019, pp. 2129–2146. DOI: 10.1145/3319535.3363229.

[BIN97] Mihir Bellare, Russell Impagliazzo, and Moni Naor. "Does Parallel Repetition Lower the Error in Computationally Sound Protocols?" In: *38th FOCS*. IEEE Computer Society Press, Oct. 1997, pp. 374–383. DOI: 10.1109/SFCS.1997.646126.

[BJL+16] Christoph Bader et al. "On the Impossibility of Tight Cryptographic Reductions". In: *EUROCRYPT 2016, Part II*. Ed. by Marc Fischlin and Jean-Sébastien Coron. Vol. 9666. LNCS. Springer, Berlin, Heidelberg, May 2016, pp. 273–304. DOI: 10.1007/978-3-662-49896-5_10.

[BLS01] Dan Boneh, Ben Lynn, and Hovav Shacham. "Short Signatures from the Weil Pairing". In: *ASIACRYPT 2001*. Ed. by Colin Boyd. Vol. 2248. LNCS. Springer, Berlin, Heidelberg, Dec. 2001, pp. 514–532. DOI: 10.1007/3-540-45682-1_30.

[BR93] Mihir Bellare and Phillip Rogaway. "Random Oracles are Practical: A Paradigm for Designing Efficient Protocols". In: *ACM CCS 93*. Ed. by Dorothy E. Denning et al. ACM Press, Nov. 1993, pp. 62–73. DOI: 10.1145/168588.168596.

[BR96] Mihir Bellare and Phillip Rogaway. "The Exact Security of Digital Signatures: How to Sign with RSA and Rabin". In: *EUROCRYPT'96*. Ed. by Ueli M. Maurer. Vol. 1070. LNCS. Springer, Berlin, Heidelberg, May 1996, pp. 399–416. DOI: 10.1007/3-540-68339-9_34.

[CCM+24] Miranda Christ et al. "Accountable Secret Leader Election". In: *6th Conference on Advances in Financial Technologies*. 2024.

[Cha81] David Chaum. "Verification by Anonymous Monitors". In: *CRYPTO'81*. Ed. by Allen Gersho. Vol. ECE Report 82-04. U.C. Santa Barbara, Dept. of Elec. and Computer Eng., 1981, pp. 138–139.

[CHY+05] Sherman SM Chow et al. "An e-lottery scheme using verifiable random function". In: *International conference on computational science and its applications*. Springer. 2005, pp. 651–660.

[CMS99] Christian Cachin, Silvio Micali, and Markus Stadler. "Computationally Private Information Retrieval with Polylogarithmic Communication". In: *EUROCRYPT'99*. Ed. by Jacques Stern. Vol. 1592. LNCS. Springer, Berlin, Heidelberg, May 1999, pp. 402–414. DOI: 10.1007/3-540-48910-X_28.

[Cor00] Jean-Sébastien Coron. "On the Exact Security of Full Domain Hash". In: *CRYPTO 2000*. Ed. by Mihir Bellare. Vol. 1880. LNCS. Springer, Berlin, Heidelberg, Aug. 2000, pp. 229–235. DOI: 10.1007/3-540-44598-6_14.

[Cor02] Jean-Sébastien Coron. "Optimal Security Proofs for PSS and Other Signature Schemes". In: *EUROCRYPT 2002*. Ed. by Lars R. Knudsen. Vol. 2332. LNCS. Springer, Berlin, Heidelberg, Apr. 2002, pp. 272–287. DOI: 10.1007/3-540-46035-7_18.

[DGJ+21] Denis Diemert et al. "More Efficient Digital Signatures with Tight Multi-user Security". In: *PKC 2021, Part II*. Ed. by Juan Garay. Vol. 12711. LNCS. Springer, Cham, May 2021, pp. 1–31. DOI: 10.1007/978-3-030-75248-4_1.

[DSS05] C. Dods, Nigel P. Smart, and Martijn Stam. "Hash Based Digital Signature Schemes". In: *10th IMA International Conference on Cryptography and Coding*. Ed. by Nigel P. Smart. Vol. 3796. LNCS. Springer, Berlin, Heidelberg, Dec. 2005, pp. 96–115. DOI: 10.1007/11586821_8.

[FHN+12] Sebastian Faust et al. "Signature Schemes Secure against Hard-to-Invert Leakage". In: *ASIACRYPT 2012*. Ed. by Xiaoyun Wang and Kazue Sako. Vol. 7658. LNCS. Springer, Berlin, Heidelberg, Dec. 2012, pp. 98–115. DOI: 10.1007/978-3-642-34961-4_8.

[GCS+17] Fuchun Guo et al. "Optimal Security Reductions for Unique Signatures: Bypassing Impossibilities with a Counterexample". In: *CRYPTO 2017, Part II*. Ed. by Jonathan Katz and Hovav Shacham. Vol. 10402. LNCS. Springer, Cham, Aug. 2017, pp. 517–547. DOI: 10.1007/978-3-319-63715-0_18.

[GJ18] Kristian Gjøsteen and Tibor Jager. "Practical and Tightly-Secure Digital Signatures and Authenticated Key Exchange". In: *CRYPTO 2018, Part II*. Ed. by Hovav Shacham and Alexandra Boldyreva. Vol. 10992. LNCS. Springer, Cham, Aug. 2018, pp. 95–125. DOI: 10.1007/978-3-319-96881-0_4.

[GNP+15] Sharon Goldberg et al. "NSEC5: Provably Preventing DNSSEC Zone Enumeration". In: *NDSS 2015*. The Internet Society, Feb. 2015. DOI: 10.14722/ndss.2015.23211.

[GO93] Shafi Goldwasser and Rafail Ostrovsky. "Invariant Signatures and Non-Interactive Zero-Knowledge Proofs are Equivalent (Extended Abstract)". In: *CRYPTO'92*. Ed. by Ernest F. Brickell. Vol. 740. LNCS. Springer, Berlin, Heidelberg, Aug. 1993, pp. 228–245. DOI: 10.1007/3-540-48071-4_16.

[GS22] Fuchun Guo and Willy Susilo. "Optimal Tightness for Chain-Based Unique Signatures". In: *EUROCRYPT 2022, Part II*. Ed. by Orr Dunkelman and Stefan Dziembowski. Vol. 13276. LNCS. Springer, Cham, May 2022, pp. 553–583. DOI: 10.1007/978-3-031-07085-3_19.

[HJ16] Dennis Hofheinz and Tibor Jager. "Verifiable Random Functions from Standard Assumptions". In: *TCC 2016-A, Part I*. Ed. by Eyal Kushilevitz and Tal Malkin. Vol. 9562. LNCS. Springer, Berlin, Heidelberg, Jan. 2016, pp. 336–362. DOI: 10.1007/978-3-662-49096-9_14.

[HJK+21] Shuai Han et al. "Authenticated Key Exchange and Signatures with Tight Security in the Standard Model". In: *CRYPTO 2021, Part IV*. Ed. by Tal Malkin and Chris Peikert. Vol. 12828. LNCS. Virtual Event: Springer, Cham, Aug. 2021, pp. 670–700. DOI: 10.1007/978-3-030-84259-8_23.

[HOS25] Keitaro Hashimoto, Wakaha Ogata, and Yusuke Sakai. *Signatures with Tight Adaptive Corruptions from Search Assumptions*. Cryptology ePrint Archive, Report 2025/115. 2025. URL: https://eprint.iacr.org/2025/115.

[KK12] Saqib A. Kakvi and Eike Kiltz. "Optimal Security Proofs for Full Domain Hash, Revisited". In: *EUROCRYPT 2012*. Ed. by David Pointcheval and Thomas Johansson. Vol. 7237. LNCS. Springer, Berlin, Heidelberg, Apr. 2012, pp. 537–553. DOI: 10.1007/978-3-642-29011-4_32.

[KKM12] Saqib A. Kakvi, Eike Kiltz, and Alexander May. "Certifying RSA". In: *ASIACRYPT 2012*. Ed. by Xiaoyun Wang and Kazue Sako. Vol. 7658. LNCS. Springer, Berlin, Heidelberg, Dec. 2012, pp. 404–414. DOI: 10.1007/978-3-642-34961-4_25.

[KW03] Jonathan Katz and Nan Wang. "Efficiency Improvements for Signature Schemes with Tight Security Reductions". In: *ACM CCS 2003*. Ed. by Sushil Jajodia, Vijayalakshmi Atluri, and Trent Jaeger. ACM Press, Oct. 2003, pp. 155–164. DOI: 10.1145/948109.948132.

[Lam79] Leslie Lamport. *Constructing Digital Signatures from a One-way Function*. Technical Report SRI-CSL-98. SRI International Computer Science Laboratory, Oct. 1979.

[Lub96] Michael Luby. *Pseudorandomness and cryptographic applications*. Vol. 1. Princeton University Press, 1996.

[Lys02] Anna Lysyanskaya. "Unique Signatures and Verifiable Random Functions from the DH-DDH Separation". In: *CRYPTO 2002*. Ed. by Moti Yung. Vol. 2442. LNCS. Springer, Berlin, Heidelberg, Aug. 2002, pp. 597–612. DOI: 10.1007/3-540-45708-9_38.

[MP18] Andrew Morgan and Rafael Pass. "On the Security Loss of Unique Signatures". In: *TCC 2018, Part I*. Ed. by Amos Beimel and Stefan Dziembowski. Vol. 11239. LNCS. Springer, Cham, Nov. 2018, pp. 507–536. DOI: 10.1007/978-3-030-03807-6_19.

[MRV99a] Silvio Micali, Michael Rabin, and Salil Vadhan. "Verifiable random functions". In: *Proceedings of the 40th Annual Symposium on Foundations of Computer Science (FOCS '99)*. IEEE. 1999, pp. 120–130.

[MRV99b] Silvio Micali, Michael O. Rabin, and Salil P. Vadhan. "Verifiable Random Functions". In: *40th FOCS*. IEEE Computer Society Press, Oct. 1999, pp. 120–130. DOI: 10.1109/SFFCS.1999.814584.

[MS04] Alfred Menezes and Nigel P. Smart. "Security of Signature Schemes in a Multi-User Setting". In: *DCC* 33.3 (2004), pp. 261–274. DOI: 10.1023/B:DESI.0000036250.18062.3f.

[Nie21] David Niehues. "Verifiable Random Functions with Optimal Tightness". In: *PKC 2021, Part II*. Ed. by Juan Garay. Vol. 12711. LNCS. Springer, Cham, May 2021, pp. 61–91. DOI: 10.1007/978-3-030-75248-4_3.

[Per01] Adrian Perrig. "The BiBa One-Time Signature and Broadcast Authentication Protocol". In: *ACM CCS 2001*. Ed. by Michael K. Reiter and Pierangela Samarati. ACM Press, Nov. 2001, pp. 28–37. DOI: 10.1145/501983.501988.

[Rab79] Michael O. Rabin. *Digital signatures and public key functions as intractable as factorization*. Technical Report MIT/LCS/TR-212. Massachusetts Institute of Technology, Jan. 1979, p. 16.

[RR02] Leonid Reyzin and Natan Reyzin. "Better than BiBa: Short One-Time Signatures with Fast Signing and Verifying". In: *ACISP 02*. Ed. by Lynn Margaret Batten and Jennifer Seberry. Vol. 2384. LNCS. Springer, Berlin, Heidelberg, July 2002, pp. 144–153. DOI: 10.1007/3-540-45450-0_11.

[RSA78] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems". In: *Communications of the Association for Computing Machinery* 21.2 (Feb. 1978), pp. 120–126. DOI: 10.1145/359340.359342.

[Sha18] Hovav Shacham. "Short Unique Signatures from RSA with a Tight Security Reduction (in the Random Oracle Model)". In: *FC 2018*. Ed. by Sarah Meiklejohn and Kazue Sako. Vol. 10957. LNCS. Springer, Berlin, Heidelberg, Feb. 2018, pp. 61–79. DOI: 10.1007/978-3-662-58387-6_4.

[WLG+19] Ge Wu et al. "Tightly Secure Public-Key Cryptographic Schemes from One-More Assumptions". In: *J. Comput. Sci. Technol.* 34.6 (2019), pp. 1366–1379.

[YAG20] Mahmoud Yehia, Riham AlTawy, and T. Aaron Gulliver. "Hash-Based Signatures Revisited: A Dynamic FORS with Adaptive Chosen Message Security". In: *AFRICACRYPT 20*. Ed. by Abderrahmane Nitaj and Amr M. Youssef. Vol. 12174. LNCS. Springer, Cham, July 2020, pp. 239–257. DOI: 10.1007/978-3-030-51938-4_12.

A RSA Instantiation

In this section we prove that textbook RSA satisfies tight RUF-RMA security. Combined with our transformation in Section 3 we can construct a tight EUF-CMA secure signature scheme. This section will be very similarly structured to our instantiation based on bilinear groups in Section 5, however a key difference is that we only obtain single-user RUF-RMA security in this section, as RSA does not have the key-rerandomization property we used to prove MU-RUF-RMA security.

Definition A.1 (RSA Generator) *Let $\mathcal{RSA}$ be an algorithm that on input 1^λ returns a tuple (N, e, d), where N is an integer $N = p \cdot q$ and e, d are integers written modulo $(q-1)(p-1)$ such that $e^{-1} = d \mod (p-1)(q-1)$. Then we call $\mathcal{RSA}$ an RSA generator.*

We in particular restrict ourselves to RSA generators which are PPT. One can even construct a certified RSA generator using the certification algorithm by Kakhi, Kiltz and May [KKM12]. This immediately gives public key verifiability (Definition 2.2) on the RSA signature scheme defined below.

Definition A.2 (RSA Assumption) *Let $\mathcal{RSA}$ be an RSA generator, the RSA assumption for $\mathcal{RSA}$, states that for any PPT algorithm $\mathcal{A}$ the probability*

$$\Pr[x^e = y | x \leftarrow\$ \mathcal{A}(N, e, y), (N, e, d) \leftarrow\$ \mathcal{RSA}(1^\lambda), y \leftarrow\$ \mathbb{Z}_N]$$

is negligible in λ. Furthermore, we call (N, e, y) an RSA problem instance.

Definition A.3 (RSA Signature Scheme[RSA78]) *Given an RSA generator $\mathcal{RSA}$ we define a signature scheme* $(\mathsf{Setup}, \mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ *consisting of the following algorithms:*

- *A setup algorithm which is simply the identity* $1^\lambda \leftarrow \mathsf{Setup}(1^\lambda)$.
- *A key generation algorithm* Gen *which on input* (1^λ) *computes* $\mathsf{sk} = (N, e, d) \leftarrow\$ \mathcal{RSA}(1^\lambda)$. *The resulting public key is* $\mathsf{pk} = (N, e)$.
- *A signing algorithm signs elements $m \in \mathbb{Z}_N^\times$ by computing $\sigma = m^d$ as the signature of a message m.*
- *The verification algorithm checks whether $\sigma^e = m$.*

Uniqueness follows from the fact that $x \mapsto x^e$ and $x \mapsto x^d$ are a permutation and its inverse respectively.

What remains is for us to prove that this scheme obtains RUF-RMA security

Lemma A.4 (Weak Unforgeability) *If the RSA assumption holds for the RSA generator $\mathcal{RSA}$ then the corresponding signature scheme in Definition A.3 is RUF-RMA secure. Specifically, if there exists an adversary $\mathcal{A}$ against the RUF-RMA security game of the bilinear signature scheme then there exists an adversary against the RSA assumption bilinear CDH adversary $\mathcal{R}^{\mathcal{A}}$ such that*

$$\mathsf{Adv}_{\mathcal{A}}^{RUF\text{-}RMA}(\lambda, q) \leq \mathsf{Adv}_{\mathcal{R}^{\mathcal{A}}}^{CDH}(\lambda).$$

Furthermore, the time it takes to run $\mathcal{R}^{\mathcal{A}}$ is approximately equal to the runtime of $\mathcal{A}$.

Proof. We prove this Lemma by providing a reduction from the weak-unforgeability problem to the bilinear assumption. Our reduction is instantiated on the input (N, e, y) and sends $\mathsf{pk} = (N, e)$ to the adversary $\mathcal{A}$. It will now simulate the CO and CH oracles as follows. Upon a query to the CO oracle it samples a random $r \leftarrow\$ \mathbb{Z}_N^\times$, and outputs $(m, \sigma) = (r^e, r)$. Upon a query to the CH oracle it samples $r \in \mathbb{Z}_N^\times$ and outputs $m := y \cdot r^e$. If the adversary terminates with output (m, σ) for an $m \in Q_{\mathsf{CH}}$ then we test $\sigma^e = m$, if this holds then we can compute $z = \sigma / r$ which satisfies $z^e = \sigma^e / r^e = y$, which is the answer to the RSA problem instance.

So clearly, if $\mathcal{A}$ beats the security experiment we are able to solve the RSA problem instance. We will briefly argue that the reduction accurately simulates the RUF-RMA security experiment. As the public and secret key are generated in the same manner, we only need to show that the distribution of m is still uniformly random when we query either of the CO and CH oracles. This follows directly from the fact that we sample the variable r uniformly at random, and because $r \mapsto g^{r \cdot e}$ and $r \mapsto g^{r \cdot e} \cdot y$ are both bijections for image $\mathbb{Z}_N^\times$. □

For our tight multi-user signature scheme we need an instantiation which tightly achieves the following multi-user variant of weak unforgeability:

For tight multi-user unforgeability we need the additional property that we can re-randomize public keys. We note that this property is not easily satisfied by RSA, which is why we present a scheme based on bilinear groups.

B MU-RUF-RMA to MU-EUF-CMA transformation

We will briefly prove that our tight single user signature scheme is also secure in the multi-user model without corruptions. We do so by proving that our general transformation in Section 3 transforms any MU-RUF-RMA scheme into a MU-EUF-CMA secure scheme. This proof follows the exact same structure as Lemma 3.2, we only additionally have to keep track of the users involved.

Lemma 3.4 *Let* $(\mathsf{Setup}^{\mathsf{in}}, \mathsf{Gen}^{\mathsf{in}}, \mathsf{Sign}^{\mathsf{in}}, \mathsf{Ver}^{\mathsf{in}})$ *be a unique signature scheme and let* $(\mathsf{Setup}, \mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ *be the outer signature scheme constructed using* S^{in} *as an inner signature scheme, following the HaSS (Section 3)transformation. If the inner signature scheme* S^{in} *is MU-RUF-RMA secure, then* S *is MU-EUF-CMA secure, where for both schemes we need to perform approximately the same amount of work*

to beat their respective security games. Specifically, for any PPT MU-EUF-CMA adversary $\mathcal{A}$ against S there exists a PPT MU-RUF-RMA adversary $\mathcal{R}^{\mathcal{A}}$ such that

$$\Pr[\mathsf{Exp}^{\textit{MU-EUF-CMA}}_{\mathcal{A}}(1^\lambda, q, N) = 1] \leq 4 \cdot \Pr[\mathsf{Exp}^{\textit{MU-RUF-RMA}}_{\mathcal{R}^{\mathcal{A}}}(1^\lambda, 3q_{\mathsf{RO}}, N) = 1]$$

where the randomness is taken over the random choice of RO and the random coins of $\mathcal{A}$, $\mathcal{R}^{\mathcal{A}}$ and the security games. The variables q and q_{RO} denote the amount of signature queries and random oracle queries made by $\mathcal{A}$ respectively. Additionally, the algorithm $\mathcal{R}^{\mathcal{A}}$ is PPT and only runs $\mathcal{A}$ once.

Proof. Our goal is to transform the MU-EUF-CMA adversary $\mathcal{A}$ into a MU-RUF-RMA adversary against the scheme S^{in}. For this we create a reduction $\mathcal{R}$ which is an adversary against the MU-RUF-RMA game and therefore has access to its oracles. This reduction then pretends to be the MU-EUF-CMA security game and runs $\mathcal{A}$ using oracles controlled by $\mathcal{R}$. We will first describe the reduction, then we argue that $\mathcal{A}$ cannot detect that it is not in the true security game, and finally we will argue about our success probability.

Reduction. First, assume that $\mathcal{A}$ will always query $\mathcal{H}_1(\mathsf{pk}_j, m)$ before $\mathcal{O}_{\mathsf{Sign}}(j, m)$, $H_2(\mathsf{pk}_j, i, m, \sigma_i)$ or before providing a forgery on m. We can make this assumption without loss of generality as we could simply make a wrapper around $\mathcal{A}$ to make these additional queries. Upon a query of $\mathcal{H}_1(\mathsf{pk}_j, m)$ the reduction first samples bits $b_i \in \{0, 1\}$ for $i \in \{1, 2, 3\}$ uniformly at random. These bits will become the results of the $\mathcal{H}_2(\mathsf{pk}_j, i, m, \sigma_i)$ queries regardless of the eventual values of the σ_i, this allows the reduction to already determine which inner signatures σ_i are needed for the outer signature on the message m. For each i, if σ_i would be needed for the final signature the reduction queries $(x, y) \leftarrow\!\$\ \mathcal{O}_{\mathsf{CO}}(j)$, this is used to define $h_i := x$, $\sigma_i := y$, and $\mathcal{H}_2(\mathsf{pk}_j, i, m, \sigma_i) := b_i$. If on the other hand, σ_i would not be needed in the outer signature, then the reduction will instead query its challenge oracle $x \leftarrow\!\$\ \mathcal{O}_{\mathsf{CH}}(j)$ and defines $h_i := x$. Finally, the reduction outputs the RO response as (h_1, h_2, h_3). If the adversary later queries $\mathcal{O}_{\mathsf{Sign}}(j, m)$, then we can answer this signature query correctly as we know exactly the σ_i needed.

We briefly argue that all random variables involved are distributed identically to how they would be in the MU-EUF-CMA security experiment. Clearly the key generation algorithm is identical and per definition of $\mathcal{O}_{\mathsf{CH}}$ and $\mathcal{O}_{\mathsf{CO}}$, which are used as the output of our hash function $\mathcal{H}_1$, are uniformly random and independent of all other random variables.

Whenever $\mathcal{A}$ makes a query of the form $\mathcal{H}_2(\mathsf{pk}_j, i, m, \sigma_i)$ the reduction finds $\mathcal{H}_1(\mathsf{pk}_j, m) = (h_1, h_2, h_3)$ and checks whether σ_i is a signature for h_i w.r.t pk_j. If this holds then there are two possible cases (1) h_i was output by $\mathcal{O}_{\mathsf{CH}}(j)$ or (2) by $\mathcal{O}_{\mathsf{CO}}(j)$. Which case we are in is determined by the value of each $\mathcal{H}_2(\mathsf{pk}_j, k, m, \sigma_k)$ for $k \in \{1, 2, 3\}$. In the former case (1) The reduction can use this information to beat the MU-RUF-RMA security game, and at the end of the computation it will output (h_i, σ_i). We call this event $\mathbf{W}$, and conclude that the event $\mathbf{W}$ causes $\mathcal{R}^{\mathcal{A}}$ to beat the RUF-RMA security game.

Analysis. Our goal is to argue that the probability of $\mathbf{W}$ is high. First we assume without loss of generality that the adversary has queried $\mathcal{H}_2(\mathsf{pk}_j, i, m, \sigma_i)$ for all σ_i in the eventual forgery. Let (m^*, σ^*, j^*) be the forgery produced by $\mathcal{A}$ and Q_{H2} be the set of random oracle queries to $\mathcal{H}_2$. We find that

$$\begin{aligned}
&\mathsf{Ver}(\mathsf{pk}_{j^*}, m^*, \sigma^*) = 1 \land m^* \notin Q_{\mathsf{CO}} \\
\implies &\exists \sigma_i \in \sigma^* \text{ s.t. } \mathsf{Ver}^{\mathsf{in}}(\mathsf{pk}_{j^*}, \mathcal{H}_1(\mathsf{pk}_{j^*}, m^*)_i, \sigma_i) = 1 \land m^* \notin Q_{\mathsf{CO}} \\
\implies &\exists (\mathsf{pk}_j, i, m, \sigma_i) \in Q_{H2} \text{ s.t. } \mathsf{Ver}^{\mathsf{in}}(\mathsf{pk}_j, \mathcal{H}_1(\mathsf{pk}_j, m)_i, \sigma_i) = 1 \land m \notin Q_{\mathsf{CO}}
\end{aligned}$$

and therefore

$$\Pr[\exists (\mathsf{pk}_j, i, m, \sigma_i) \in Q_{H2} \text{ s.t. } \mathsf{Ver}^{\mathsf{in}}(\mathsf{pk}_j, \mathcal{H}_1(\mathsf{pk}_j, m)_i, \sigma_i)_i = 1 \land m \notin Q_{\mathsf{CO}}] \geq \varepsilon_{\mathcal{A}},$$

where $\varepsilon_{\mathcal{A}}$ is the success probability of $\mathcal{A}$. We now consider the first query $(\mathsf{pk}_{\hat{j}}, i, \hat{m}, \hat{\sigma}_i)$ to $\mathcal{H}_2$, such that $\mathsf{Ver}^{\mathsf{in}}(\mathsf{pk}_{\hat{j}}, \mathcal{H}_1(\mathsf{pk}_{\hat{j}}, \hat{m})_i, \hat{\sigma}_i) = 1$ and $\hat{m} \notin Q_{\mathsf{CO}}$ at the time of the query to $\mathcal{H}_2$. If such a first query does not exist then we say $(\mathsf{pk}_{\hat{j}}, i, \hat{m}, \hat{\sigma}_i) = \bot$, from the above derivation we know that

$$\Pr[(\mathsf{pk}_{\hat{j}}, i, \hat{m}, \hat{\sigma}_i) \neq \bot] \geq \varepsilon_{\mathcal{A}}.$$

We will now consider the probability that the query $(\mathsf{pk}_{\hat{j}}, i, \hat{m}, \hat{\sigma}_i)$ causes the event $\mathbf{W}$, this probability is a lower bound on $\Pr[\mathbf{W}]$. Recall that the event $\mathbf{W}$ occurs if the inner signature $\hat{\sigma}_i$ on h_i is not needed

for the outer signature on $\hat{m}$, we write this probability as follows:

$$\begin{aligned}\Pr[\mathbf{W}] &\geq \Pr[(\mathsf{pk}_{\hat{j}}, i, \hat{m}, \hat{\sigma}_i) \neq \bot \wedge \hat{\sigma}_i \notin \mathsf{Sign}(\mathsf{sk}_{\hat{j}}, \hat{m})] \\ &= \Pr[(\mathsf{pk}_{\hat{j}}, i, \hat{m}, \hat{\sigma}_i) \neq \bot \wedge \hat{b_{i+1}} = 0 \wedge \hat{b_{i+2}} = 1] \\ &= \Pr[(\mathsf{pk}_{\hat{j}}, i, \hat{m}, \hat{\sigma}_i) \neq \bot] \Pr[\hat{b_{i+1}} = 0 \wedge \hat{b_{i+2}} = 1] \\ &\geq \varepsilon_{\mathcal{A}}/4.\end{aligned}$$

Where $\hat{b}_i$ corresponds to the bits in the image of $\mathcal{H}_2$ which are used to compute the signature of $\hat{m}$. As $(\mathsf{pk}_{\hat{j}}, i, \hat{m}, \hat{\sigma}_i)$ is the first query to $\mathcal{H}_2$ for an inner signature on a message m which has not been the input of a signature query before, it follows that the choice of $(\mathsf{pk}_{\hat{j}}, i, \hat{m}, \hat{\sigma}_i)$ is independent of the choice of b_i's. This independence was used in the derivation above. We find that the probability that $\mathcal{R}^{\mathcal{A}}$ beats the RUF-RMA security game is at least $\varepsilon_{\mathcal{A}}/4$, which proves our Lemma. □

C A Tight Unique Signature Scheme Secure Against Semi-Adaptive Corruptions

In this section we will describe and prove security of a signature scheme which is secure in the multi-user model with semi-adaptive corruptions. The security notion we target is MU-EUF-CMA-C security as defined in Definition C.1 with the additional restriction that per public key, the adversary can only corrupt the key before asking signature queries for said key. This weakens the security notion, but the advantage over our previous construction is that we again only need a constant number of operations and random oracle queries.

$\mathsf{Exp}^{\text{MU-EUF-CMA-SAC}}_{\mathcal{A}}(1^\lambda, q, N)$:

$Q_{\mathsf{CO}}, U_{\mathsf{CO}}, T \leftarrow \emptyset; c_1, \ldots, c_N \leftarrow 0$
$\mathcal{PP} \leftarrow\$ \mathsf{Setup}(1^\lambda)$
$(\mathsf{pk}_1, \mathsf{sk}_1), \ldots, (\mathsf{pk}_N, \mathsf{sk}_N) \leftarrow\$ \mathsf{Gen}(1^\lambda, \mathcal{PP})$
$(m, \sigma, i^*) \leftarrow\$ \mathcal{A}^{\mathcal{O}_{\mathsf{Sign}}(\cdot), \mathsf{RO}(\cdot), \mathcal{O}_{\mathsf{CO}}}(\mathcal{PP}, \mathsf{pk}_1, \ldots, \mathsf{pk}_N)$
if $\mathsf{Ver}^{\mathsf{RO}(\cdot)}(\mathsf{pk}_{i^*}, \sigma, m) = 1 \wedge i^* \notin U_{\mathsf{CO}} \wedge (i^*, m) \notin Q_{\mathsf{CO}}$
  $\wedge \forall i \in [n] : c_i \leq q$:
  return 1
else
  return 0

$\mathcal{O}_{\mathsf{CO}}(i)$:

if $\exists (i', m) \in Q_{\mathsf{CO}}$ s.t. $i' = i$
  return $\bot$
$U_{\mathsf{CO}} \leftarrow U_{\mathsf{CO}} \cup \{i\}$
return sk_i

$\mathcal{O}_{\mathsf{Sign}}(i, m)$:

$c_i \leftarrow c_i + 1$
$Q_{\mathsf{CO}} \leftarrow Q_{\mathsf{CO}} \cup \{(i, m)\}$
$\sigma \leftarrow \mathsf{Sign}^{\mathsf{RO}(\cdot)}(\mathsf{sk}_i, m)$
return σ

$\mathsf{RO}(x)$:

if $\exists y' : (x, y') \in T$:
  $y \leftarrow y'$
else :
  $y \leftarrow\$ \mathcal{Y}$
  $T \leftarrow T \cup \{(x, y)\}$
return y

Figure 4: The "multi-user existential unforgeability against chosen message attacks and semi-adaptive corruptions" (MU-EUF-CMA-SAC) security experiment.

Definition C.1 *We say a signature scheme is MU-EUF-CMA-SAC-secure for some $q \geq 1, N \geq 1$ if any adversary has at most negligible advantage against the experiment given in Figure 4. More precisely there exists some negligible function* negl *such that for all PPT adversaries $\mathcal{A}$ we have that*

$$\mathsf{Adv}^{MU\text{-}EUF\text{-}CMA\text{-}SAC}_{\mathcal{A}}(\lambda, q, N) = \Pr[\mathsf{Exp}^{MU\text{-}EUF\text{-}CMA\text{-}SAC}_{\mathcal{A}}(1^\lambda, q, N) = 1] \leq \mathsf{negl}(\lambda),$$

where the probability is taken over the random choice of RO*, the random coins of the experiment and the random coins of $\mathcal{A}$. For xxx = EUF-CMA we may omit N and write* $\mathsf{Exp}^{EUF\text{-}CMA}_{\mathcal{A}}(1^\lambda, q)$.

HaSS with Redundant Keys. We will now describe our scheme as a transformation of a tight unique MU-EUF-CMA secure scheme, which we will refer to as the inner signature scheme $(\mathsf{Setup}^{\mathsf{in}}, \mathsf{Gen}^{\mathsf{in}}, \mathsf{Sign}^{\mathsf{in}}, \mathsf{Ver}^{\mathsf{in}})$. Let $\mathcal{H}_3 : 0,1^* \to \{0,1\}$ be a hash function modelled as a random oracle. Our *outer* signature scheme S consists of the algorithms $(\mathsf{Setup}, \mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ which we will each describe below:

- The Setup algorithm is identical to that of the inner signature scheme.
- The key generation algorithm generates two key pairs $(\mathsf{sk}^{(1)}, \mathsf{pk}^{(1)}), (\mathsf{sk}^{(2)}, \mathsf{pk}^{(2)})$. Then it uses each key pair to sign the number zero as a message, let $\sigma^{(1)}\sigma^{(2)}$ be the respective signatures. Finally, the algorithm computes a hash $k = \mathcal{H}_3(\mathsf{pk}^{(1)}, \sigma^{(1)}, \mathsf{pk}^{(2)}, \sigma^{(2)})$ it then outputs $(\mathsf{pk}^{(1)}, \mathsf{pk}^{(2)})$ as a public key and $(\mathsf{sk}^{(k)}, \sigma^{(1)}, \sigma^{(2)})$ as secret key, the key $\mathsf{sk}^{(1-k)}$ is intentionally deleted.
- The signing algorithm uses $\mathsf{sk}^{(k)}$ to sign a message using the inner signature algorithm. If the resulting inner signature is σ then the outer signature is $(\sigma, (\sigma^{(1)}, \sigma^{(2)}))$.
- For a signature of the form $(\sigma, (\sigma^{(1)}, \sigma^{(2)}))$ the verification algorithm first uses the inner verification algorithm to check $1 = \mathsf{Ver}^{\mathsf{in}}(\mathsf{pk}^{(i)}, \sigma^{(i)}, 0)$ for $i \in \{1, 2\}$, if this holds, then the algorithm computes $k = \mathcal{H}_3(\mathsf{pk}^{(1)}, \sigma^{(1)}, \mathsf{pk}^{(2)}, \sigma^{(2)})$. Finally, the inner verification algorithm is used to confirm $1 = \mathsf{Ver}^{\mathsf{in}}(\mathsf{pk}^{(k)}, \sigma, m)$.

Our transformation essentially adds a second public key and a verification step to check which of the two keys is the 'true' public key. For practical implementations, we only have to send the signatures $(\sigma^{(1)}, \sigma^{(2)})$ once, after which the scheme becomes as expensive as the inner scheme.

Lemma C.2 *Let* $(\mathsf{Setup}^{\mathsf{in}}, \mathsf{Gen}^{\mathsf{in}}, \mathsf{Sign}^{\mathsf{in}}, \mathsf{Ver}^{\mathsf{in}})$ *be a unique signature scheme and let* $(\mathsf{Setup}, \mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ *be the outer signature scheme constructed using* S^{in} *as an inner signature scheme, following the above transformation. If the inner signature scheme* S^{in} *is MU-EUF-CMA secure, then* S *is MU-EUF-CMA-SAC secure, where for both schemes we need to perform approximately the same amount of work to beat their respective security games. Specifically, for any PPT EUF-CMA adversary* $\mathcal{A}$ *against* S *there exists a PPT RUF-RMA adversary* $\mathcal{R}^{\mathcal{A}}$ *such that*

$$\Pr[\mathsf{Exp}^{\textit{MU-EUF-CMA-SAC}}_{\mathcal{A}}(1^\lambda, q, N) = 1] \leq \Pr[\mathsf{Exp}^{\textit{MU-EUF-CMA}}_{\mathcal{R}^{\mathcal{A}}}(1^\lambda, q+1, N) = 1]$$

where the randomness is taken over the random choice of RO *and the random coins of* $\mathcal{A}$, $\mathcal{R}^{\mathcal{A}}$ *and the security games. The variable* q_{RO} *denotes the amount of random oracle queries made by* $\mathcal{A}$*. Additionally, the algorithm* $\mathcal{R}^{\mathcal{A}}$ *is PPT and only runs* $\mathcal{A}$ *once.*

Proof. We give a short proof sketch. We describe the reduction which uses an MU-EUF-CMA-SAC adversary against the outer scheme to attack the MU-EUF-CMA security of the inner scheme. This reduction receives N public keys from its own security game, it additionally generates N public keys itself. We call these two sets of keys the challenge keys and corrupted keys respectively, notably we only know the corresponding secret keys for the "corrupted" keys. For each $i \in [N]$ we flip a coin $c \leftarrow\!\!\$\ 0,1$ and define $\mathsf{pk}_i^{(c)}$ as the i-th corrupted key and $\mathsf{pk}_i^{(1-c)}$ as the i-th challenge key. The keys $(\mathsf{pk}_i^{(0)}, \mathsf{pk}_i^{(1)})$ are then sent to the adversary as the public keys of the outer scheme.

On the first corruption or signature query of user i the reduction will query its own signing oracle for user i on the message: 0., while it computes the other inner signature on 0 for the corrupted key itself. If this first query is a corruption query, then the reduction defines $\mathcal{H}_3(\mathsf{pk}_i^{(1)}, \sigma^{(1)}, \mathsf{pk}_i^{(2)}, \sigma^{(2)}) := c$, otherwise it defines the hash as $1 - c$ instead. Either way, the hash outcome is still generated uniformly at random.

The result is that for corruption queries from $\mathcal{A}$ we return our own generated corrupted key pair, while for uncorrupted parties we instead use the challenge public key. To answer signature queries the reduction is therefore required to use the signing oracle of its own security game.

The reduction wins its security game if $\mathcal{A}$ provides the reduction with a fresh forgery on a challenge public key. There are two ways this can occur: (1) if the adversary queries $\mathcal{H}_3(\mathsf{pk}_i^{(1)}, \sigma^{(1)}, \mathsf{pk}_i^{(2)}, \sigma^{(2)})$ for the correct unique inner signatures $\sigma^{(i)}$ before we have defined its outcome, then this is a fresh forgery. (2) if the adversary provides a forgery for an uncorrupted party, then per definition this must be a forgery w.r.t. the challenge public key. We therefore find that the reduction's success probability is equal to that of the adversary. □