Tight Proofs for Signature Schemes without Random Oracles

Sven Schäge

Horst Görtz Institute for IT-Security
Ruhr-University of Bochum
sven.schaege@rub.de

Abstract. We present the first tight security proofs for two general classes of Strong RSA based signature schemes. Among the affected signature schemes are the Cramer-Shoup, Camenisch-Lysyanskaya, Zhu, and Fischlin signature scheme. We also present two bilinear variants of our signature classes that produce short signatures. Similar to before, we show that these variants have tight security proofs under the the Strong Diffie-Hellman (SDH) assumption. We so obtain very efficient SDH-based variants of the Cramer-Shoup, Fischlin, and Zhu signature scheme and the first tight security proof of the recent Camenisch-Lysyanskaya scheme that was proposed and proven secure under the SDH assumption. Central to our results is a new proof technique that allows the simulator to avoid guessing which of the attacker's signature queries are re-used in the forgery. In contrast to previous proofs, our security reduction does not lose a factor of q here.

Keywords: signature class, tight security, SRSA, SDH, standard model.

1 Introduction

Provable Security and Tight Reductions. The central idea of provable security is to design a cryptographic scheme in such a way that if an attacker $\mathcal{A}$ could efficiently break its security properties then one can also construct an efficient algorithm $\mathcal{B}$, to break a supposedly hard problem. In this way, we prove the security of the scheme *by reduction* from the hardness assumption. Now, if $\mathcal{B}$ has almost the same success probability as $\mathcal{A}$ while running in roughly the same time we say that the security reduction is tight. Otherwise, the security reduction is said to be loose. It is no secret why cryptographers are interested in tight security proofs: besides being theoretically interesting, they allow for shorter security parameters and better efficiency. This work was also motivated by the observation that for several of the existing Strong RSA (SRSA) based signature schemes without random oracles we do not know if tight security proofs exist. Those schemes which we know to have a tight security proof, also have some limitations concerning practicability (which in turn cannot be found among the signature schemes with a loose security reduction). In 2007, Chevallier-Mames and Joye addressed this problem in the following way [6]: they took a tightly secure signature scheme, the Gennaro-Halevi-Rabin scheme [10], and improved its efficiency by re-designing one of its most time-consuming functions. The problem with such an approach is that it only affects *new* implementations of the considered signature scheme. Therefore, we take the same approach as Bernstein at EUROCRYPT '08 who proved tight security for the *original* Rabin-Williams signature scheme in the random-oracle model [2]. However, in contrast to Bernstein we concentrate on schemes that are secure in the standard model.

Contribution. In this work, we ask the following question: are there tight security proofs for the existing practical signature schemes by Cramer-Shoup [8], Zhu [19], Camenisch-Lysyanskaya [4] and Fischlin [9] (which we only know to have loose security reductions)? We answer this question in the affirmative and present the first tight proofs for the above signature schemes. However, our result is not limited to the original schemes. In our analysis, we generalize the schemes by Camenisch-Lysyanskaya, Fischlin and Zhu by introducing a new family of randomization functions, called combining functions. The result of this generalization is an abstract signature scheme termed 'combining scheme'. In a similar way, we introduce a second general class of signature schemes called 'chameleon hash scheme' that can be regarded as a generalization of the Cramer-Shoup signature scheme. Then, we prove the combining signature scheme and the chameleon hash scheme to be *tightly* secure under the SRSA assumption when instantiated with any secure combining function, respectively chameleon hash function. Finally, we show that our results do not only hold under the SRSA assumption. We analyze whether there also exist tight security reductions for analogous schemes based on the SDH assumption in bilinear groups. Interestingly, most of the above schemes have not been considered yet under the SDH assumption (except for the Camenisch-Lysyanskaya scheme), although, at the same security level, the group description is much shorter in bilinear groups than in factoring based groups. We develop an SDH-based variant of the combining signature scheme and the chameleon hash scheme and prove it to be existentially unforgeable under adaptive chosen message attacks with a *tight* security reduction. In doing so, we present the first SDH-based variants of the Fischlin, the Zhu and the Cramer-Shoup signature scheme and the first tight security proof of the SDH-based Camenisch-Lysyanskaya scheme. When instantiated with existing combining functions (respectively chameleon hash functions), we obtain short and efficient signature schemes. All our results (on the combining class) can easily be extended to signature schemes for message blocks (as defined in [4,5]), where we can even use distinct combining functions for each message block. Our results can be interpreted in two *positive* ways: 1) Existing implementations of the affected signature schemes (with a fixed parameter size) provide higher security than expected. 2) New implementations can have shorter security parameters what transfers to higher efficiency.

Technical Contribution. In the existing proofs, the simulator partitions the set of forgeries by at first guessing $j \in \{1, \ldots, q\}$ where q is the number of signature queries made by the attacker. Only if the attacker's forgery shares some common values with the answer to the j-th signature query the simulator can break the SRSA assumption. Otherwise the simulator just aborts. The number of signature queries rises polynomially in the security parameter and the security proof loses a factor of q here. Our main contribution is a new technique that renders the initial guess unnecessary. As a consequence, *any* forgery helps the simulator to break the SRSA assumption. This results in a tight security proof.

Related Work. Our work is related to the existing hash-and-sign signature schemes without random oracles that are proven secure under the SRSA or the SDH assumption. We subsequently give a brief overview on the available results. In 1988, Goldwasser, Micali and Rivest published the first provably secure, but inefficient signature scheme [11]. More than a decade later, in 1999, Gennaro, Halevi, and Rabin [10] presented a signature scheme that is secure in the standard model under the Flexible or Strong RSA assumption (SRSA). This scheme is more efficient, both the key and the signature size are less than two group elements (à 1024 bits), but as a drawback, it relies on an impractical function that injectively maps messages to primes [7]. Advantageously, the Gennaro-Halevi-Rabin signature scheme is known to have a tight security proof. At the same time and also based on the SRSA assumption, Cramer and Shoup [8] proposed an efficient standard model signature scheme, that unlike [10] does not require to map messages to primes. In contrast, primes can be drawn uniformly at random from the set of primes of a given bitlength. Based on this work, Zhu [18,19], Fischlin [9], Camenisch and Lysyanskaya [4], and Hofheinz and Kiltz [12] in the following years presented further SRSA-based schemes. These schemes are either more efficient than the Cramer-Shoup scheme or very suitable in protocols for issuing signatures on committed values. In 2004, Boneh and Boyen presented the first hash-and-sign signature scheme that makes use of bilinear groups [3]. The big advantage of bilinear groups is the very compact representation of group elements. The Boneh-Boyen signature scheme is proven tightly secure under a new flexible assumption, the q-Strong Diffie Hellman (SDH) assumption. In 2004, Camenisch and Lysyanskaya also presented a signature scheme that relies on bilinear groups [5]. Unlike the Boneh-Boyen scheme, their scheme is proven secure under the LRSW [14] assumption. However, in the same paper they also propose a variant that is based on the SDH assumption in bilinear groups. The corresponding security proof was provided four years later in [15,1]. Similar to the original Camenisch-Lysyanskaya scheme the security proof of the SDH scheme is loose.

2 Preliminaries

Before presenting our results we briefly review the necessary formal and mathematical definitions. For convenience, we also describe two general setup and key generation procedures (settings) in Section 2.7 and Section 2.8. When describing our signature schemes in Sections 3.1, 3.2, 3.5 we will refer to the corresponding setting and only describe the signature generation and verification algorithms.

2.1 Notation

For $a, b \in \mathbb{Z},\ a \leq b$ we write $[a; b]$ to denote the set $\{a, a+1, \ldots, b-1, b\}$. For a string x, we write $|x|_2$ to denote its bit length. If $z \in \mathbb{Z}$, we write $|z|$ to denote the absolute value of z. For a set X, we use $|X|$ to refer to its size and $x \overset{\$}{\leftarrow} X$ to indicate that x is drawn from X uniformly at random. For $n \in \mathbb{N}$, we use QR_n to denote the set of quadratic residues modulo n, i.e. $QR_n = \{x | \exists y \in \mathbb{Z}_n^* : y^2 = x \bmod n\}$. If $\mathcal{A}$ is an algorithm we use $\mathcal{A}(x_1, x_2, \ldots)$ to denote that $\mathcal{A}$ has input parameters $x_1, x_2, \ldots$. Accordingly, $y \leftarrow \mathcal{A}(x_1, x_2, \ldots)$ means that $\mathcal{A}$ outputs y when running with inputs $x_1, x_2, \ldots$. PPT refers to probabilistic polynomial-time. We write $\kappa \in \mathbb{N}$ to indicate the security parameter and 1^κ to describe the string that consist of κ ones. In the following, we implicitly assume that the size of the generated key material is always polynomially dependent on the security parameter.

2.2 Signature Scheme

A digital signature scheme $\mathcal{S}$ consists of three algorithms. The PPT algorithm KeyGen on input 1^κ generates a secret and public key pair (SK, PK). The PPT algorithm Sign takes as input a secret key SK and the message m and outputs a signature σ. Finally, the deterministic polynomial time algorithm Verify takes a public key PK, a message m and a signature σ to check whether σ is a legitimate signature on m signed by the holder of the secret key corresponding to PK. Accordingly, the algorithm outputs 1 to indicate a successful verification and 0 otherwise.

2.3 Strong Existential Unforgeability

The standard notion of security for signature schemes is due to Goldwasser, Micali and Rivest [11]. In this paper, we use a slightly stronger definition called strong existential unforgeability. The signature scheme $\mathcal{S} = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ is strongly existentially unforgeable under an adaptive chosen message attack if it is infeasible for a forger, who only knows the public key and the global parameters, to produce, after obtaining polynomially (in the security parameter) many signatures $\sigma_1, \ldots, \sigma_q$ on messages $m_1, \ldots, m_q$ of its choice from a signing oracle $\mathcal{O}(SK, \cdot)$, a new message/signature pair.

Definition 1. *We say that $\mathcal{S}$ is (q, t, ϵ)-secure, if for all t-time adversaries $\mathcal{A}$ that send at most q queries to the signing oracle $\mathcal{O}(SK, \cdot)$ it holds that*

$$\Pr\begin{bmatrix}(SK, PK) \leftarrow \mathsf{KeyGen}(1^\kappa),\ (m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathcal{O}(SK,\cdot)}(PK), \\ \mathsf{Verify}(PK, m^*, \sigma^*) = 1\end{bmatrix} \leq \epsilon,$$

where the probability is taken over the random coins of KeyGen *and $\mathcal{A}$ and (m^*, σ^*) is not among the message/signature pairs obtained using $\mathcal{O}(SK, \cdot)$ (i.e. $(m^*, \sigma^*) \notin \{(m_1, \sigma_1), \ldots, (m_q, \sigma_q)\}$).*

2.4 Collision-Resistant Hashing

Definition 2 (Collision-resistant hash function). *Let $\mathcal{H}_k$ for $k \in \mathbb{N}$ be a collection of functions of the form $h : \{0,1\}^* \to \{0,1\}^k$. Let $\mathcal{H} = \{\mathcal{H}_k\}_{k \in \mathbb{N}}$. $\mathcal{H}$ is called (t_h, ϵ_h)-collision-resistant if for all t_h-time adversaries $\mathcal{A}$ it holds that*

$$\Pr\left[\begin{array}{c} h \xleftarrow{\$} \mathcal{H}_k,\ (m, m') \leftarrow \mathcal{A}(h),\ m \neq m', \\ m, m' \in \{0,1\}^*,\ h(m) = h(m') \end{array}\right] \leq \epsilon_h,$$

where the probability is over the random bits of $\mathcal{A}$.

2.5 Chameleon Hash Function

A chameleon hash function $\mathcal{CH} = (\mathsf{CHGen}, \mathsf{CHEval}, \mathsf{CHColl})$ consists of three algorithms [13]. The PPT algorithm CHGen takes as input the security parameter κ and outputs a secret key $SK_{\mathcal{CH}}$ and a public key $PK_{\mathcal{CH}}$. Given $PK_{\mathcal{CH}}$, a random r from a randomization space $\mathcal{R}$ and a message m from a message space $\mathcal{M}$, the algorithm CHEval outputs a chameleon hash value c in the hash space $\mathcal{C}$. Analogously, CHColl deterministically outputs, on input $SK_{\mathcal{CH}}$ and $(r, m, m') \in \mathcal{R} \times \mathcal{M} \times \mathcal{M}$, $r' \in \mathcal{R}$ such that $\mathsf{CHEval}(PK_{\mathcal{CH}}, m, r) = \mathsf{CHEval}(PK_{\mathcal{CH}}, m', r')$.

Definition 3 (Collision-resistant chameleon hash function). *We say that $\mathcal{CH}$ is $(\epsilon_{\mathcal{CH}}, t_{\mathcal{CH}})$-collision-resistant if for all $t_{\mathcal{CH}}$-time adversaries $\mathcal{A}$ that are only given $PK_{\mathcal{CH}}$ it holds that*

$$\Pr\left[\begin{array}{c} (SK_{\mathcal{CH}}, PK_{\mathcal{CH}}) \leftarrow \mathsf{CHGen}(1^\kappa),\ (m, m', r, r') \leftarrow \mathcal{A}(PK_{\mathcal{CH}}), \\ r, r' \in \mathcal{R},\ m, m' \in \mathcal{M},\ m' \neq m, \\ \mathsf{CHEval}(PK_{\mathcal{CH}}, r, m) = \mathsf{CHEval}(PK_{\mathcal{CH}}, r', m') \end{array}\right] \leq \epsilon_{\mathcal{CH}},$$

where the probability is over the random choices of $PK_{\mathcal{CH}}$ and the coin tosses of $\mathcal{A}$.

We also require that for an arbitrary but fixed public key $PK_{\mathcal{CH}}$ output by CHGen, all messages $m \in \mathcal{M}$ generate equally distributed hash values when drawing $r \in \mathcal{R}$ uniformly at random and outputting $\mathsf{CHEval}(PK_{\mathcal{CH}}, r, m)$. We write $ch(r, m)$ to denote $\mathsf{CHEval}(PK_{\mathcal{CH}}, r, m)$ and $ch^{-1}(r, m, m')$ for $\mathsf{CHColl}(SK_{\mathcal{CH}}, r, m, m')$ if the keys are obvious from the context. The security of chameleon hash functions can be based on very standard assumptions like the discrete logarithm assumption [13] or the factoring assumption [13,17] which are weaker than the SDH, respectively the SRSA assumption.

2.6 Combining Function

In this section, we introduce a new family of functions called combining functions. We will subsequently use the concept of combining functions to generalize several existing signature schemes.

Definition 4 (Combining Functions). *Let $\mathcal{V}_k$ for $k \in \mathbb{N}$ be a collection of functions of the form $z : \mathcal{R} \times \mathcal{M} \rightarrow \mathcal{Z}$ with $|\mathcal{Z}| \leq 2^k$. Let $\mathcal{V} = \{\mathcal{V}_k\}_{k \in \mathbb{N}}$. We say that $\mathcal{V}$ is $(t_{comb}, \epsilon_{comb}, \delta_{comb})$-combining if for all attackers $\mathcal{A}$ there exist negligible functions $\epsilon_{comb}(k)$ and $\delta_{comb}(k)$ and the following properties hold for $z \overset{\$}{\leftarrow} \mathcal{V}_k$.*

1. *for all $m \in \mathcal{M}$ it holds that $|\mathcal{R}| = |\mathcal{Z}_m|$ where $\mathcal{Z}_m$ is defined as $\mathcal{Z}_m = z(\mathcal{R}, m)$. For all $m \in \mathcal{M}$ and all $t \in \mathcal{Z}$ there exists an efficient algorithm $z^{-1}(t, m)$ that, if $t \in \mathcal{Z}_m$, outputs the unique value $r \in \mathcal{R}$ such that $z(r, m) = t$, and $\perp$ otherwise.*
2. *for $t \overset{\$}{\leftarrow} \mathcal{Z}$ and $r' \overset{\$}{\leftarrow} \mathcal{R}$ we have for the maximal (over all $m \in \mathcal{M}$) statistical distance between r' and $z^{-1}(t, m)$ that*

$$\max_{m \in \mathcal{M}} \left\{ 1/2 \cdot \sum_{r \in \mathcal{R}} \left| \Pr[r' = r] - \Pr[z^{-1}(t, m) = r] \right| \right\} \leq \delta_{comb}.$$

3. *for all $r \in \mathcal{R}$, it holds for all t_{comb}-time attackers $\mathcal{A}$ that*

$$\Pr\left[\begin{matrix}(m, m') \leftarrow \mathcal{A}(z, r),\ m, m' \in \mathcal{M}, \\ m \neq m',\ z(r, m) = z(r, m')\end{matrix}\right] \leq \epsilon_{comb},$$

where the probability is taken over the random bits of $\mathcal{A}$.

In the following, we assume that when used in signature schemes, $z \overset{\$}{\leftarrow} \mathcal{V}_k$ is chosen uniformly at random during the key generation phase.

Table 1. Examples of statistically secure combining functions. Let $\mathcal{V} = \{\mathcal{V}_k\}_{k \in \mathbb{N}}$ with $\mathcal{V}_k = \{z(r, m)\}$, $l, l_r, l_m \in \mathbb{N}$, $l_r > l_m$ and p be prime.

	$z(r,m)$	$\mathcal{R}$	$\mathcal{M}$	$\mathcal{Z}$	combining
EX1	$r + m \bmod p$	$\mathbb{Z}_p$	$\mathbb{Z}_p$	$\mathbb{Z}_p$	$(\cdot, 0, 0)$
EX2	$r \oplus m$	$\{0,1\}^l$	$\{0,1\}^l$	$\{0,1\}^l$	$(\cdot, 0, 0)$
EX3	$r + m$	$[0; 2^{l_r} - 1]$	$[0; 2^{l_m} - 1]$	$[0; 2^{l_r} + 2^{l_m} - 2]$	$(\cdot, 0, 2^{l_m - l_r})$

In Table 1, we present three concrete examples (EX1, EX2, EX3) of statistically secure combining functions. The following lemma shows that these examples are valid combining functions with respect to Definition 4.

Lemma 1. EX1 *and* EX2 *constitute* $(\cdot, 0, 0)$*-combining functions and* EX3 *constitutes a* $(\cdot, 0, 2^{l_m - l_r})$*-combining function.*

Proof. Let us first analyze EX1 and EX2. We have that $\mathcal{M} = \mathcal{R} = \mathcal{Z} = \mathcal{Z}_m$ for all $m \in \mathcal{M}$ and we can efficiently compute r as $r = t - m \bmod p$ or $r = t \oplus m$ for *all* given $t \in \mathcal{Z}$ and $m \in \mathcal{M}$. Furthermore, since z is bijective in both input parameters $z^{-1}(t, m)$ is uniformly distributed in $\mathcal{R}$ for all $m \in \mathcal{M}$ and random $t \in \mathcal{Z}$. Thus, $\delta_{\text{comb}} = 0$. Finally, since z is a bijection in the second input parameter, it is collision-free (property 3) in both examples and we have that $\epsilon_{\text{comb}} = 0$. Now, let us analyze EX3. For given $m \in \mathcal{M}$ and $t \in \mathcal{Z}$, $z^{-1}(t, m)$ outputs $r = t - m$ if $t - m \in \mathcal{R}$ and $\perp$ otherwise. To show that z is collision-free, observe that $m \neq m'$ implies $r + m \neq r + m'$ for all $r \in \mathcal{R}$. To analyze $D = \max_{m \in \mathcal{M}} \left\{ 1/2 \cdot \sum_{r \in \mathcal{R}} \left| \Pr[r' = r] - \Pr[z^{-1}(t, m) = r] \right| \right\}$ first note that for $t' \overset{\$}{\leftarrow} \mathcal{Z}_m$, $z^{-1}(t', m)$ is uniform in $\mathcal{R}$ since $|\mathcal{Z}_m| = |\mathcal{R}|$ implies that $z^{-1}(\cdot, m)$ defines a bijection from $\mathcal{Z}_m$ to $\mathcal{R}$. For $t' \overset{\$}{\leftarrow} \mathcal{Z}_m$ and $t \overset{\$}{\leftarrow} \mathcal{Z}$ we get

$$D \leq \max_{m \in \mathcal{M}} \left\{ 1/2 \cdot \sum_{t_0 \in \mathcal{Z}_m} \left| \Pr[t' = t_0] - \Pr[t = t_0] \right| \right\} \leq 2^{l_m - l_r}$$

Three further examples of combining functions can be obtained when first applying a (t_h, ϵ_h)-collision-resistant hash function that maps (long) messages to $\mathcal{M}$. Lemma 2 guarantees that the results are still combining according to Definition 4. The proof of Lemma 2 is straight-forward and can be found in the full version.

Lemma 2. *Let $\mathcal{V}$ be a $(t_{comb}, \epsilon_{comb}, \delta_{comb})$-combining function and $\mathcal{H}$ be a (t_h, ϵ_h)-collision-resistant hash function. Then $\mathcal{V}' = \{\mathcal{V}'_k\}_{k \in \mathbb{N}}$ with $\mathcal{V}'_k = \{z(r, h(m)) | z \overset{\$}{\leftarrow} \mathcal{V}_k, h \overset{\$}{\leftarrow} \mathcal{H}_k\}$ is $(\min\{t_{comb}, t_h\}, \epsilon_{comb} + \epsilon_h, \delta_{comb})$-combining.*

2.7 The Strong RSA Setting

Definition 5 (Strong RSA assumption (SRSA)). *Given an RSA modulus $n = pq$, where p, q are sufficiently large primes, and an element $u \in \mathbb{Z}_n^*$, we say that the $(t_{SRSA}, \epsilon_{SRSA})$-SRSA assumption holds if for all t_{SRSA}-time adversaries $\mathcal{A}$*

$$\Pr\left[(x, y) \leftarrow \mathcal{A}(n, u),\ x \in \mathbb{Z}_n^*,\ y > 1,\ x^y = u \bmod n\right] \leq \epsilon_{SRSA},$$

where the probability is over the random choices of u, n and the random coins of $\mathcal{A}$.

Definition 6 (SRSA setting). *In this setting,* $\mathsf{KeyGen}(1^\kappa)$ *outputs the key pair $(SK = (p, q), PK = n)$ for a safe modulus $n = pq$ such that $p = 2p' + 1$, $q = 2q' + 1$, and p, q, p', q' are primes. All computations are performed in the cyclic group QR_n. Let $l_i = l_i(\kappa)$ for $i \in \{n, t, c, e, m\}$ be polynomials. We require that $|n|_2 = l_n$ and $|p'|_2 = |q'|_2 = l_n/2 - 1$. Furthermore, we assume that the $(t_{SRSA}, \epsilon_{SRSA})$-SRSA assumption holds. We let u, v, w be public random generators of QR_n with unknown $\log_u v$, $\log_u w$, and $\log_v w$. When using combining functions $z(r, m)$, we assume that $\mathcal{M} \subseteq [0; 2^{l_m} - 1]$, $\mathcal{Z} \subseteq [0; 2^{l_z} - 1]$ and $\mathcal{R} \subseteq [0; 2^{l_r} - 1]$. We let $E \subseteq [2^{l_e - 1}; 2^{l_e} - 1]$ denote the set of l_e-bit primes. Finally, we require that $l_m \leq l_c, l_z, l_r < l_e < l_n/2 - 1$.*

2.8 The Strong Diffie-Hellman Setting

Definition 7 (Bilinear groups). *Let $\mathbb{G}_1 = <g_1>, \mathbb{G}_2 = <g_2>$ and $\mathbb{G}_T$ be groups of prime order p. The function $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ is a bilinear pairing if it holds that 1) for all $a \in \mathbb{G}_1$, $b \in \mathbb{G}_2$, and $x, y \in \mathbb{Z}_p$ we have $e(a^x, b^y) = e(a, b)^{xy}$ (bilinearity), 2) $e(g_1, g_2) \neq 1_{\mathbb{G}_T}$ is a generator of $\mathbb{G}_T$ (non-degeneracy), and 3) e is efficiently computable (efficiency). We call $(\mathbb{G}_1, g_1, \mathbb{G}_2, g_2, \mathbb{G}_T, p, e)$ a bilinear group.*

Definition 8 (SDH assumption (SDH)). *Let $(\mathbb{G}_1, \hat{g}_1, \mathbb{G}_2, \hat{g}_2, \mathbb{G}_T, p, e)$ be a bilinear group. We say that the $(q_{SDH}, t_{SDH}, \epsilon_{SDH})$-SDH assumption holds if for all t_{SDH}-time attackers $\mathcal{A}$ that are given a $(q_{SDH} + 3)$-tuple of elements $W = \left(g_1, g_1^x, g_1^{(x^2)}, \ldots, g_1^{(x^{q_{SDH}})}, g_2, g_2^x\right) \in \mathbb{G}_1^{q_{SDH}+1} \times \mathbb{G}_2^2$ it holds that*

$$\Pr\left[(s, c) \leftarrow \mathcal{A}(W),\ c \in \mathbb{Z}_p,\ s \in \mathbb{G}_1,\ e(s, g_2^x g_2^c) = e(g_1, g_2)\right] \leq \epsilon_{SDH},$$

where the probability is over the random choices of the generators $g_1 \in \mathbb{G}_1$, $g_2 \in \mathbb{G}_2$, $x \in \mathbb{Z}_p$ and the random bits of $\mathcal{A}$.

Definition 9 (SDH setting). *In the SDH setting, all computations are performed in the cyclic groups of $(\mathbb{G}_1, g_1, \mathbb{G}_2, g_2, \mathbb{G}_T, p, e)$ such that $|p|_2 = l_p = l_p(\kappa)$. The PPT $\mathsf{KeyGen}(1^\kappa)$ chooses $x \overset{\$}{\leftarrow} \mathbb{Z}_p$ and outputs $(SK = x, PK = g_2^x)$. We assume that the $(q_{SDH}, t_{SDH}, \epsilon_{SDH})$-SDH assumption holds. Finally, we suppose that the values $a, b, c \in \mathbb{G}_1$ are public random generators of $\mathbb{G}_1$ such that $\log_a b$, $\log_a c$, and $\log_b c$ are unknown. In case of combining functions $z(r, m)$, we assume that $\mathcal{Z} \subseteq \mathbb{Z}_p$ and $\mathcal{R} \subseteq \mathbb{Z}_p$.*

3 Signature Classes

For convenience, we now introduce two general signature classes. The combining signature scheme $\mathcal{S}_{\text{CMB}}$ constitutes a useful abstraction of the Camenisch-Lysyanskaya, the Fischlin, and the Zhu signature scheme using combining functions. The chameleon signature scheme $\mathcal{S}_{\text{CH}}$ can be regarded as a general variant of the original Cramer-Shoup signature scheme where we do not specify a concrete instantiation of the chameleon hash function.

3.1 SRSA-Based Combining Signature Scheme $\mathcal{S}_{\text{CMB,SRSA}}$

In the SRSA setting, $\mathsf{Sign}(SK, m)$ randomly draws $r \in \mathcal{R}$ and $e \in E$ and computes a signature $\sigma = (r, s, e)$ on message m with $s = \left(uv^r w^{z(r,m)}\right)^{\frac{1}{e}}$. Let us now show that our construction generalizes the claimed signature schemes. Observe that we can easily obtain the Fischlin scheme [9] if we instantiate the combining function with EX2 of Table 1. Furthermore, we can also get the Camenisch-Lysyanskaya scheme [4] using EX3. This becomes obvious if we substitute v by

$v' = vw$ as $uv^r w^{r+m} = u(vw)^r w^m = u(v')^r w^m$[1]. We note that when we use the Camenisch-Lysyanskaya scheme with long messages we must first apply a collision-resistant hash function to the message. What we essentially get is Zhu's scheme [18,19]. By Lemma 2, the resulting function is still combining. The verification routine $\mathsf{Verify}(PK, m, \sigma)$ takes a purported signature $\sigma = (r, s, e)$ and checks if $s^e \overset{?}{=} uv^r w^{z(r,m)}$, if $|e|_2 = l_e$, and if e is odd.

3.2 SDH-Based Combining Signature Scheme $\mathcal{S}_{\mathrm{CMB,SDH}}$

We also present an SDH-based variant $\mathcal{S}_{\mathrm{CMB,SDH}}$ of the combining signature scheme. We remark that for the Camenisch-Lysyanskaya scheme there already exists a corresponding SDH-based variant, originally introduced in [5] and proven secure in [15,1]. Similar to $\mathcal{S}_{\mathrm{CMB,SRSA}}$, we obtain the SDH-based Camenisch-Lysyanskaya scheme when instantiating the combining function with EX1. In the same way, we can also get SDH-based variants of the Fischlin signature scheme (using EX2) and of Zhu's scheme (using Lemma 2). In the SDH-based combining scheme, $\mathsf{Sign}(SK, m)$ at first chooses a random $r \in \mathcal{R}$ and a random $t \in \mathbb{Z}_p \setminus \{-x\}$. It then computes the signature $\sigma = (r, s, t)$ with $s = \left(ab^r c^{z(r,m)}\right)^{\frac{1}{x+t}}$. Given a signature $\sigma = (r, s, t)$, $\mathsf{Verify}(PK, m, \sigma)$ checks if $e\left(s, PK g_2^t\right) \overset{?}{=} e\left(ab^r c^{z(r,m)}, g_2\right)$.

3.3 SRSA-Based Chameleon Hash Signature Scheme $\mathcal{S}_{\mathrm{CH,SRSA}}$

The scheme $\mathcal{S}_{\mathrm{CH,SRSA}}$ is defined in the SRSA setting. $\mathsf{KeyGen}(1^\kappa)$ additionally generates the key material $(SK_{\mathcal{CH}}, PK_{\mathcal{CH}})$ for a chameleon hash function. The value $PK_{\mathcal{CH}}$ is added to the scheme's public key. ($SK_{\mathcal{CH}}$ is not required. However, it may be useful when turning the signature scheme into an online-offline signature scheme [17].) The signature generation algorithm $\mathsf{Sign}(SK, m)$ first chooses a random $r \in \mathcal{R}$ and a random prime $e \in E$. It then outputs the signature $\sigma = (r, s, e)$ on a message m where $s = \left(uv^{ch(r,m)}\right)^{\frac{1}{e}}$. To verify a purported signature $\sigma = (r, s, e)$ on m, $\mathsf{Verify}(PK, m, \sigma)$ checks if e is odd, if $|e|_2 = l_e$, and if $s^e \overset{?}{=} uv^{ch(r,m)}$.

3.4 SDH-Based Chameleon Hash Signature Scheme $\mathcal{S}_{\mathrm{CH,SDH}}$

Let us now define a new variant of the chameleon hash signature scheme that is based on the SDH assumption. Again, $\mathsf{KeyGen}(1^\kappa)$ also adds the public key

[1] To be precise, our generalization slightly differs from the Camenisch-Lysyanskaya scheme. In the original scheme, it is required that $l_r = l_n + l_m + 160$. As a result, the authors recommend for 160 bit long messages that $l_r = 1346$, $l_s = 1024$, and $l_e = 162$. In our scheme, we simply require that $l_m \leq l_r < l_e < l_n/2 - 1$. Then, we can set $l_r = 320$, $l_s = 1024$, and $l_e = 321$ for a probability $\epsilon_{\mathrm{comb}} = 2^{-160}$. Therefore, the signature size of our signature scheme is much shorter (only $(320 + 1024 + 321)/(1346 + 1024 + 162) \approx 66\%$ of the original signature size) and the scheme is more efficient (since shorter exponents imply faster exponentations) than the original scheme.

$PK_{\mathcal{CH}}$ of a chameleon hash function to PK. In the SDH setting, $\mathsf{Sign}(SK, m)$ first chooses a random $r \in \mathcal{R}$ and a random $t \in \mathbb{Z}_p \backslash \{-x\}$. Using $SK = x$, it then outputs the signature σ on m as $\sigma = (r, s, t)$ where $s = \left(ab^{ch(r,m)}\right)^{\frac{1}{x+t}}$. To verify a given signature $\sigma = (r, s, t)$ on m, $\mathsf{Verify}(PK, m, \sigma)$ checks if $e\left(s, PKg_2^t\right) \overset{?}{=} e\left(ab^{ch(r,m)}, g_2\right)$. A suitable chameleon hash function can for example be found in [13].

3.5 The Cramer-Shoup Signature Scheme $\mathcal{S}_{\mathbf{CS,SRSA}}$

Let us now review the Cramer-Shoup signature scheme that is defined in the SRSA setting. The Cramer-Shoup scheme $\mathcal{S}_{\mathrm{CS,SRSA}}$ additionally requires a collision-resistant hash function $h : \{0,1\}^* \rightarrow \{0,1\}^{l_c}$. The message space is so extended to $\mathcal{M} = \{0,1\}^*$. Suppose $l_c < l_e < l_n/2 - 1$.

- $\mathsf{KeyGen}(1^\kappa)$ additionally computes a random l_e-bit prime $\tilde{e}$. The secret key is $SK = (p, q)$ the public key is $PK = (n, \tilde{e})$.
- $\mathsf{Sign}(SK, m)$ first chooses a random $r \in QR_n$ and evaluates (the chameleon hash function) $c = r^{\tilde{e}}/v^{h(m)} \mod n$. Then it draws a random l_e-bit prime $e \neq \tilde{e}$ and computes the value $s = \left(uv^{h(c)}\right)^{1/e} \bmod n$. The signature is $\sigma = (r, s, e)$.
- $\mathsf{Verify}(PK, m, \sigma)$ re-computes $c = r^{\tilde{e}}/v^{h(m)} \mod n$ and checks whether $s \overset{?}{=} \left(uv^{h(c)}\right)^{1/e} \bmod n$, if e is odd, and if $|e|_2 = l_e$.

Unfortunately, the proof of the more general chameleon hash scheme class does not formally transfer to the Cramer-Shoup signature scheme because in the Cramer-Shoup scheme the key material of its chameleon hash function is not chosen independently. In particular, the chameleon hash function uses the same RSA modulus and the same value v. This requires slightly more care in the security proof. We provide a full proof of the Cramer-Shoup signature scheme in the full version.

4 Security

Theorem 1. *The Cramer-Shoup signature scheme, the combining signature class (in both the SRSA and the SDH setting), and the chameleon signature class (in both the SRSA and the SDH setting) are tightly secure against adaptive chosen message attacks. In particular, this implies that the Camenisch-Lysyanskaya, the Fischlin, the Zhu, and the SDH-based Camenisch-Lysyanskaya scheme are tightly secure against strong existential forgeries under adaptive chosen message attacks.*

We subsequently provide the intuition behind our security proofs. In Section 4.4, we present a *full* proof of security for $\mathcal{S}_{\mathrm{CMB,SRSA}}$, which seems to us to be the technically most involved reduction. The proof of $\mathcal{S}_{\mathrm{CMB,SDH}}$ proceeds analogously and appears in the full version. We then informally show how to transfer our technique to $\mathcal{S}_{\mathrm{CH}}$. In the full version we also provide a full proof of security of the Cramer-Shoup signature scheme.

4.1 The SRSA-Based Schemes

Let us first consider the SRSA-based schemes, where $\mathcal{B}$ is given an SRSA challenge $(\hat{u}, n)$ with $\hat{u} \in \mathbb{Z}_n^*$. Assume that attacker $\mathcal{A}$ issues q signature queries $m_1, \ldots, m_q \in \mathcal{M}$. As a response to each query m_i with $i \in [1; q]$, $\mathcal{A}$ receives a corresponding signature $\sigma_i = (r_i, s_i, e_i) \in \mathcal{R} \times QR_n \times E$.

Recall that the existing security proofs for schemes of the combining class (e.g. [9]) consider two forgers that loosely reduce from the SRSA assumption. This is the case when it holds for $\mathcal{A}$'s forgery $(m^*, (r^*, s^*, e^*))$ that $\gcd(e^*, \prod_{i=1}^{q} e_i) \neq 1$[2]. Given that $|e^*|_2 = l_e$ this means that $e^* = e_j$ for some $j \in [1; q]$. Let us concentrate on the case that $r^* \neq r_j$. The proof of the remaining case ($e^* = e_j$, $r^* = r_j$ and $m^* \neq m_j$) is very similar. It additionally exploits the properties of the combining function.

The proofs in [8,9,18,4,19] work as follows: the simulator $\mathcal{B}$ at first guesses $j \overset{\$}{\leftarrow} \{1, \ldots, q\}$. By construction, $\mathcal{B}$ can answer all signature queries but only if $\mathcal{A}$ outputs a forgery where $e^* = e_j$ it can extract a solution to the SRSA challenge. In all other cases (if $e^* = e_i$ for some $i \in \{1, \ldots, q\} \setminus \{j\}$), $\mathcal{B}$ just aborts. Since the number of signature queries q rises polynomially in the security parameter, the probability for $\mathcal{B}$ to correctly guess j in advance is q^{-1} and thus not negligible. However, the security reduction loses a factor of q here.

Our aim is to improve this reduction step. Ideally, we have that *any* forgery which contains $e^* \in \{e_1, \ldots, e_q\}$ helps the simulator to break the SRSA assumption. As a result, the simulator can completely avoid guessing. The main task is to re-design the way $\mathcal{B}$ computes $\mathcal{A}$'s input parameters: for *every* $i \in \{1, \ldots, q\}$, we must have exactly one choice of r_i such that $\mathcal{B}$ can simulate the signing oracle without having to break the SRSA assumption. On the other hand, if $\mathcal{A}$ outputs $(m^*, (r^*, s^*, e^*))$ with $e^* = e_i$ for some $i \in [1; q]$ and $r^* \neq r_i$, $\mathcal{B}$ must be able to compute a solution to the SRSA challenge. Let us now go into more detail.

For simplicity, assume that $\mathcal{B}$ can setup $\mathcal{A}$'s input parameters such that the verification of a signature $\sigma = (r, s, e)$ always reduces to

$$s^e = \hat{u}^{f(r)} \bmod n. \qquad (1)$$

Suppose that neither $\hat{u}$ nor $f : \mathcal{R} \rightarrow \mathbb{N}$ are ever revealed to $\mathcal{A}$. We exploit that the r_i are chosen independently at random. So, they can be specified *prior* to the signature queries. Now, $\mathcal{B}$'s strategy to simulate the signing oracle is to define $r_1, \ldots, r_q$ such that for every $i \in [1; q]$ it can compute a prime $e_i \in E$ with $e_i | f(r_i)$. Without having to break the SRSA assumption, $\mathcal{B}$ can then compute $s_i = \hat{u}^{f(r_i)/e_i}$ and output the i-th signature as (r_i, s_i, e_i).

Let us now turn our attention to the extraction phase where $\mathcal{B}$ is given $\mathcal{A}$'s forgery $(m^*, (r^*, s^*, e^*))$. By assumption we have $e^* = e_i$ for some $i \in [1; q]$ and $r^* \neq r_i$. $\mathcal{B}$ wants to have that $\gcd(e^*, f(r^*)) = D < e^*$ (or $f(r^*) \neq 0 \bmod e^*$) because then it can find a solution to the SRSA challenge by computing $a, b \in \mathbb{Z} \setminus \{0\}$ with $af(r^*)/D + be^*/D = 1$ using extended Euclidean algorithm and outputting

[2] The proof of the case $\gcd(e^*, \prod_{i=1}^{q} e_i) = 1$ is straight-forward.

$$(s^*)^a \hat{u}^b = \hat{u}^{D/e^*}, e^*/D.$$

$\mathcal{B}$'s strategy to guarantee $\gcd(e^*, f(r^*)) = D < e^*$ is to ensure that we have $e^* = e_i \not| f(r^*)$. Unfortunately, $\mathcal{B}$ cannot foresee r^*. Therefore, the best solution is to design f such that $e_i \not| f(r^*)$ for *all* $r^* \neq r_i$.

Obviously, $\mathcal{B}$ makes strong demands on f. We now present our construction of f and argue that it perfectly fulfills all requirements. We define f as

$$f(r) = \sum_{i=1}^{q} r_i \prod_{\substack{j=1\\ j\neq i}}^{q} e_j - r \sum_{i=1}^{q} \prod_{\substack{j=1\\ j\neq i}}^{q} e_j, \tag{2}$$

for $r_1, \ldots, r_q \in \mathcal{R}$. Furthermore, $e_1, \ldots, e_q \in E$ must be distinct primes. First, observe that for every $k \in [1;q]$ the function reduces to $f(r_k) = \sum_{i=1, i\neq k}^{q} (r_i - r_k) \prod_{j=1, j\neq i}^{q} e_j$ and thus $f(r_k) = 0 \bmod e_k$. On the other hand, it holds for $r \neq r_k$ that $f(r) = (r_k - r) \prod_{j=1, j\neq k}^{q} e_j \bmod e_k$. Since $l_r < l_e$, we have that $|r_k - r| < e_k$ and as the e_i are distinct primes, we finally get that $\gcd((r_k - r)\prod_{j=1, j\neq k}^{q} e_j, e_k) = 1$ and thus $f(r) \neq 0 \bmod e_k$ for $r \neq r_k$.

4.2 The SDH-Based Schemes

Under the SDH assumption, the situation is very similar. Here we also analyze three possible types of forgeries $(m^*, (r^*, s^*, t^*))$: 1.) $t^* \notin \{t_1, \ldots, t_q\}$, 2.) $t^* = t_i$ with $i \in [1;q]$ but $r^* \neq r_i$, and 3.) $t^* = t_i$, $r^* = r_i$ (but $m^* \neq m_i$) with $i \in [1;q]$. Again, we concentrate on the second case. At the beginning, $\mathcal{B}$ is given an SDH challenge $\left(\hat{g}_1, \hat{g}_1^x, \hat{g}_1^{(x^2)}, \ldots, \hat{g}_1^{(x^q)}, g_2, g_2^x\right)$. This time, $\mathcal{B}$ chooses $PK = g_2^x$. In the SDH setting, Equation (1) transfers to

$$e(s, PK g_2^t) = e(\hat{g}_1^{f(r,x)}, g_2) \Leftrightarrow s^{x+t} = \hat{g}_1^{f(r,x)}. \tag{3}$$

In contrast to the SRSA setting, f is now a polynomial with indeterminate x and maximal degree q. Again, $\mathcal{B}$ must keep $f(r,x)$ and the $\hat{g}_1^{(x^i)}$ secret from $\mathcal{A}$. We define

$$f(r,x) = \sum_{i=1}^{q} r_i \prod_{\substack{j=1\\ j\neq i}}^{q} (x + t_j) - r \sum_{i=1}^{q} \prod_{\substack{j=1\\ j\neq i}}^{q} (x + t_j),$$

for $r_1, \ldots, r_q \in \mathcal{R}$ and distinct $t_1, \ldots, t_q \in \mathbb{Z}_p$. Using the SDH challenge, $\mathcal{B}$ can easily compute $\hat{g}_1^{f(r,x)}$ since $f(r,x)$ has maximal degree q. Observe that it always holds that $(f(r,x) - (r_k - r)\prod_{j=1, j\neq k}^{q}(x + t_j))/(x + t_k) \in \mathbb{Z}$. If $r = r_k$, we surely have that $f(r,x)/(x + t_k) \in \mathbb{Z}$. If $r \neq r_k$, then long division gives us $D \in \mathbb{Z}$ with $D \neq 0$ and a new polynomial $\tilde{f}_{t_k}(r,x)$ with coefficients in $\mathbb{Z}$ such that $f(r,x) = \tilde{f}_{t_k}(r,x)(x + t_k) + D$. Similar to the SRSA class, we can find a solution to the SDH challenge from $\mathcal{A}$'s forgery as

$$\left((s^*)\hat{g}_1^{-\tilde{f}_{t^*}(r^*,x)}\right)^{1/D} = \hat{g}_1^{1/(x+t^*)}, t^*.$$

4.3 Security of the Chameleon Hash Signature Class

The chameleon hash class is also tightly secure in the SRSA and the SDH setting. For convenience let $c_i = ch(r_i, m_i)$ for $i \in [1; q]$ and $c^* = ch(r^*, m^*)$. Altogether there are again three types of forgeries to consider: 1) $e^* \notin \{e_1, \ldots, e_q\}$ ($t^* \notin \{t_1, \ldots, t_q\}$), 2) $e^* = e_i$ ($t^* = t_i$) but $c^* \neq c_i$, and 3) $e^* = e_i$ ($t^* = t_i$), $c^* = c_i$ but $m^* \neq m_i$. The proof of 1) is straight-forward and very similar to the proof of Type I forgers of the combining class. The proof of 3) clearly reduces to the security properties of the chameleon hash function. The proof of 2) requires our new technique to set up $f(c)$ ($f(c, x)$). Recall Section 4 where we analyzed the equations $s^e = \hat{u}^{f(c)}$ and $f(c) = \sum_{i=1}^{q} c_i \prod_{j=1, j\neq i}^{q} e_j - c \sum_{i=1}^{q} \prod_{j=1, j\neq i}^{q} e_j$ in the SRSA setting (and $s^{x+t} = \hat{g}_1^{f(c,x)}$ and $f(c, x) = \sum_{i=1}^{q} c_i \prod_{j=1, j\neq i}^{q} (x + t_j) - c \sum_{i=1}^{q} \prod_{j=1, j\neq i}^{q} (x + t_j)$ in the SDH setting).

In the proof of the combining class the c_i are random values ($c_i = r_i$) that can be specified prior to the simulation phase. In the proof of the chameleon hash class we take a similar approach. Now the c_i are the output values of a chameleon hash function. In the initialization phase of the proof we choose q random input pairs $(m'_i, r'_i) \in \mathcal{M} \times \mathcal{R}$, $i \in [1; q]$ to compute the $c_i = \mathsf{CHEval}(PK_{\mathcal{CH}}, m'_i, r'_i)$. Then we prepare the function $f(c)$ ($f(c, x)$) with $C = \{c_1, \ldots, c_q\}$ and a set of q random primes l_e-bit primes (random values $t_1, \ldots, t_q \in \mathbb{Z}_p$) as in the proofs of the combining class. Next, we embed $f(c)$ ($f(c, x)$) in the exponents of the two group elements u, v (a, b). In the simulation phase we give the simulator $SK_{\mathcal{CH}}$ to map the attacker's messages m_i to the prepared c_i by computing $r_i = \mathsf{CHColl}(SK_{\mathcal{CH}}, r'_i, m'_i, m_i)$. In this way we can successfully simulate the signing oracle. In the extraction phase, the properties of the chameleon hash function guarantee that $c^* \notin \{c_1, \ldots, c_q\}$ (otherwise we can break the security of the chameleon hash function). This ensures that we can find a solution to the SRSA challenge (SDH challenge).

4.4 Security Analysis of $\mathcal{S}_{\mathbf{CMB,SRSA}}$

Lemma 3. *In the SRSA setting, suppose the $(t_{SRSA}, \epsilon_{SRSA})$-SRSA assumption holds and $\mathcal{V}$ is a $(t_{comb}, \epsilon_{comb}, \delta_{comb})$-combining function. Then, the combining signature class as presented in Section 3.1 is (q, t, ϵ)-secure*[3] *against adaptive chosen message attacks provided that*

$$q = q_{SRSA}, \quad \epsilon \leq \frac{9}{2}\epsilon_{SRSA} + 3\epsilon_{comb} + 3q\delta_{comb} + \frac{3q^2}{|E|} + 9 \cdot 2^{2-l_n/2}, \quad t \approx t_{SRSA}.$$

The proof of Lemma 3 is the first step in the proof of Theorem 1. It implies that the original Camenisch-Lysyanskaya, the Fischlin and the Zhu's signature scheme are tightly secure against existential forgeries under adaptive chosen message attacks.

[3] Using explicit bounds on the prime counting function [16], we can lower bound the number of primes in E for $l_e \geq 7$ as $|E| > (2^{l_e} - 1)/(\ln(2^{l_e} - 1) + 2) - (2^{l_e - 1} - 1)/(\ln(2^{l_e - 1} - 1) - 4)$.

Proof. Assume that $\mathcal{A}$ is a forger that (q,t,ϵ)-breaks the strong existential unforgeability of $\mathcal{S}_{\mathrm{CMB,SRSA}}$. Then, we can construct a simulator $\mathcal{B}$ that, by interacting with $\mathcal{A}$, solves the SRSA problem in time t_{SRSA} with advantage ϵ_{SRSA}. We consider three types of forgers that after q queries $m_1,\ldots,m_q$ and corresponding responses $(r_1,s_1,e_1),\ldots,(r_q,s_q,e_q)$ partition the set of all possible forgeries $(m^*,(r^*,s^*,e^*))$. In the proof, we treat all types of attackers differently. At the beginning, we let $\mathcal{B}$ guess with probability at least $\frac{1}{3}$ which forgery $\mathcal{A}$ outputs. Lemma 3 then follows by a standard hybrid argument. We assume that $\mathcal{B}$ is given an SRSA challenge instance $(\hat{u},n)$. Let $\Pr[S_i]$ denote the success probability of an attacker to successfully forge signatures in Game i.

Type I Forger $(e^* \notin \{e_1,\ldots,e_q\})$
Game$_0$. This is the original attack game. By assumption, $\mathcal{A}$ (q,t,ϵ)-breaks $\mathcal{S}_{\mathrm{CMB,SRSA}}$ when interacting with the signing oracle $\mathcal{O}(SK,\cdot)$. We have that,

$$\Pr[S_0] = \epsilon \ . \tag{4}$$

Game$_1$. Now, $\mathcal{B}$ constructs the values u,v,w using the SRSA challenge instead of choosing them randomly from QR_n. First, $\mathcal{B}$ chooses q random primes $e_1,\ldots,e_q \xleftarrow{\$} E$ and three random elements $t_0',t_0'' \xleftarrow{\$} \mathbb{Z}_{(n-1)/4}$ and $t_0 \xleftarrow{\$} \mathbb{Z}_{3(n-1)/4}$. In the following let $\bar{e} := \prod_{k=1}^{q} e_k$, $\bar{e}_i := \prod_{k=1,k\neq i}^{q} e_k$ and $\bar{e}_{i,j} := \prod_{k=1,k\neq i,k\neq j}^{q} e_k$. The simulator computes $u = \hat{u}^{2t_0\bar{e}}$, $v = \hat{u}^{2t_0'\bar{e}}$, $w = \hat{u}^{2t_0''\bar{e}}$ using the SRSA challenge. Since the t_0,t_0',t_0'' are not chosen uniformly at random from $\mathbb{Z}_{p'q'}$ we must analyze the success probability for $\mathcal{A}$ to detect our construction. Observe that $(n-1)/4 = p'q' + (p'+q')/2 > p'q'$. Without loss of generality let $p' > q'$. Now, the probability of a randomly chosen $x \in \mathbb{Z}_{(n-1)/4}$ not to be in $\mathbb{Z}_{p'q'}$ is

$$\Pr[x \xleftarrow{\$} \mathbb{Z}_{(n-1)/4},\ x \notin \mathbb{Z}_{p'q'}] = 1 - \frac{|\mathbb{Z}_{p'q'}|}{|\mathbb{Z}_{(n-1)/4}|} < \frac{1}{q'+1} < 2^{-(|q'|_2-1)} \ .$$

With the same arguments we can show that t_0 is also distributed almost uniformly at random in $\mathbb{Z}_{p'q'}$ and $\mathbb{Z}_{3p'q'}$. Since the e_i are primes smaller than p' and q' it holds that $e_i \not| p'q'$. Therefore, the distribution of the generators is almost equal to the previous game and we get by a union bound that

$$\Pr[S_1] \geq \Pr[S_0] - 3 \cdot 2^{-(l_n/2-2)} \ . \tag{5}$$

Game$_2$. Now, $\mathcal{B}$ simulates $\mathcal{O}(SK,\cdot)$ by answering $\mathcal{A}$'s signature queries. Subsequently, set $z_j = z(e_j,m_j)$ and $z^* = z(e^*,m^*)$. The simulator $\mathcal{B}$ sets $PK = n$ and for all $j \in \{1,\ldots,q\}$ it chooses a random $r_j \in \mathcal{R}$ and outputs $\sigma_j = (r_j,s_j,e_j)$ with $s_j = (uv^{r_j}w^{z_j})^{\frac{1}{e_j}} = \hat{u}^{2(t_0+t_0'r_j+t_0''z_j)\bar{e}_j}$. The distribution of the so computed values is equal to the previous game and

$$\Pr[S_2] = \Pr[S_1] \ . \tag{6}$$

Game$_3$. Now, consider $\mathcal{A}$'s forgery $(m^*,(r^*,s^*,e^*))$. Define $\hat{e} = (t_0+t_0'r^*+t_0''z^*)$. For $\mathcal{A}$'s forgery it holds that $(s^*)^{e^*} = \hat{u}^{2\bar{e}\hat{e}}$. We also have that

$\gcd(e^*, 2\bar{e}\hat{e}) = \gcd(e^*, \hat{e})$ since by assumption we know $\gcd(e^*, 2\bar{e}) = 1$. We will now analyze the probability for the event $\gcd(e^*, \hat{e}) < e^*$ to happen. If $\gcd(e^*, \hat{e}) = e^*$ (or $\hat{e} = 0 \bmod e^*$) $\mathcal{B}$, simply aborts and restarts. Since $|e^*|_2 = l_e$, it holds that $\gcd(e^*, p'q') < e^*$. Write $t_0 \in \mathbb{Z}_{3(n-1)/4}$ as $t_0 = t_{0,1} + p'q't_{0,2}$ where $t_{0,2} \in [0; 2]$ and $t_{0,1} \in [0, p'q' - 1]$ and observe that $\mathcal{A}$'s view is independent from $t_{0,2}$. Let $T = \hat{e} - p'q't_{0,2}$. We now argue that there exists at most one $\tilde{t}_{0,2} \in [0; 2]$ such that $T + \tilde{t}_{0,2}p'q' = 0 \mod e^*$. This is crucial because if $\mathcal{A}$ produces forgeries with $T + \tilde{t}_{0,2}p'q' = 0 \mod e^*$ for *all* $\tilde{t}_{0,2} \in [0; 2]$ it always holds that $\gcd(e^*, \hat{e}) = e^*$ and $\mathcal{B}$ cannot extract a solution to the SRSA challenge (using the techniques described below). Assume there exists at least one such $\tilde{t}_{0,2}$. Then, we have that $T + \tilde{t}_{0,2}p'q' = 0 \mod e^*$. Let us analyze the remaining possibilities $\tilde{t}_{0,2} \pm 1$ and $\tilde{t}_{0,2} \pm 2$ as $A = T + \tilde{t}_{0,2}p'q' \pm p'q' \mod e^*$ and $B = T + \tilde{t}_{0,2}p'q' \pm 2p'q' \mod e^*$. Since $\gcd(e^*, p'q') < e^*$ we know that $p'q' \neq 0 \bmod e^*$. As $T + \tilde{t}_{0,2}p'q' = 0 \mod e^*$ we must have that $A \neq 0 \bmod e^*$. Also, because e^* is odd we know that $2p'q' \neq 0 \bmod e^*$ and thus $B \neq 0 \bmod e^*$. So, because there can only exist at most one $\tilde{t}_{0,2} \in [0; 2]$ with $\gcd(e^*, \hat{e}) = e^*$ and since this $\tilde{t}_{0,2}$ is hidden from $\mathcal{A}$'s view, $\mathcal{A}$'s probability to output it is at most $1/3$. This means that with probability at least $2/3$, $\mathcal{B}$ has that $\gcd(e^*, \hat{e}) = d < e^*$. Using $\mathcal{A}$'s forgery $(m^*, (r^*, s^*, e^*))$, $\mathcal{B}$ can break the SRSA assumption by computing $a, b \in \mathbb{Z}$ with $\gcd(e^*/d, 2\bar{e}\hat{e}/d) = ae^*/d + b2\bar{e}\hat{e}/d = 1$ and

$$\hat{u}^{d/e^*} = \hat{u}^a (s^*)^b, e^*/d.$$

Finally, we have that

$$\Pr[S_3] \geq 2 \cdot \Pr[S_2]/3 \tag{7}$$

and

$$\Pr[S_3] = \epsilon_{\text{SRSA}} \ . \tag{8}$$

Plugging in Equations (4)–(8), we get that $\epsilon \leq \frac{3}{2}\epsilon_{\text{SRSA}} + 3 \cdot 2^{2-l_n/2}$.

Type II Forger ($e^* = e_i$ and $r^* \neq r_i$)
We only present the differences to the previous proof.

Game$_1$. First, $\mathcal{B}$ randomly chooses q *distinct* l_e-bit primes $e_1, \ldots, e_q$ and q random elements $r_1, \ldots, r_q \in \mathcal{R}$. Additionally, it chooses three random elements t_0, t_0', t_0'' from $\mathbb{Z}_{(n-1)/4}$. Next, $\mathcal{B}$ computes $u = \hat{u}^{2\left(t_0\bar{e} + \sum_{i=1}^{q} r_i\bar{e}_i\right)}$, $v = \hat{u}^{2\left(t_0'\bar{e} - \sum_{i=1}^{q} \bar{e}_i\right)}$, and $w = \hat{u}^{2t_0''\bar{e}}$ using the SRSA challenge. Again,

$$\Pr[S_1] \geq \Pr[S_0] - 3 \cdot 2^{-(l_n/2-2)} \ . \tag{9}$$

Game$_2$. Now $\mathcal{B}$ simulates the signing oracle $\mathcal{O}(SK, \cdot)$. On each signature query m_j with $j \in \{1, \ldots, q\}$, $\mathcal{B}$ responds with $\sigma_j = (r_j, s_j, e_j)$ using the precomputed r_j and e_j and computing s_j as

$$s_j = \hat{u}^{2\left((t_0 + t_0'r_j + t_0''z_j)\bar{e}_j + \sum_{i=1, i\neq j}^{q} (r_i - r_j)\bar{e}_{i,j}\right)} \ .$$

Since we have chosen the e_i to be distinct primes we have by a union bound that

$$\Pr[S_2] \geq \Pr[S_1] - \frac{q^2}{|E|} \ . \tag{10}$$

Game₃. Now consider $\mathcal{A}$'s forgery $(m^*, (r^*, s^*, e^*))$. By assumption there is an $i \in \{1, \ldots, q\}$ with $e^* = e_i$ and $r_i \neq r^*$. Then we have that

$$\left((s^*) \cdot \hat{u}^{-2\left((t_0+t_0'r^*+t_0''z^*)\bar{e}_i+\sum_{j=1,j\neq i}^{q}(r_j-r^*)\bar{e}_{i,j}\right)}\right)^{e_i} = \hat{u}^{2(r_i-r^*)\bar{e}_i} \ .$$

Since $|r_i - r^*| < e_i$ and e_i is an odd prime, we get $\gcd(2(r_i - r^*), e_i) = 1$ and as before we can compute $\hat{u}^{\frac{1}{e_i}}$ which is a solution to the SRSA challenge.

$$\Pr[S_3] = \epsilon_{\mathrm{SRSA}} \ . \tag{11}$$

Summing up Equations (9)–(11), we get $\epsilon \leq \epsilon_{\mathrm{SRSA}} + q^2/|E| + 3 \cdot 2^{2-l_n/2}$.

Type III Forger $(e^* = e_i$ and $r^* = r_i)$
There are only minor differences as compared to the previous proof.

Game₁. First, $\mathcal{B}$ randomly chooses q l_e-bit primes $e_1, \ldots, e_q$ and q random $z_1, \ldots, z_q \in \mathcal{Z}$, Then, $\mathcal{B}$ draws three random elements t_0, t_0', t_0'' from $\mathbb{Z}_{(n-1)/4}$. Next, $\mathcal{B}$ computes u, v, and w as $u = \hat{u}^{2\left(t_0\bar{e}+\sum_{i=1}^{q} z_i\bar{e}_i\right)}$, $v = \hat{u}^{2t_0'\bar{e}}$, and $w = \hat{u}^{2\left(t_0''\bar{e}-\sum_{i=1}^{q}\bar{e}_i\right)}$.

$$\Pr[S_1] \geq \Pr[S_0] - 3 \cdot 2^{-(l_n/2-2)} \ . \tag{12}$$

Game₂. This game is equal to the previous game except that we require the e_i to be all distinct. We have that

$$\Pr[S_2] \geq \Pr[S_1] - \frac{q^2}{|E|} \ . \tag{13}$$

Game₃. Now $\mathcal{B}$ simulates the signing oracle. For each queries m_j with $j \in \{1, \ldots, q\}$, $\mathcal{B}$ computes $r_j = z^{-1}(z_j, m_j)$. If $r_j \notin \mathcal{R}$, $\mathcal{B}$ aborts. Otherwise it outputs the signature $\sigma_j = (r_j, s_j, e_j)$ with s_j being computed as

$$s_j = \left(uv^{r_j}w^{z_j}\right)^{\frac{1}{e_j}} = \hat{u}^{2\left((t_0+t_0'r_j+t_0''z_j)\bar{e}_j+\sum_{i=1,i\neq j}^{q}(z_i-z_j)\bar{e}_{i,j}\right)} \ .$$

The properties of the combining function guarantee that the r_j are statistically close to uniform over $\mathcal{R}$ such that,

$$\Pr[S_3] \geq \Pr[S_2] - q\delta_{\mathrm{comb}} \ . \tag{14}$$

Game₄. This game is like the previous one except that $\mathcal{B}$ aborts whenever there is a collision such that $z_i = z(r_i, m_i) = z(r_i, m^*) = z^*$ for some r_i. Observe that we must have $m^* \neq m_i$, otherwise $\mathcal{A}$ just replayed the i-the message/signature pair. For all t_{comb}-time attackers this happens with probability at most ϵ_{comb}. Therefore,

$$\Pr[S_4] \geq \Pr[S_3] - \epsilon_{\mathrm{comb}} \ . \tag{15}$$

Consider $\mathcal{A}$'s forgery $(m^*, (r^*, s^*, e^*))$. By assumption, there is one index $i \in \{1, \ldots, q\}$ with $e^* = e_i$ and $r^* = r_i$. For this index it holds that

$$\left((s^*) \cdot \hat{u}^{-2\left((t_0+t_0'r^*+t_0''z^*)\bar{e}_i+\sum_{j=1,j\neq i}^{q}(z_j-z^*)\bar{e}_{i,j}\right)}\right)^{e_i} = \hat{u}^{2(z_i-z^*)\bar{e}_i} \ .$$

Since we have excluded collisions, it follows that $z_i \neq z^*$. As $|z_i - z^*| \leq e_i$, $\mathcal{B}$ can compute $\hat{u}^{\frac{1}{e_i}}$ as a solution to the SRSA challenge. Finally,

$$\Pr[S_4] = \epsilon_{\mathrm{SRSA}} \ . \tag{16}$$

Equations (12)–(16) show $\epsilon \leq \epsilon_{\mathrm{SRSA}} + \epsilon_{\mathrm{comb}} + q\delta_{\mathrm{comb}} + q^2/|E| + 3 \cdot 2^{2-l_n/2}$.

Acknowledgement. I would like to thank Mathias Herrmann, Tibor Jager, Eike Kiltz, and Maike Ritzenhofen for useful comments on earlier drafts of this paper and the anonymous referees of EUROCRYPT'11 for helpful comments and suggestions.

References

1. Au, M.H., Susilo, W., Mu, Y.: Constant-size dynamic k-TAA. In: Prisco, R.D., Yung, M. (eds.) SCN 2006. LNCS, vol. 4116, pp. 111–125. Springer, Heidelberg (2006)
2. Bernstein, D.J.: Proving tight security for rabin-williams signatures. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 70–87. Springer, Heidelberg (2008)
3. Boneh, D., Boyen, X.: Short signatures without random oracles and the SDH assumption in bilinear groups. J. Cryptology 21(2), 149–177 (2008)
4. Camenisch, J., Lysyanskaya, A.: A signature scheme with efficient protocols. In: Cimato, S., Galdi, C., Persiano, G. (eds.) SCN 2002. LNCS, vol. 2576, pp. 268–289. Springer, Heidelberg (2003)
5. Camenisch, J., Lysyanskaya, A.: Signature schemes and anonymous credentials from bilinear maps. In: Franklin, M.K. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 56–72. Springer, Heidelberg (2004)
6. Chevallier-Mames, B., Joye, M.: A practical and tightly secure signature scheme without hash function. In: Abe, M. (ed.) CT-RSA 2007. LNCS, vol. 4377, pp. 339–356. Springer, Heidelberg (2006)
7. Coron, J.S., Naccache, D.: Security analysis of the gennaro-halevi-rabin signature scheme. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 91–101. Springer, Heidelberg (2000)
8. Cramer, R., Shoup, V.: Signature schemes based on the Strong RSA assumption. ACM Trans. Inf. Syst. Secur. 3(3), 161–185 (2000)
9. Fischlin, M.: The cramer-shoup strong-RSA Signature scheme revisited. In: Desmedt, Y. (ed.) PKC 2003. LNCS, vol. 2567, pp. 116–129. Springer, Heidelberg (2002)
10. Gennaro, R., Halevi, S., Rabin, T.: Secure hash-and-sign signatures without the random oracle. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 123–139. Springer, Heidelberg (1999)

11. Goldwasser, S., Micali, S., Rivest, R.L.: A digital signature scheme secure against adaptive chosen-message attacks. SIAM J. Comput. 17(2), 281–308 (1988)
12. Hofheinz, D., Kiltz, E.: Programmable hash functions and their applications. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 21–38. Springer, Heidelberg (2008)
13. Krawczyk, H., Rabin, T.: Chameleon signatures. In: NDSS. The Internet Society, San Diego (2000)
14. Lysyanskaya, A., Rivest, R.L., Sahai, A., Wolf, S.: Pseudonym systems. In: Heys, H.M., Adams, C.M. (eds.) SAC 1999. LNCS, vol. 1758, pp. 184–199. Springer, Heidelberg (2000)
15. Okamoto, T.: Efficient blind and partially blind signatures without random oracles. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 80–99. Springer, Heidelberg (2006)
16. Rosser, B.: Explicit bounds for some functions of prime numbers. American Journal of Mathematics 63(1), 211–232 (1941)
17. Shamir, A., Tauman, Y.: Improved online/offline signature schemes. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 355–367. Springer, Heidelberg (2001)
18. Zhu, H.: New digital signature scheme attaining immunity to adaptive-chosen message attack. Chinese Journal of Electronics 10(4), 484–486 (2001)
19. Zhu, H.: A formal proof of Zhu's signature scheme. Cryptology ePrint Archive, Report 2003/155 (2003), `http://eprint.iacr.org/`