

# Security of Verifiably Encrypted Signatures and a Construction Without Random Oracles

Markus Rückert\* and Dominique Schröder\*\*

TU Darmstadt, Germany

rueckert@cdc.informatik.tu-darmstadt.de      schroeder@me.com

**Abstract.** In a verifiably encrypted signature scheme, signers encrypt their signature under the public key of a trusted third party and prove that they did so correctly. The security properties, due to Boneh et al. (Eurocrypt 2003), are unforgeability and opacity.

This paper proposes two novel fundamental requirements for verifiably encrypted signatures, called *extractability* and *abuse-freeness*, and analyzes its effects on the established security model. Extractability ensures that the trusted third party is always able to extract a valid signature from a valid verifiably encrypted signature and abuse-freeness guarantees that a malicious signer, who cooperates with the trusted party, is not able to forge a verifiably encrypted signature. We further show that both properties are not covered by the model of Boneh et al. The second main contribution of this paper is a verifiably encrypted signature scheme, provably secure without random oracles, that is more efficient and greatly improves the public key size of the only other construction in the standard model by Lu et al. (Eurocrypt 2006). Moreover, we present strengthened definitions for unforgeability and opacity in the spirit of strong unforgeability of digital signature schemes.

## 1 Introduction

The concept of verifiably encrypted signature (VES) schemes was proposed by Boneh, Gentry, Lynn, and Shacham [5]. There, a signer encrypts its signature under the public key a trusted third party, called the adjudicator, and then attaches a proof about its content. The purpose of this proof is that verification will then confirm that the signer has truly signed a certain object. The necessity for such verification can be exemplified by a popular application, namely online contract signing, which is a type of optimistic fair exchange protocol [1,4,8].

Suppose Alice and Bob wish to sign the same contract. Both want to be sure that the other party will also produce a signature before revealing their own. Following the protocol, Alice and Bob exchange verifiably encrypted signatures. After they ascertained the correctness of the encrypted signature, they reveal the corresponding ordinary signature. If, for example, Alice is not willing to disclose

---

\* This work was supported by CASED ([www.cased.de](http://www.cased.de)).

\*\* Dominique Schröder was supported by the Emmy Noether Program Fi 940/2-1 of the German Research Foundation (DFG).

Scheme	ROM	Strongly secure	Key size ( $sk/pk$ )	Signature size	VES creation	Vf
BGLS	Yes	No	160 / 160 bits	320 bits	2 E + 1 M	3 P
LOSSW	No	No	160 bits / 10 KB (*)	480 bits	4 E + ( $\text{ham}(m) + 3$ ) M	3 P
Section 6	No	No	320 / 640 bits	480 bits	1 I + 3 E + 1 M	2 P
Full version [14]	No	Yes	640 / 960 bits	800 bits	2 I + 4 E + 1 M	3 P

**Table 1.** Comparison between the different verifiably encrypted signature schemes. The column “ROM” states whether security is proven in the random oracle model. The column “Strongly Secure” determines whether the scheme is secure in our stronger model. Let  $\text{ham}(m)$  be the hamming weight of a bit string  $m$ , I an inversion, M a multiplication, and E an exponentiation. Let P be the cost for a pairing evaluation. Since pairings dominate the computational costs, other operations were omitted in the “Verification” column. We instantiate the schemes using Barreto-Naehrig curves [6] with a 160-bit point representation. (\*) This value is taken from [13]. They need approximately 160 group elements.

her signature, then Bob can take her verifiably encrypted signature together with the transcript to the adjudicator, who uncovers Alice’s ordinary signature. This fail-safe mechanism prevents Alice from misusing this one-sided commitment to a contract for purposes such as: legal actions, blackmail, or simply negotiating a better deal elsewhere.

The security of verifiably encrypted signatures is defined via unforgeability and opacity [5]. Roughly speaking, unforgeability ensures that a malicious user cannot produce signatures on behalf of another party. Opacity guarantees that only the adjudicator and the signer can disclose an ordinary signature from a verifiably encrypted signature.

Boneh et al. illustrated their concept with a first construction (BGLS), provably secure in the random oracle model [5]. Later, Zhang et al. presented a more efficient solution (ZSNS) [16], also in the random oracle model. As the uninstanciability result of Canetti, Goldreich, and Halevi [7] disputes the soundness of the random oracle methodology, it has inspired many researchers to find secure and efficient schemes outside the random oracle model. To the best of our knowledge, Lu et al. [13] presented the first verifiably encrypted signature scheme (LOSSW), which is secure in the standard model, at Eurocrypt 2006. Their scheme is based on the Waters signature scheme [15] and its major drawback is that they need a large public key (approximately 160 group elements).

*Our Contribution.* Surprisingly, the original security model for verifiably encrypted signature schemes does *not* guarantee that the adjudicator is always able to extract a valid signature from a valid verifiably encrypted signature. Considering again a protocol based on VES for optimistic fair exchange. We show that every VES can easily be turned into a scheme which remains secure,

but where a malicious signer can output a verifiably encrypted signature such that the ordinary signature is hidden irrecoverably. This implies that a VES that does not support extractability is not suitable for such protocols. Thus, as our first result, we extend the model of [5] to ensure *extractability*. Subsequently, we study the effect of extractability on Boneh et al.’s security model. Though no explicit proof of extractability exists for previous constructions, they already support the property due to a close similarity of the signature verification algorithm and the verification algorithm for encrypted signatures. Extractability of the BGLS and of the LOSSW scheme can be proven analogously to the proof of Theorem 6.

Furthermore, we propose a definition of *abuse-freeness* in the VES context. Basically, an abuse-free VES guarantees that an adversary who colludes with the adjudicator is not able to derive a verifiably encrypted signature on behalf of another signer. We show that for a “natural” class of VES schemes, abuse-freeness is already implied. Since the instantiation of [5] and [13] fall into this class, our results give more confidence about the security of their schemes.

As a round-up of the model discussion, we introduce strengthened definitions for unforgeability and opacity, namely *strong unforgeability* and *strong opacity*, which is closely related to the need for strong unforgeability in digital signature schemes. It prevents eavesdroppers from replaying the fair exchange protocol with re-randomized verifiably encrypted signatures.

Note that neither BGLS nor LOSSW satisfy these stronger notions, as one can re-randomize the encrypted signature. We show how a slight modification of our construction in Section 6 yields a scheme that is provably secure in the stronger model.

As our second result, we present a new verifiably encrypted signature scheme based on the Boneh-Boyen signature [3]. This scheme greatly improves the key size and efficiency of LOSSW, while achieving the same signature size. Table 1 compares our work with previous schemes. Note that the construction in Section 6 involves a public key size of four group elements (in comparison to the LOSSW instantiation, our scheme reduces the key size by a factor of 40), and only needs two pairing computations for the verification (rather than three in LOSSW).

Another extension to the security model is to give the adversary access to the adjudication oracle for different users as proposed by Hess [12]. Here, however, we follow the model of Boneh et al. concerning only the two user setting.

*Organization.* We start out by introducing our notation and some basic definitions in Section 2. In Section 3, we recall the model for verifiably encrypted signatures, along with the corresponding security definitions. Subsequently, in Section 4 and Section 5, we introduce *extractability* and *abuse-freeness* thereby extending the model due to Boneh et al., and we argue why these notions are necessary. The stronger security model, with strong unforgeability and strong opacity is described in Appendix C. Our verifiably encrypted signature scheme is presented in Section 6, along with the security proofs. We show in the full version of this paper that a modified version of the construction satisfies the stronger model [14].

## 2 Notation and Basic Definitions

*Bilinear Maps.* Let  $(\mathbb{G}_1, *)$ ,  $(\mathbb{G}_2, *)$ , and  $(\mathbb{G}_T, *)$  denote three groups of prime order  $p$  with the following properties: all group actions can be computed efficiently;  $g_1$  is a generator of  $\mathbb{G}_1$  and  $g_2$  is a generator of  $\mathbb{G}_2$ ;  $\psi$  is a group homomorphism from  $\mathbb{G}_2$  to  $\mathbb{G}_1$ , with  $\psi(g_2) = g_1$ ;  $\mathbf{e} : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$  is an efficiently computable map.  $\mathbf{e}$  is bilinear and non-degenerate, i.e.  $\forall u \in \mathbb{G}_1 \forall v \in \mathbb{G}_2 \forall a, b \in \mathbb{Z}$ :  $\mathbf{e}(u^a, v^b) = \mathbf{e}(u, v)^{ab}$  and  $z = \mathbf{e}(g_1, g_2) \neq 1$  generates  $\mathbb{G}_T$ .

We assume that  $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, g_1, g_2, \mathbf{e}$ , and  $z$  are fixed and public parameters. By  $a_1 \| \dots \| a_\ell$  we denote the encoding of  $a_1, \dots, a_\ell$  such that  $a_1, \dots, a_\ell$  are uniquely recoverable. With  $x \stackrel{\$}{\leftarrow} X$  we denote choosing  $x$  uniformly at random from the finite set  $X$ .  $\{x_i\}_1^q$  is the set of  $x_1, \dots, x_q$ . Furthermore,  $n$  always denotes the security parameter.

*Secure Signature Schemes.* Security of signature schemes  $\text{DSig} = (\text{Kg}, \text{Sign}, \text{Vf})$  is proven against existential forgery under chosen message attacks (EU-CMA) [10]. In this model, an adversary adaptively invokes a signing oracle and is successful if it outputs a signature on a *new* message. A stronger notion is strong unforgeability under chosen message attacks (SU-CMA), where it is sufficient for an adversary to output a new message-signature pair.

*Boneh-Boyen Signature Scheme.* We recall the strongly unforgeable Boneh-Boyen ( $\mathcal{BB}$ ) signature scheme. *Key Generation:*  $\text{Kg}(1^n)$  selects  $x, y \stackrel{\$}{\leftarrow} \mathbb{Z}_p^*$ , computes  $u \leftarrow g_2^x$  and  $v \leftarrow g_2^y$ . The public key is  $\text{spk} \leftarrow (u, v)$  and the private key is  $\text{ssk} \leftarrow (x, y)$ ; *Signing:*  $\text{Sign}(\text{ssk}, m)$  takes as input the secret key  $(x, y)$  as well as a message  $m \in \mathbb{Z}_p$ . It picks  $r \stackrel{\$}{\leftarrow} \mathbb{Z}_p \setminus \{-\frac{x+m}{y}\}$  and computes  $\sigma \leftarrow g_1^{1/(x+m+yr)}$ , where  $1/(x+m+yr)$  is computed modulo  $p$ . The output is  $(r, \sigma)$ ; *Signature Verification:*  $\text{Vf}(\text{spk}, (r, \sigma), m)$  returns 1 iff  $\mathbf{e}(\sigma, u g_2^m v^r) = z$ , otherwise returns 0.

## 3 Verifiably Encrypted Signatures

According to [5] verifiably encrypted signature schemes are defined as  $\text{VES} = (\text{Kg}, \text{AdjKg}, \text{Sign}, \text{Vf}, \text{Create}, \text{VesVf}, \text{Adj})$  with the following specification and security model:

*Key Generation:*  $\text{Kg}(1^n)$  outputs a private signing key  $sk$  and a public verification key  $pk$ ; *Signing*  $\text{Sign}(sk, m)$  outputs a signature  $\sigma$  under  $sk$  on a message  $m$  chosen from the message space  $\mathcal{M}$ ; *Verification:*  $\text{Vf}(pk, \sigma, m)$  outputs 1 iff  $\sigma$  is a valid signature on  $m$  under  $pk$ ; *Adjudicator Key Generation:*  $\text{AdjKg}(1^n)$  outputs a key pair  $(ask, apk)$ , where  $ask$  is the private key and  $apk$  the corresponding public key of the adjudicator; *VES Creation:*  $\text{Create}(sk, apk, m)$  receives a secret key  $sk$ , the adjudicator's public key  $apk$ , and a message  $m \in \mathcal{M}$ . It returns a verifiably encrypted signature  $\omega$  on  $m$ ; *VES Verification:*  $\text{VesVf}(apk, pk, \omega, m)$  gets the adjudicator's public key  $apk$ , a public key  $pk$ , a verifiably encrypted signature  $\omega$ , and a message  $m$ . It returns a bit, indicating the validity of  $\omega$ ; *Adjudication:*  $\text{Adj}(ask, apk, pk, \omega, m)$  accepts as input the key pair  $(ask, apk)$  of the adjudicator,

the public key  $pk$  of a signer, a verifiably encrypted signature  $\omega$ , and a message  $m$ . If  $\omega$  is valid, it extracts an ordinary signature<sup>1</sup>  $\sigma$  on  $m$  and returns  $\sigma$ .

A scheme VES is *complete*<sup>2</sup> if for all adjudication key pairs  $(ask, apk) \leftarrow \text{AdjKg}(1^n)$  and for all signature key pairs  $(sk, pk) \leftarrow \text{Kg}(1^n)$  the following holds:  $\text{VesVf}(apk, pk, \text{Create}(sk, apk, m), m) = 1$  and  $\text{Vf}(pk, \text{Adj}(ask, apk, pk, \text{Create}(sk, apk, m)), m) = 1$  for all  $m \in \mathcal{M}$ .

*Security Model.* The security of verifiably encrypted signatures is defined by unforgeability and opacity [5]. *Unforgeability* requires that it is hard to forge a verifiably encrypted signature and *opacity* implies that it is hard to extract ordinary signatures.

Both intuitions are formalized in experiments, where the adversary  $\mathcal{A}$  is given the public keys of the signer and the adjudicator. Moreover,  $\mathcal{A}$  has access to two oracles: a verifiably-encrypted-signature creation oracle  $\mathsf{C}$  that, upon input of a message  $m$ , returns a corresponding verifiably encrypted signature  $\omega$ ; and an adjudication oracle  $\mathsf{A}$  that extracts and returns a signature  $\sigma$  when queried with a message/verifiably encrypted signature pair  $(m, \omega)$ .

**Definition 1.** *A scheme VES is secure if the following holds:*

**Unforgeability:** *For any efficient algorithm  $\mathcal{A}$ , the probability that the following experiment evaluates to 1 is negligible (as a function of  $n$ ).*

**Experiment**  $\text{VesForge}_{\mathcal{A}}^{\text{VES}}(n)$   
 $(ask, apk) \leftarrow \text{AdjKg}(1^n)$   
 $(sk, pk) \leftarrow \text{Kg}(1^n)$   
 $(m^*, \omega^*) \leftarrow \mathcal{A}^{\mathsf{C}(sk, apk, \cdot), \mathsf{A}(ask, apk, pk, \cdot, \cdot)}(pk, apk)$   
 Return 1 iff  $\text{VesVf}(apk, pk, \omega^*, m^*) = 1$  and  
 $\mathcal{A}$  has never queried  $\mathsf{C}(sk, apk, \cdot)$  or  $\mathsf{A}(ask, apk, pk, \cdot, \cdot)$  about  $m^*$ .

**Opacity:** *For any efficient algorithm  $\mathcal{A}$ , the probability that the following experiment evaluates to 1 is negligible (as a function of  $n$ ).*

**Experiment**  $\text{Opac}_{\mathcal{A}}^{\text{VES}}(n)$   
 $(ask, apk) \leftarrow \text{AdjKg}(1^n)$   
 $(sk, pk) \leftarrow \text{Kg}(1^n)$   
 $(m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{C}(sk, apk, \cdot), \mathsf{A}(ask, apk, pk, \cdot, \cdot)}(pk, apk)$   
 Return 1 iff  $\text{Vf}(pk, \sigma^*, m^*) = 1$  and  
 $\mathcal{A}$  has never queried  $\mathsf{A}(ask, apk, pk, \cdot, \cdot)$  about  $m^*$ .

A scheme is called  $(t, q_{\mathsf{C}}, q_{\mathsf{A}}, \epsilon)$ -unforgeable (-opaque), if no adversary, running in time at most  $t$ , making at most  $q_{\mathsf{C}}$  verifiably-encrypted-signature oracle queries  $\mathsf{C}$ , and at most  $q_{\mathsf{A}}$  queries to the adjudication oracle  $\mathsf{A}$ , can succeed with probability at least  $\epsilon$  in the  $\text{VesForge}$  (respectively  $\text{Opac}$ ) experiment.

<sup>1</sup> Not necessarily the same signature, cf. [13].

<sup>2</sup> Note that in [5] this condition is called validity.

*Simplification.* As a first modification of this security model, we state and prove that it is possible to remove a redundant restriction from the definition of unforgeability. One might think that an adversary can succeed by modifying some “ciphertext”  $\omega$  such that the adjudicator extracts a fresh signature that can be encrypted once again to obtain a fresh  $\omega^*$ . We prove that the constraint that the adversary is not allowed to output a forgery for a message  $m^*$  already queried to  $A$ , without having queried  $m^*$  to  $C$  before, is unnecessary. In other words,  $A$  does not help to forge verifiably encrypted signatures.

Let  $\text{VesForge}'$  be the unforgeability experiment, in which an adversary is allowed to query everything to  $A$ , even its final output  $m^*$ . The idea is that a forger which is able to invoke the oracle  $A$  with a fresh tuple  $(m, \omega)$ , i.e. without having queried  $m$  to  $C$  beforehand, can already be used to break unforgeability.

**Theorem 1.** *VES is unforgeable w.r.t. to  $\text{VesForge}'$  if and only if it is unforgeable w.r.t. to  $\text{VesForge}$ .*

*Proof.* The first step is to prove that an adversary which breaks unforgeability in  $\text{VesForge}$  can be used to break unforgeability in  $\text{VesForge}'$ . Since this direction follows easily, we omit it. In the second part of the proof, consider an adversary  $\mathcal{A}$  that succeeds in the unforgeability experiment  $\text{VesForge}'$  with noticeable probability  $\epsilon(n)$ . We then construct an algorithm  $\mathcal{B}$  against unforgeability in  $\text{VesForge}$ , which runs  $\mathcal{A}$  as a black-box. Algorithm  $\mathcal{B}$  answers all oracle queries with its own oracles, i.e. it relays the entire communication between  $\mathcal{A}$  and the oracles. Whenever  $\mathcal{A}$  invokes the adjudication oracle  $A$  on a “fresh” and valid pair  $(m^*, \omega^*)$  (i.e., the adversary has not queried  $m^*$  to  $C$  before), then  $\mathcal{B}$  stops, outputting this pair as its forgery. Otherwise, if  $\mathcal{A}$  never performs such queries,  $\mathcal{B}$  forwards the final output of  $\mathcal{A}$ .

For the analysis, observe that  $\mathcal{A}$  may query the adjudication oracle on a “fresh” and valid pair  $(m^*, \omega^*)$ . On the one hand, the adversary  $\mathcal{A}$  would still succeed in experiment  $\text{VesForge}'$ , outputting a verifiably encrypted signature  $\omega^*$  on  $m^*$ , but on the other hand,  $\mathcal{A}$  cannot succeed in  $\text{VesForge}$  as  $\text{VesForge}$  does not allow  $\mathcal{A}$  to query the adjudication oracle about the final output of  $\mathcal{A}$ . But if  $\mathcal{A}$  is in position to perform a query consisting of a “fresh” and valid pair  $(m^*, \omega^*)$ , then  $\mathcal{B}$  directly outputs this tuple as its successful forgery. This tuple is a valid forgery, because  $\mathcal{B}$  never actually queried  $A$  about  $(m^*, \omega^*)$ . Additionally, note that  $\mathcal{B}$  is efficient as  $\mathcal{A}$  runs in polynomial time and  $\mathcal{B}$  can handle all queries efficiently.  $\square$

The following two sections justify the need for two additional security requirements, namely extractability and abuse-freeness.

## 4 The Need for Extractability

In the following, we formalize what should be a fundamental requirement for verifiably encrypted signatures, namely *extractability*. This property entails that if a verifiably encrypted signature  $\omega$  is valid, then the adjudicator is able to extract a valid signature  $\sigma$  with overwhelming probability.

**Definition 2 (Extractability).** A verifiably encrypted signature scheme VES is extractable if for any efficient algorithm  $\mathcal{A}$ , the probability that the following experiment evaluates to 1 is negligible (as a function of  $n$ ).

**Experiment**  $\text{Extract}_{\mathcal{A}}^{\text{VES}}(n)$   
 $(ask, apk) \leftarrow \text{AdjKg}(1^n)$   
 $(m^*, \omega^*, pk^*) \leftarrow \mathcal{A}^{\text{A}(ask, apk, \cdot, \cdot)}(apk)$   
Let  $\sigma^* \leftarrow \text{Adj}(ask, apk, pk^*, \omega^*, m^*)$   
Return 1 iff  $\text{VesVf}(apk, pk^*, \omega^*, m^*) = 1$  and  $\text{Vf}(pk^*, \sigma^*, m^*) = 0$ .

Observe that, in this case, the adjudication oracle  $\mathbf{A}$  takes as input the adjudicator key pair  $(ask, apk)$ , to which  $\mathcal{A}$  attaches tuples  $(pk^*, \omega^*, m^*)$  which consist of: a public key  $pk^*$ , a verifiably encrypted signature  $\omega^*$ , and a message  $m^*$ . Thus, extractability as defined above must hold for all pairs  $(m^*, \omega^*)$ , even for those not properly generated (i.e.  $\omega^*$  was not created for  $m^*$ ) and even in case  $pk^*$  is not chosen honestly. Note that  $pk^*$  serves as  $\mathcal{A}$ 's public key and that  $\mathcal{A}$  may not have a corresponding secret key  $sk^*$ .

If we do not allow the adversary to choose its public key dishonestly, we may still consider a model similar to the one above and we call the corresponding property *weak-extractability*. Note that a scheme that satisfies weak-extractability can always be turned into an extractable scheme by having the signer prove the correct form of its public key to the (universally trusted) adjudicator. This could be done, for example, by letting the signer hand its private key over to the trusted third party, using rewinding techniques, or using NIZKs such as in [11]. The adjudicator may then sign the public key or otherwise vouch for its validity. We motivate the need for extractability, showing that every verifiably encrypted signature scheme, that is secure in the model of [5], can easily be turned into one which is not extractable.

**Theorem 2.** *If there exists a secure scheme VES in the sense of [5], then there exists a scheme VES' which is secure but not extractable.*

The basic idea is that the verifiably encrypted signature may consist of two *independent* parts. The first part is used in the  $\text{VesVf}$  verification process and the second part is an encryption of the signature. As the parts are independent, a malicious signer can easily set the second part to an empty string, while computing the first part honestly.

*Proof.* We assume that the bit length of a verifiably encrypted signature is  $\text{out}(n)$ . VES' is defined as follows: *Key Generation, Signing, Verification:* Same as in VES; *VES Creation:* Given a message  $m \in \mathcal{M}$ , a signing key  $sk$ , and the public key of the adjudicator  $apk$ ,  $\text{Create}'$  computes  $\omega' \leftarrow \text{Create}(sk, apk, m)$  and outputs  $(\omega_1 \parallel \omega_2) \leftarrow (\omega' \parallel \omega') \in \{0, 1\}^{2 \cdot \text{out}(n)}$ ; *VES Verification:* Given a verifiably encrypted signature  $\omega_1 \parallel \omega_2$  on  $m$ , algorithm  $\text{VesVf}'$  outputs 1 iff  $\text{VesVf}(apk, pk, \omega_1, m)$  evaluates to 1 and 0 otherwise; *Adjudication:*  $\text{Adj}'(ask, apk, pk, \omega_1 \parallel \omega_2, m)$  outputs  $\sigma \leftarrow \text{Adj}(ask, apk, pk, \omega_2, m)$ .

Obviously, if VES is complete, unforgeable, and opaque, so is VES'. However, now the following adversary  $\mathcal{A}$  contradicts extractability, and even weak-extractability.

**Setup:**  $\mathcal{A}$  receives the adjudicator’s public key  $apk$  and honestly generates  $(sk, pk) \leftarrow \text{Kg}(1^n)$ .

**VES Creation:** When  $\mathcal{A}$  signs a message  $m$ , it calls  $(\omega_1 \parallel \omega_2) \leftarrow \text{Create}'(sk, apk, m)$  and outputs  $(m^*, \omega^*, pk^*) \leftarrow (m, \omega_1 \parallel 0^{\text{out}(n)}, pk)$ .

Since  $\omega_1$  remains unchanged in  $\text{Create}'$ ,  $\text{VesVf}'$  always returns 1. The algorithm  $\text{Adj}'$ , however, cannot extract a valid (ordinary) signature out of the second part because it is  $0^{\text{out}(n)}$ . Thus,  $\mathcal{A}$  breaks extractability with probability 1. Observe that the adjudicator  $\text{Adj}'$  does not fail because  $0^{\text{out}(n)}$  is “some” special string, but simply because  $0^{\text{out}(n)} \neq \omega_1$ .  $\square$

*Relation to the Security Model.* In the following, we show a helpful implication that facilitates security proofs in our extended model that entails unforgeability, opaqueness, and extractability. We mainly rely on the verifiably encrypted signature schemes having a common property, which we call *key-independence*. This property states that computing the encrypted signature can be performed, independently, by the following algorithms: one that computes the signature  $\sigma$  as in  $\text{DSig}$ , and a second algorithm that computes  $\omega$ , the verifiable encryption of  $\sigma$ . In other words, one can use an oracle  $\text{Sign}(ssk, \cdot)$  and transform its output into a verifiably encrypted signature independently of  $ssk$ .

**Definition 3 (Key-Independence).** *Let a signer’s private key  $sk$  consist of two independent elements  $sk = (kisk, ssk)$  and let  $pk = (kipk, spk)$  be the corresponding public key pair. VES is key-independent if there exists an efficient (encryption) algorithm  $\text{KI-Enc}$  such that  $\text{KI-Enc}(apk, kipk, kisk, \text{Sign}(ssk, m), m) \equiv \text{Create}(sk, apk, m)$  for all  $m \in \mathcal{M}$ .*

Note that the keys  $kisk$  and  $kipk$  are possibly the empty string, as in the case of the (key-independent) schemes of Boneh et al. [5] and of Lu et al. [13]. There, the algorithm  $\text{KI-Enc}$  is the encryption algorithm of the El Gamal public key encryption scheme.

**Theorem 3.** *Let VES be an extractable and key-independent verifiably encrypted signature scheme. VES is unforgeable if and only if the underlying signature scheme  $\text{DSig}$  is unforgeable.*

*Proof.* We have to show two directions. We begin with the (interesting) direction, showing that the existence of an algorithm  $\mathcal{A}_1$  that successfully forges a verifiably encrypted signature implies the existence of an adversary  $\mathcal{B}$  that successfully breaks  $\text{DSig}$ .  $\mathcal{B}$  gets as input the public key  $spk$  of the underlying signature scheme  $\text{DSig}$  and has access to a signing oracle  $\text{Sign}(ssk, \cdot)$ , that upon input a message  $m$  returns the corresponding signature. Subsequently,  $\mathcal{B}$  picks a key pair for the simulation of the adjudicator  $(ask, apk) \leftarrow \text{AdjKg}(1^n)$  and a VES key pair  $(sk, pk) \leftarrow \text{Kg}(1^n)$ . It then replaces the signature verification key in  $pk$  with  $spk$ , i.e.,  $pk = (kipk, spk)$ , and runs  $\mathcal{A}_1$  in a black-box simulation on input  $(apk, pk)$ . During the simulation,  $\mathcal{A}_1$  may invoke its creation oracle  $\text{C}$  on a message  $m$ . Algorithm  $\mathcal{B}$  answers this query as follows. It first generates the signature  $\sigma \leftarrow \text{Sign}(ssk, m)$  with the help of its external signing oracle and

outputs  $\omega \leftarrow \text{KI-Enc}(apk, kpk, kisk, \sigma, m)$ . Whenever  $\mathcal{A}_1$  invokes its adjudication oracle  $\mathbf{A}$  on a valid tuple  $(m, \omega)$ ,  $\mathcal{B}$  returns  $\sigma \leftarrow \text{Adj}(ask, apk, pk, \omega, m)$ . Eventually,  $\mathcal{A}_1$  stops, outputting a tuple  $(m^*, \omega^*)$ ; then  $\mathcal{B}$  computes  $\sigma^* \leftarrow \text{Adj}(ask, apk, pk, \omega^*, m^*)$  and outputs  $(m^*, \sigma^*)$  as its forged signature.

For the analysis, it is assumed that  $\mathcal{A}_1$  succeeds with non-negligible probability  $\epsilon(n)$ . Observe that  $\mathcal{B}$  performs a perfect simulation from  $\mathcal{A}_1$ 's point of view because VES is key-independent, i.e.,  $\mathcal{B}$  can choose the keys for KI-Enc independently of  $ssk$ .

Note that  $\mathcal{A}_1$  succeeds if it outputs a “fresh” tuple  $(m^*, \omega^*)$ . Here, the freshness condition means that  $\mathcal{A}_1$  has neither queried its creation oracle nor the adjudication oracle about  $m^*$ . But if  $\mathcal{A}_1$  has never sent  $m^*$  to one of the oracles, then  $\mathcal{B}$  has never queried its signing oracle about  $m^*$ . Since the scheme VES is extractable,  $\mathcal{B}$  always outputs a valid message-signature pair  $(m^*, \sigma^*)$  whenever  $\mathcal{A}_1$  provides a valid verifiably encrypted signature. This, however, contradicts the assumption that DSig is unforgeable.

The other direction shows how to break unforgeability of the verifiably encrypted signature scheme with the help of an adversary  $\mathcal{A}_2$  that forges the underlying signature scheme. The idea of the proof is to output the key-independent encryption (using KI-Enc) of the forgery obtained from  $\mathcal{A}_2$ .  $\square$

## 5 The Need for Abuse-Freeness

Garay, Jakobsson, and MacKenzie already consider abuse-freeness for optimistic fair exchange [9]. Their definition demands that no single signer should be able to prove to any third party that he can determine the outcome of the protocol. Since VES schemes are typically non-interactive, and since the verification equation ensures that the contained signature is valid, this definition seems inapplicable to the VES scenario.

Intuitively, abuse-freeness means that an adversary who may covertly cooperate with the adjudicator is unable to compute a verifiably encrypted signature on behalf of another party. We model this in an experiment where the malicious signer  $\mathcal{A}$  receives the private key of an adjudicator and the public key of the honest signer which we model as oracle  $\mathbf{C}$ . The adversary  $\mathcal{A}$  succeeds if it outputs a “fresh” tuple  $(m^*, \omega^*)$ , i.e., a message  $m^*$  and an encrypted signature  $\omega^*$  s.t.  $\mathcal{A}$  has never queried  $m^*$  to  $\mathbf{C}$ . Observe that giving  $\mathcal{A}$  access to an adjudication oracle would be redundant, since  $\mathbf{A}$  can be simulated with  $ask$ .

**Definition 4 (Abuse-freeness).** VES is abuse-free if for any efficient algorithm  $\mathcal{A}$  the probability that experiment Abuse evaluates to 1 is negligible (as a function of  $n$ ), where

**Experiment** Abuse $_A^{\text{VES}}(n)$   
 $(apk, ask) \leftarrow \text{AdjKg}(1^n)$   
 $(sk, pk) \leftarrow \text{Kg}(1^n)$   
 $(m^*, \omega^*) \leftarrow \mathcal{A}^{\mathbf{C}(sk, apk, \cdot)}(apk, ask, pk)$   
 Return 1 iff  $\text{VesVf}(apk, pk, \omega^*, m^*) = 1$  and  
 $\mathcal{A}$  has never queried  $\mathbf{C}(sk, apk, \cdot)$  about  $m^*$ .

This definition can be strengthened even further as  $\mathcal{A}$  could be allowed to choose the public key  $apk$ . We call schemes satisfying the stronger notion *strongly abuse-free* (see [14]).

*Relation to the Security Model.* We discuss the relation between abuse-freeness and the other security requirements. The interesting point is that for key-independent, extractable schemes, abuse-freeness is already guaranteed. In addition, we can separate abuse-free VES schemes from those satisfying the model of Boneh et al. For the separation, we need to recall the definition of public-key encryption schemes. A public-key encryption scheme  $\mathcal{E}$  is a tuple of efficient algorithms (Pk-Kg, Enc, Dec), where  $(pk_{\mathcal{E}}, sk_{\mathcal{E}}) \leftarrow \text{Pk-Kg}(1^n)$  is a key-generation algorithm that outputs a public-encryption key  $pk_{\mathcal{E}}$  and a private-decryption key  $sk_{\mathcal{E}}$ . The encryption algorithm  $C \leftarrow \text{Enc}(pk, m)$  takes as input a message  $m$  from some underlying plaintext space  $\mathcal{M}$  and outputs a ciphertext  $C$ . The decryption algorithm  $m \leftarrow \text{Dec}(sk_{\mathcal{E}}, C)$ , upon input the private key  $sk_{\mathcal{E}}$  and a ciphertext  $C$ , returns the plaintext  $m$ . It is assumed that  $\text{Prob}[\text{Dec}(sk_{\mathcal{E}}, \text{Enc}(pk, m)) = m] = 1$  (except for a negligible amount).

**Definition 5 (CPA Indistinguishability).** *A public key encryption scheme  $\mathcal{E} = (\text{Pk-Kg}, \text{Enc}, \text{Dec})$  is indistinguishable under chosen plaintext attacks (IND-CPA) if for any efficient algorithm  $\mathcal{A}$  the probability that the experiment  $\text{INDCPA}_{\mathcal{A}}^{\mathcal{E}}$  evaluates to 1 is negligibly close to  $1/2$ , where*

*Experiment  $\text{INDCPA}_{\mathcal{A}}^{\mathcal{E}}(n)$*   
 $(pk_{\mathcal{E}}, sk_{\mathcal{E}}) \leftarrow \text{Pk-Kg}(1^n)$   
 $b \leftarrow \{0, 1\}$   
 $b^* \leftarrow \mathcal{A}^{\text{Enc}(pk_{\mathcal{E}}, b, \cdot)}(pk_{\mathcal{E}})$  // Enc takes  $m_0, m_1 \in \mathcal{M}$ , s.t.  $|m_0| = |m_1|$  as input.  
*Return 1 iff  $b^* = b$ .*

**Theorem 4.** *If an IND-CPA secure public-key encryption scheme  $\mathcal{E}$ , and a secure verifiably encrypted signature scheme VES exist, then there is a secure scheme VES', which is not abuse-free.*

*Proof.* We build the scheme VES' out of VES, such that VES' is unforgeable and opaque, but such that a malicious adjudicator is able to reveal the private signing key. VES' is defined as:

**Key Generation:**  $\text{Kg}' \equiv \text{Kg}$ .  $\text{AdjKg}'$  calls  $(ask, apk) \leftarrow \text{AdjKg}(1^n)$  and  $(sk_{\mathcal{E}}, pk_{\mathcal{E}}) \leftarrow \text{Pk-Kg}(1^n)$ . It outputs  $(ask', apk') \leftarrow ((ask, sk_{\mathcal{E}}), (apk, pk_{\mathcal{E}}))$ .  
**VES Creation:**  $\text{Create}'(sk, apk', m)$  executes  $\omega' \leftarrow \text{Create}(sk, apk, m)$  and  $a \leftarrow \text{Enc}(pk_{\mathcal{E}}, sk)$ . It returns  $\omega' \leftarrow (\omega, a)$ .  
**VES Verification:**  $\text{VesVf}'(apk', pk, \omega', m)$  outputs  $\text{VesVf}(apk, pk, \omega, m)$ .  
**Adjudication:**  $\text{Adj}'(ask', apk', pk, \omega', m)$  outputs the result of  $\text{Adj}(ask, pk, \omega, m)$ .

Completeness, unforgeability, and opacity of VES' directly carry over from VES. Observe that the encryption scheme  $\mathcal{E}$  is a IND-CPA secure public-key encryption scheme, thus it does not reveal a single bit of the signing key. With the help of a malicious adjudicator, however, this is indeed possible.

Concerning abuse-freeness, the adversary  $\mathcal{A}$  gets an adjudication key pair  $(ask, apk)$  together with a public key  $pk$ . It selects two messages  $m_1$  and  $m_2$ , and invokes the creation oracle  $\mathsf{C}$  on  $m_1$ , obtaining  $(\omega', a)$ . Subsequently,  $\mathcal{A}$  extracts the private key  $sk \leftarrow \text{Dec}(sk_{\mathcal{E}}, a)$  and uses the private key  $sk$  to forge  $\omega^* \leftarrow \text{Create}(sk, apk, m_2)$ . A straightforward analysis shows that  $\mathcal{A}$  is efficient and always succeeds.  $\square$

In the following, we show that any key-independent, extractable verifiably encrypted signature scheme is also abuse-free. Again, this result helps reduce the effort of proving security.

**Theorem 5.** *A key-independent, extractable, and secure scheme VES is abuse-free if the underlying signature scheme DSig is unforgeable.*

*Proof.* Suppose that there exists an adversary  $\mathcal{A}$  that successfully breaks abuse-freeness with noticeable probability. We then show that  $\mathcal{A}$  can be used to forge ordinary signatures in DSig. The reduction  $\mathcal{B}$  against the unforgeability of DSig receives a public key  $spk$ . It generates  $(ask, apk) \leftarrow \text{AdjKg}(1^n)$ ,  $(sk, pk) \leftarrow \text{Kg}(1^n)$ , replaces the public signature verification key for DSig in  $pk$  with  $spk$  (the resulting key is  $pk'$ ), and runs  $\mathcal{A}(apk, ask, pk')$  as a black-box. Whenever  $\mathcal{A}$  queries  $m$  to  $\mathsf{C}$ ,  $\mathcal{B}$  calls its signing oracle  $\sigma \leftarrow \text{Sign}(sk, m)$  and computes  $\omega \leftarrow \text{KI-Enc}(apk, kispk, kisk, \sigma, m)$ . Finally,  $\mathcal{A}$  stops and outputs  $(m^*, \omega^*)$ .  $\mathcal{B}$  extracts the corresponding signature  $\sigma^* \leftarrow \text{Adj}(ask, apk, pk', \omega^*, m^*)$  and returns  $(m^*, \sigma^*)$ . Assuming that  $\mathcal{A}$  succeeds with noticeable probability  $\epsilon(n)$ , then  $\mathcal{A}$  has not queried  $m^*$  to  $\mathsf{C}$ ; as a consequence,  $\mathcal{B}$ 's attack is legitimate and it simulates  $\mathcal{A}$ 's environment perfectly, because VES is key-independent. Furthermore,  $\mathcal{B}$  is efficient, and as VES is extractable,  $\mathcal{B}$  succeeds with the same probability  $\epsilon(n)$  (except for a negligible part). This, however, is a contradiction.  $\square$

## 6 An Efficient Instantiation

In this section, we present an efficient verifiably encrypted signature scheme that is based on the Boneh-Boyer ( $\mathcal{BB}$ ) signature scheme. It is secure, extractable, and abuse-free in the standard model. For a simpler notation, we omit the generation of publicly known system parameters, and recall that  $z = \mathbf{e}(g_1, g_2)$ .

**Construction 1.** *Our instantiation works as follows.*

**Adjudicator Key Generation:**  $\text{AdjKg}(1^n)$  returns  $apk \leftarrow u_a = g_2^\beta$ , and  $ask \leftarrow \beta$ , for  $\beta \xleftarrow{\$} \mathbb{Z}_p^*$ .

**Key Generation:**  $\text{Kg}(1^n)$  calls  $((x, y), (u, v)) \leftarrow \mathcal{BB}.\text{Kg}(1^n)$ , computes  $\rho_1 \leftarrow (apk)^x$ ,  $\rho_2 \leftarrow (apk)^y$ , and returns the key pair  $((x, y), (u, v, \rho_1, \rho_2))$ , where  $sk = (x, y)$ ,  $pk = (u, v, \rho_1, \rho_2)$ .

**Signing, Verification:** Defined as in the  $\mathcal{BB}$  digital signature scheme.

**VES Creation:**  $\text{Create}(sk, apk, m)$  parses  $sk = (x, y)$  and  $apk = u_a$ . It computes  $(r, \sigma)$  using  $\mathcal{BB}.\text{Sign}((x, y), m)$ , selects  $s \xleftarrow{\$} \mathbb{Z}_p$  and sets  $\mu \leftarrow \psi(g_2)^s$ ,  $\sigma' \leftarrow \psi(u_a)^s$ . Then, it encrypts  $\sigma$  as  $\varpi = \sigma \sigma'$  and returns  $(r, \varpi, \mu)$ .

**VES Verification:**  $\text{VesVf}(apk, pk, \omega, m)$  parses  $apk = u_a$ ,  $pk = (u, v, \rho_1, \rho_2)$ ,  $\omega = (r, \varpi, \mu)$ . It returns 1 iff  $\mathbf{e}(\varpi, u g_2^m v^r) \cdot \mathbf{e}(\mu, \rho_1 \rho_2^r u_a^m)^{-1} = z$ ,  $\mathbf{e}(g_1, \rho_1) = \mathbf{e}(\psi(u_a), u)$ , and  $\mathbf{e}(g_1, \rho_2) = \mathbf{e}(\psi(u_a), v)$ .

**Adjudication:**  $\text{Adj}(ask, apk, pk, \omega, m)$  parses  $ask = \beta$ ,  $apk = u_a$ ,  $pk = (u, v, \rho_1, \rho_2)$ , and  $\omega = (r, \varpi, \mu)$ . If  $\text{VesVf}(apk, pk, \omega, m) = 1$ , then output  $\sigma \leftarrow \varpi / \mu^\beta$ .

Note that Construction 1 is key-independent, because we use the El Gamal encryption. Furthermore, it is complete (see Appendix B).

*A Word on Efficiency.* Note that we create  $\rho_1$  and  $\rho_2$  in  $\text{Kg}$ , which is originally not permitted by the model because  $\text{Kg}$  does not have access to  $apk$ . It is, however, reasonable to assume the existence of a unique adjudicator, whose parameter are known and set before the initialization of the key generation. Otherwise, one could compute  $\rho_1$  and  $\rho_2$  in  $\text{Create}$ , which would be less efficient due to larger computational costs and an increased output size. Similarly, we eliminate the need to check the soundness of  $\rho_1$  and  $\rho_2$  in  $\text{VesVf}$  by assuming that all user keys are registered and that the universally trusted registration authority already verified them.

## 6.1 Proof of Security

For the following security proofs, let  $T_{\text{AdjKg}}, T_{\text{Kg}}$  be cost functions for adjudication and signature key generation, and let  $T_{\text{Create}}, T_{\text{Adj}}$  be the cost functions for creation and adjudication of verifiably encrypted signatures. The next theorem proves extractability, which implies unforgeability by Theorem 3.

**Theorem 6.** *Construction 1 is extractable.*

*Proof.* We show that if a verifiably encrypted signature  $\omega$  verifies, then it is always possible to extract a valid  $\mathcal{BB}$  signature. From  $\text{VesVf}$ , we have

$$\begin{aligned} \text{V1} & \quad \mathbf{e}(\varpi, u g_2^m v^r) \cdot \mathbf{e}(\mu, \rho_1 \rho_2^r u_a^m)^{-1} = z; \\ \text{V2} & \quad \mathbf{e}(g_1, \rho_1) = \mathbf{e}(\psi(u_a), u) \text{ and } \mathbf{e}(g_1, \rho_2) = \mathbf{e}(\psi(u_a), v). \end{aligned}$$

After applying the adjudication algorithm on  $\omega$ ,  $\text{Vf}$  evaluates:

$$\begin{aligned} \mathbf{e}(\varpi / \mu^\beta, u g_2^m v^r) &= \mathbf{e}(\varpi, u g_2^m v^r) \cdot \mathbf{e}(\mu, u g_2^m v^r)^{-\beta} \\ &\stackrel{\text{V1}}{=} \mathbf{e}(\mu, \rho_1 \rho_2^r u_a^m) \cdot z \cdot \mathbf{e}(\mu, u g_2^m v^r)^{-\beta} \\ &\stackrel{\text{V2}}{=} \mathbf{e}(\mu, u^\beta v^{r\beta} g_2^{\beta m}) \cdot z \cdot \mathbf{e}(\mu, u g_2^m v^r)^{-\beta} = z. \end{aligned}$$

□

**Corollary 1.** *If the  $\mathcal{BB}$  signature scheme is  $(t + T_{\text{AdjKg}}(n) + T_{\text{Kg}}(n) + q_C T_{\text{Create}}(n) + (q_A + 1) T_{\text{Adj}}(n), q_C, \epsilon)$ -unforgeable, then Construction 1 is  $(t, q_S, q_A, \epsilon)$ -unforgeable.*

*Proof.* The proof follows immediately from Theorem 3 in conjunction with Theorem 6. □

The opacity of our verifiably encrypted signature scheme depends on the assumption that, given  $q$  tuples  $(c_i, g_1^{1/(x+c_i)})$ ,  $i = 2, \dots, q+1$ , it is difficult to extract the value  $g_1^{1/(x+c_1)}$  from an El Gamal encryption  $(c_1, g_1^{\beta s+1/(x+c_1)})$ . It is well known that the El Gamal encryption is provably one-way if the computational Diffie-Hellman (CDH) problem is hard, and that the scheme is a CPA secure encryption scheme if the decisional Diffie-Hellman (DDH) holds. More formally, we require that the following problem is computational infeasible:

**Definition 6 ( $q$ -SDH Extraction Problem).** *In the  $q$ -SDH extraction problem (SDHE), an adversary gets as input*

$$\left( g_1, g_1^s, g_2, g_2^\beta, g_2^x, g_2^{\beta x}, \left( c_1, g_1^{\beta s+1/(x+c_1)} \right), \left\{ \left( c_i, g_1^{1/(x+c_i)} \right) \right\}_{i=2}^{q+1} \right)$$

and is required to compute  $\left( c_1, g_1^{1/(x+c_1)} \right)$ .

**Definition 7.** *The  $q$ -SDHE problem is  $(t, \epsilon)$ -hard if no  $t$ -time algorithm  $\mathcal{A}$  has advantage at least  $\epsilon$  in solving the  $q$ -SDHE problem, i.e., no such algorithm has advantage*

$$\text{Prob} \left[ \left( c_1, g_1^{1/(x+c_1)} \right) \leftarrow \mathcal{A} \left( g_1, g_1^s, g_2, g_2^\beta, g_2^x, g_2^{\beta x}, \left( c_1, g_1^{\beta s+1/(x+c_1)} \right), \left\{ \left( c_i, g_1^{1/(x+c_i)} \right) \right\}_{i=2}^{q+1} \right) \right] \geq \epsilon.$$

We assume that  $q$ -SDHE is  $(t, \epsilon)$ -hard for any polynomial  $t$  in  $n$  and a negligible  $\epsilon$ . Based on this assumption, we can now prove that Construction 1 is opaque.

**Theorem 7.** *If the  $q_C$ -SDHE extraction problem is  $(t + T_{\text{Kg}}(n) + q_C T_{\text{Create}}(n) + q_A T_{\text{Adj}}(n), \epsilon/q_C)$ -hard then our scheme is  $(t, q_C, q_A, \epsilon)$ -opaque.*

*Proof.* A natural observation is that there are two possibilities to break opacity. One is to directly forge the underlying signature scheme; the second one is to extract an ordinary signature. Since the case that the adversary forges the underlying  $\mathcal{BB}$  signature is already covered, we concentrate on the second class, of adversaries that “decrypt” a given verifiably encrypted signature. We show how to use such an adversary in order to refute the  $q_C$ -SDHE assumption.

The proof follows [2, Lemma 10] in the way of simulating adaptive queries, but differs in the point that the adversary extracts a previously queried encrypted element. We distinguish two classes of adversaries. We say that an algorithm  $\mathcal{A}$  is a

1. type-1 adversary, denoted  $\mathcal{A}_1$ , if it
  - (a) makes a verifiably encryption query for a message  $m = -x$ , or
  - (b) outputs an extraction  $(m^*, r^*, g_1^{1/(x+y r^*+m^*)})$ , where  $m^* + y r^* \notin \{c_1, \dots, c_{q+1}\}$ .
2. type-2 adversary, denoted  $\mathcal{A}_2$ , if it

- (a) never makes a verifiably encryption query for a message  $m = -x$ , and
- (b) outputs an extraction  $(m^*, r^*, g_1^{1/(x+yr^*+m^*)})$ ,  
where  $m^* + yr^* \in \{c_1, \dots, c_{q+1}\}$ .

Note that these types cover all possible adversaries, and observe that they are identical to the partitions in [3]. As already pointed out by Boneh and Boyen, the type-1 adversary directly leads to a forgery of the underlying signature scheme, thus we omit this part of the proof and refer the reader to [3].

Now we show how to solve the  $q_C$ -SDHE problem by giving a reduction  $\mathcal{B}_2$  black-box access to a type-2 adversary  $\mathcal{A}_2$ . The idea of the proof is that we use the technique of Boneh and Boyen in order to answer the queries adaptively. We guess which answer of  $\mathcal{C}$  that  $\mathcal{A}_2$  will decrypt and inject the SDHE challenge  $(c_1, g_1^{\beta s+1/(x+c_1)})$ .

*Type-2 adversary.* We describe the simulator  $\mathcal{B}_2$  interacting with a type-2 adversary, denoted by  $\mathcal{A}_2$ , in order to solve  $q_C$ -SDHE.

**Setup:** The algorithm  $\mathcal{B}_2$  gets as input  $g_1, g_1^s, g_2, g_2^\beta, g_2^x, g_2^{\beta x}$ , together with the values  $(c_1, g_1^{\beta s+1/(x+c_1)})$ ,  $\left\{ (c_i, g_1^{1/(x+c_i)}) \right\}_{i=2}^{q_C+1}$ . It selects  $y \xleftarrow{\$} \mathbb{Z}_p$  and sets  $u \leftarrow g_2^x, v \leftarrow g_2^y, \rho_1 \leftarrow g_2^{\beta x}, \rho_2 \leftarrow g_2^{\beta y}$ . Furthermore,  $\mathcal{B}_2$  picks a random index  $j \xleftarrow{\$} \{1, \dots, q_C + 1\}$  and initializes a counter  $\ell \leftarrow 1$  together with a list  $Q \leftarrow \emptyset$ . It runs  $\mathcal{A}_2$  on input  $(apk, pk) \leftarrow (\psi(g_2^\beta), (u, v, \rho_1, \rho_2))$ .

**VES Queries:** Whenever  $\mathcal{A}_2$  queries  $m$  to  $\mathcal{C}$ ,  $\mathcal{B}_2$  increments  $\ell \leftarrow \ell + 1$ . Case 1 ( $\ell = j$ ):  $\mathcal{B}_2$  sets:  $r_\ell \leftarrow (c_1 - m)/y$ ,  $\mu_\ell \leftarrow g_1^s$ , and  $\varpi \leftarrow g_1^{\beta s+1/(x+c_1)}$ . Case 2 ( $\ell \neq j$ ):  $\mathcal{B}_2$  selects  $s_\ell \xleftarrow{\$} \mathbb{Z}_p$ , sets:  $\mu_\ell \leftarrow g_1^{s_\ell}$  and  $r_\ell \leftarrow (c_\ell - m)/y$ , and computes  $\varpi \leftarrow g_1^{\beta s_\ell} g_1^{1/(x+c_\ell)}$ .  $\mathcal{B}_2$  stores  $(m_\ell \leftarrow m, r_\ell, g_1^{1/(x+c_\ell)})$  in  $Q$ . In either case,  $\mathcal{B}_2$  returns  $(r_\ell, \varpi_\ell, \mu_\ell)$ .

**Adjudication Queries:** Whenever  $\mathcal{A}_2$  queries a tuple  $(m, (r, \varpi, \mu))$  to  $\mathcal{A}$ ,  $\mathcal{B}_2$  checks that the tuple is valid and returns **fail** if this is not the case. Let's assume that  $(m, (r, \varpi, \mu))$  is valid. According to Theorem 1 we know that algorithm  $\mathcal{A}_2$  must have queried  $m$  to  $\mathcal{C}$ . If  $i = j$ , then  $\mathcal{B}_2$  aborts. Otherwise, if  $i \neq j$ , let  $i \in \{1, \dots, |Q|\}$  be the corresponding index of the query. Then,  $\mathcal{B}_2$  returns  $(r_\ell, g_1^{1/(x+c_\ell)})$ .

**Output:** Finally,  $\mathcal{A}_2$  stops, outputting a tuple  $(m^*, r^*, g_1^{1/(x+m^*+yr^*)})$ .  $\mathcal{B}_2$  sets  $c^* \leftarrow m^* + yr^*$ . It aborts if  $c_1 \neq c^*$ , and otherwise stops, outputting  $(c^*, g_1^{1/(x+c^*)})$ .

*Analysis.* Algorithm  $\mathcal{B}_2$  performs a perfect simulation of  $\mathcal{C}$ . Note that  $r_\ell$  is uniformly distributed over  $\mathbb{Z}_p \setminus \{-\frac{x+m_\ell}{y}\}$  and that for the oracle answers of  $\mathcal{C}$ , we have:

$$\begin{aligned} \mathbf{e}(\varpi_\ell, u g_2^{m_\ell} v^{r_\ell}) \cdot \mathbf{e}(\mu_\ell, \rho_1 \rho_2^{r_\ell} u_a^{m_\ell})^{-1} = \\ \mathbf{e}\left(\sigma_\ell g_1^{\beta s_\ell}, u g_2^{m_\ell+yr_\ell}\right) \cdot \mathbf{e}\left(\mu_\ell, u^\beta g_2^{\beta(y+m_\ell r)}\right)^{-1} = z. \end{aligned}$$

$\mathcal{B}_2$  also simulates the oracle  $\mathbf{A}$  perfectly (for  $i \neq j$ ) because for its output  $(r_\ell, \sigma_\ell)$ , we have

$$\mathbf{e}(\sigma_\ell, u g_2^{m_\ell} v^{r_j}) = \mathbf{e}(\sigma_\ell, u g_2^{m_\ell + y r_j}) = \mathbf{e}(\sigma_\ell, u g_2^{c_j}) = z.$$

Observe that  $\mathcal{A}_2$  can never query the adjudication oracle  $\mathbf{A}$  without having invoked  $\mathbf{C}$  before due to Theorem 1, i.e.  $q_{\mathbf{A}} \leq q_{\mathbf{C}}$ . The same argument is applicable if  $\mathcal{A}_2$  sends valid tuple  $(m, (r, \varpi, \mu))$  to  $\mathbf{A}$ , such that  $m$  is in the query list  $Q$ , but with a different value  $r$ . Both cases would contradict the strong unforgeability of the  $\mathcal{BB}$  signature scheme.

Assuming  $\mathcal{A}_2$  succeeds with non-negligible probability  $\epsilon(n)$ . According to the partition of adversaries, we know that  $\mathcal{A}_2$  “decrypts” a given  $\varpi$  obtained from  $\mathbf{C}$ . Since  $\mathcal{B}_2$  guesses the index of the corresponding query, its success probability is lessened by a factor of  $1/q_{\mathbf{C}}$ . However, it still succeeds with non-negligible probability  $\epsilon(n)/q_{\mathbf{C}}$  in the  $q_{\mathbf{C}}$ -SDHE problem — a contradiction.  $\square$

**Corollary 2.** *If the  $\mathcal{BB}$  signature scheme is unforgeable, then Construction 1 is abuse-free.*

*Proof.* The proof follows immediately from Theorem 6 and Theorem 5.  $\square$

## Acknowledgments

We thank Heike Busch, Marc Fischlin, Cristina Onete, Michael Schneider, and the anonymous reviewers for their valuable comments.

## References

1. N. Asokan, Victor Shoup, and Michael Waidner. *Optimistic Fair Exchange of Digital Signatures*. *IEEE Journal on Selected Areas in Communications*, 18(4):593–610, 2000.
2. Dan Boneh and Xavier Boyen. *Short Signatures Without Random Oracles*. *Advances in Cryptology — Eurocrypt’04*, Volume 3027 of *Lecture Notes in Computer Science*, pages 56–73. Springer-Verlag, 2004.
3. Dan Boneh and Xavier Boyen. *Short Signatures Without Random Oracles and the SDH Assumption in Bilinear Groups*. *Journal of Cryptology*, 21(2):149–177, 2008.
4. Bao, Deng, and Mao. *Efficient and Practical Fair Exchange Protocols with Off-Line TTP*. *RSP: 19th IEEE Computer Society Symposium on Research in Security and Privacy*. IEEE Computer Society Press, 1998.
5. Dan Boneh, Craig Gentry, Ben Lynn, and Hovav Shacham. *Aggregate and Verifiably Encrypted Signatures from Bilinear Maps*. *Advances in Cryptology — Eurocrypt’03*, *Lecture Notes in Computer Science*, pages 416–432. Springer-Verlag, 2003.
6. Paulo S. L. M. Barreto and Michael Naehrig. *Pairing-Friendly Elliptic Curves of Prime Order*. *Selected Areas in Cryptography: 12th International Workshop, SAC 2005*, Kingston, Canada, Volume 3897 of *Lecture Notes in Computer Science*, pages 319–331. Springer-Verlag, 2006.

7. Ran Canetti, Oded Goldreich, and Shai Halevi. *The random oracle methodology, revisited*. *J. ACM*, 51(4):557–594, 2004.
8. Yevgeniy Dodis, Pil Joong Lee, and Dae Hyun Yum. *Optimistic Fair Exchange in a Multi-user Setting*. Public-Key Cryptography (PKC)'07, Lecture Notes in Computer Science, pages 118–133. Springer-Verlag, 2007.
9. Juan A. Garay, Markus Jakobsson, and Philip D. MacKenzie. *Abuse-Free Optimistic Contract Signing*. Advances in Cryptology — Crypto'99, Lecture Notes in Computer Science, pages 449–466. Springer-Verlag, 1999.
10. Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. *A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks*. *SIAM J. Comput.*, 17(2):281–308, 1988.
11. Jens Groth, Rafail Ostrovsky, and Amit Sahai. *Perfect Non-interactive Zero Knowledge for NP*. Advances in Cryptology — Eurocrypt 2006, Lecture Notes in Computer Science, pages 339–358. Springer-Verlag, 2006.
12. Florian Hess. *On the security of the verifiably-encrypted signature scheme of Boneh, Gentry, Lynn and Shacham*. *Information Processing Letters*, 89(3):111 – 114, 2004.
13. Steve Lu, Rafail Ostrovsky, Amit Sahai, Hovav Shacham, and Brent Waters. *Sequential Aggregate Signatures and Multisignatures Without Random Oracles*. Advances in Cryptology — Eurocrypt 2006, Lecture Notes in Computer Science, pages 465–485. Springer-Verlag, 2006.
14. Markus Rückert and Dominique Schröder. *Security of Verifiably Encrypted Signatures and a Construction Without Random Oracles (Extended Version)*. Number 2009/027 in Cryptology eprint archive. [eprint.iacr.org](http://eprint.iacr.org), 2009.
15. Brent Waters. *Efficient Identity-Based Encryption Without Random Oracles*. Advances in Cryptology — Eurocrypt'05, Lecture Notes in Computer Science, pages 114–127. Springer-Verlag, 2005.
16. Fangguo Zhang, Reihaneh Safavi-Naini, and Willy Susilo. *Efficient Verifiably Encrypted Signature and Partially Blind Signature from Bilinear Pairings*. Progress in Cryptology — Indocrypt 2003, pages 191–204. Springer-Verlag, 2003.

## A Secure Signature Schemes

Recall that a digital signature scheme DSig is defined as:

**Definition 8.** *A signature scheme consists of a triple of efficient algorithms DSig = (Kg, Sign, Vf), where:*

**Key Generation:**  $\text{Kg}(1^n)$  outputs a private signing key  $sk$  and a public verification key  $pk$ .

**Signature Generation:**  $\text{Sign}(ssk, m)$  outputs a signature  $\sigma$  under  $ssk$ , on a message  $m$  chosen from the message space  $\mathcal{M}$ .

**Signature Verification:** The algorithm  $\text{Vf}(spk, \sigma, m)$  outputs 1 iff  $\sigma$  is a valid signature on  $m$  under  $spk$ .

*Signature schemes are complete if for any  $(ssk, spk) \leftarrow \text{Kg}(1^n)$ , any message  $m \in \mathcal{M}$ , and any  $\sigma \leftarrow \text{Sign}(ssk, m)$ , we have:  $\text{Vf}(spk, \sigma, m) = 1$ .*

The security of signature schemes is proven against existential forgery under adaptive chosen message attacks (EU-CMA) [10]. In this model, an adversary

adaptively invokes a signing oracle and is successful if it outputs a signature on a *fresh* message. In the following, we use a slightly stronger notion, known as strong unforgeability (SU-CMA). Here, the adversary also succeeds if it computes a fresh message-signature *pair*.

**Definition 9.** A signature scheme  $\text{DSig}$  is strongly unforgeable under adaptive chosen message attacks (SU-CMA) if for any efficient algorithm  $\mathcal{A}$  the probability that the experiment  $\text{sForge}_{\mathcal{A}}^{\text{DSig}}$  evaluates to 1 is negligible (as a function of  $n$ ).

**Experiment**  $\text{sForge}_{\mathcal{A}}^{\text{DSig}}(n)$

$$(ssk, spk) \leftarrow \text{Kg}(1^n)$$

$$(m^*, \sigma^*) \leftarrow \mathcal{A}^{\text{Sign}(ssk, \cdot)}(pk)$$

let  $(m_i, \sigma_i)$  be the answer returned by  $\text{Sign}(ssk, \cdot)$  on input  $m_i$ , for  $i = 1, \dots, k$ .

Return 1 iff  $\forall f(\text{spk}, m^*, \sigma^*) = 1$  and  $(m^*, \sigma^*) \notin \{(m_1, \sigma_1), \dots, (m_k, \sigma_k)\}$ .

A signature scheme  $\text{DSig}$  is  $(t, q_S, \epsilon)$ -secure if no adversary running in time at most  $t$ , invoking the signing oracle at most  $q_S$  times, outputs a valid forgery  $(m^*, \sigma^*)$  with probability larger than  $\epsilon$ .

## B Completeness in Section 6

Concerning completeness, we prove the following proposition.

**Proposition 1.** *Construction 1 is complete.*

*Proof.* We show that for all honestly generated key pairs, for all messages  $m \in \mathcal{M}$ , and for any verifiably encrypted signature generated by the Create algorithm, the  $\text{VesVf}$  algorithm returns 1. We have:

$$\begin{aligned} & \mathbf{e}(\varpi, u g_2^m v^r) \cdot \mathbf{e}(\mu, \rho_1 \rho_2^x u_a^m)^{-1} = \\ & = \mathbf{e}\left(g_1^{\frac{1}{x+m+yr}} \psi(u_a)^s, u g_2^m v^r\right) \cdot \mathbf{e}\left(\mu, u_a^x u_a^{yr} g_2^{\beta m}\right)^{-1} \\ & = \mathbf{e}\left(g_1^{\frac{1}{x+m+yr}}, g_2^x g_2^m g_2^{yr}\right) \cdot \mathbf{e}(\psi(u_a)^s, u g_2^m v^r) \cdot \mathbf{e}\left(\mu, g_2^{\beta(x+m+yr)}\right)^{-1} \\ & = \mathbf{e}(g_1, g_2)^{\frac{x+m+yr}{x+m+yr}} \cdot \mathbf{e}(\mu^\beta, u g_2^m v^r) \cdot \mathbf{e}(\mu^\beta, u g_2^m v^r)^{-1} = z. \end{aligned}$$

We further show that if the adjudicator extracts a signature  $\sigma$ , then  $\sigma$  can be verified as a valid  $\mathcal{BB}$  signature, i.e., running the  $\mathcal{BB}$  verification algorithm yields:

$$\begin{aligned} \mathbf{e}(\varpi/\mu^\beta, u g_2^m v^r) &= \mathbf{e}(\sigma \psi(u_a)^s, u g_2^m v^r) \cdot \mathbf{e}(\mu, u g_2^m v^r)^{-\beta} \\ &= \mathbf{e}(\sigma, g_2^{x+m+yr}) \cdot \mathbf{e}(\psi(u_a)^s, u g_2^m v^r) \cdot \mathbf{e}(\mu^\beta, u g_2^m v^r)^{-1} \\ &= \mathbf{e}(g_1, g_2)^{\frac{x+m+yr}{x+m+yr}} \cdot \mathbf{e}(\mu^\beta, u g_2^m v^r) \cdot \mathbf{e}(\mu^\beta, u g_2^m v^r)^{-1} = z. \end{aligned}$$

□

## C A Stronger VES Model

In the following, we discuss how the security definition of VES schemes can be strengthened even further. We apply the idea of strong unforgeability in the digital signature context to the definitions of unforgeability and opacity in the VES context. We show in the full version of this paper that the new model is strictly stronger and give a first instantiation.

**Definition 10.** *A verifiably encrypted signature scheme VES is called strongly unforgeable if for any efficient algorithm  $\mathcal{A}$ , the probability that the following experiment evaluates to 1 is negligible (as a function of  $n$ ).*

**Experiment**  $\text{VesSForge}_A^{\text{VES}}(n)$   
 $(sk, pk) \leftarrow \text{Kg}(1^n)$   
 $(ask, apk) \leftarrow \text{AdjKg}(1^n)$   
 $(m^*, \omega^*) \leftarrow \mathcal{A}^{\mathcal{C}(sk, apk, \cdot), A(ask, apk, pk, \cdot, \cdot)}(pk, apk)$   
*Let  $C = \{(m_{C_1}, \omega_{C_1}), \dots, (m_{C_k}, \omega_{C_k})\}$  be the query-answer pairs of  $C$ .*  
*Return 1 iff  $\text{VesVf}(apk, pk, \omega^*, m^*) = 1$  and  $(m^*, \omega^*) \notin C$ .*

The main difference to unforgeability is that the adversary is allowed to output a forgery  $\omega^*$  for a message  $m^*$  that has already been sent to  $C$ , as long as the forged verifiably encrypted signature is different from the corresponding answer of  $C$ . This last condition ensures that verifiably encrypted signatures cannot be reused by simply re-randomizing them.

Moreover, the adversary is allowed to query  $A$  on  $m^*$  in order to obtain an ordinary signature  $\sigma^*$ . In this scenario, however, we require that reusing ordinary signatures as verifiably encrypted signatures without having knowing some secret information should be hard.

**Definition 11.** *A verifiably encrypted signature scheme VES is called strongly opaque if for any efficient algorithm  $\mathcal{A}$ , the probability that the following experiment evaluates to 1 is negligible (as a function of  $n$ ).*

**Experiment**  $\text{SOPac}_A^{\text{VES}}(n)$   
 $(sk, pk) \leftarrow \text{Kg}(1^n)$   
 $(ask, apk) \leftarrow \text{AdjKg}(1^n)$   
 $(m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathcal{C}(sk, apk, \cdot), A(ask, apk, pk, \cdot, \cdot)}(pk, apk)$   
*Let  $A = \{(m_{A_1}, \sigma_{A_1}), \dots, (m_{A_\ell}, \sigma_{A_\ell})\}$  be the query-answer pairs of  $A$ .*  
*Return 1 iff  $\text{Vf}(apk, \sigma^*, m^*) = 1$  and  $(m^*, \sigma^*) \notin A$ .*

Here again, as opposed to opacity, the adversary is allowed to query the oracles on the message it is about to output as a forgery. The forgery, however, must be different from what the adversary obtained from  $A$  on that message.

**Definition 12 (Strong Security of VES).** *A verifiably encrypted signature scheme VES is called strongly secure if it is strongly unforgeable and strongly opaque.*

We show in the full version of this paper that a modification of our construction satisfies the stronger notion.