Non-Binding (Designated Verifier) Signature

Ehsan Ebrahimi
Department of Computer Science & SnT, University of Luxembourg
ehsan.ebrahimi@uni.lu

***Abstract*—We argue that there are some scenarios in which plausible deniability might be desired for a digital signature scheme. For instance, the non-repudiation property of conventional signature schemes is problematic in designing an *Instant Messaging* system (WPES 2004). In this paper, we formally define a *non-binding signature* scheme in which the Signer is able to disavow her own signature if she wants, but, the Verifier is not able to dispute a signature generated by the Signer. That is, the Signer is able to convince a third party** Judge **that she is the owner of a signature without disclosing her secret information. We propose a signature scheme that is *non-binding* and *unforgeable*. Our signature scheme is post-quantum secure if the underlying cryptographic primitives are post-quantum secure. In addition, a modification to our *non-binding* signature scheme leads to an *Instant Messaging* system that is of independent interest.**

***Index Terms*—Designated Verifier Signature, Plausible Deniability, End-to-End Encryption.**

I. Introduction

A digital signature is probably one of the most used cryptographic primitive in real life applications, in which, a Signer is able to sign a digital message using a secret-key sk and anyone can verify this signature using a public-key pk. A digital signature has to have some security properties. For instance, anyone beside the actual signer should not be able to generate a valid signature on a new message even after seeing many signatures generated by the Signer (unforgeability). Or the Signer should not be able to disavow her own signature (non-repudiation). The non-repudiation property will be achieved inherently since a signature is publicly verifiable using pk.

However, there are some scenarios in which the non-repudiation property might not be desirable. For instance, the non-repudiation property of a digital signature scheme is discussed as a problematic issue in [10] while making a confidential and authentic online conversation using PGP [30] and a digital signature scheme. Therefore, the *Off-the-Record Messaging* (OTR) protocol [10] avoids the use a conventional signature scheme for authenticity of the message since a pair (encrypted message, signature) is verifiable using pk (of a conventional digital signature) and consequently the sender of the message would leak to an eavesdropper. In Section VI, we propose an *instant messaging* protocol based on the approach and techniques in this paper. Our *instant messaging* protocol is post-quantum secure (in contrast to [10]) if the underlying cryptographic primitives are post-quantum secure. The *Off-the-Record Messaging* protocol [10] has resulted in the Signal protocol that is currently used to transmit hundreds of billions of messages per day [16] and has led to extensive follow-up research [1, 2, 4, 5, 6, 7, 13, 17, 20, 21, 22, 23, 24, 25, 33, 36, 37, 47, 48, 51, 52, 58, 59, 60, 62].

Another scenario when a charity organization wants to offer some charity contracts to philanthropists. Obviously, a philanthropist will be doubtful to sign a charity contract that is legally binding. In addition, the philanthropist would like to keep track of his/her beneficence to challenge the charity organization in the case of corruption. For instance, a charity organization prepares a contract in which the signer donates 10k EUR to the organization in a duration of one year. Obviously, the philanthropist prefers a non-legally binding contract because he is not sure if its financial status till the end of the year allows this transaction or not. In the other hand, once the money transaction is done, the philanthropist prefers to have a proof that he has participated in the charity activity to avoid corruption. Note that inserting a clause in the contract asserting that the signer is not obligated to provide money can cause corruption as the organization may receive the fund but it claims the opposite.

A *non-binding signature* would be a solution for these scenarios. A signature scheme is *non-binding* if the Signer is able to prove to a Judge the validity of a signature generated by herself, but, she has the ability to disavow her own signatures if she wants (plausible deniability).

One common motivation for using *non-binding signatures* is to provide an initial indication of agreement or intent without committing to a formal contract or legal obligation. For example, in business negotiations, parties may use *non-binding signatures* on a term sheet or letter of intent to indicate agreement on certain key points before moving forward with more formal contract negotiations. Another motivation for using *non-binding signatures* is to facilitate collaboration and communication among parties without creating legal obligations or restrictions. For example, in the context of academic research, researchers may use *non-binding signatures* on collaborative agreements to indicate their willingness to work together on a project without creating formal legal obligations.

One example of such contracts is named *Memorandum of Understanding* (MOU) [39]. An MOU is a document that outlines the terms of a proposed agreement between two or more parties. It is often used in business, government, and other contexts to establish a framework for future negotiations or collaboration. While an MOU is not legally binding, it can be used to demonstrate a commitment to work together and establish a basis for future cooperation. A digital realization of a MOU is strongly motivated.

Beside these use cases and motivating examples, we have found the problem of inventing a non-binding signature as a challenging and non-trivial question. Specifically, as briefed above, the plausible deniability may not be achieved if a signature is publicly verifiable because the Judge can verify it as well using pk. It may seem that the techniques used in the privacy-preserving signatures, for instance Group Signatures [19], Ring Signatures [50], etc, can be deployed to construct a signature schemes that is publicly verifiable and it has the plausible deniability. We emphasize that these techniques hide the identity of the actual signer among a group of signers, but, still everyone is convinced that the signature is generated by a member from a set of users including the actual signer. We are doubtful that such techniques would lead to a signature scheme that is publicly verifiable and it has the plausible deniability property. The philosophical reasoning is that when a public-key associated to a user (or to a group of users) is used to verify a signature publicly, the identities of these users are known to the public. Unless, there is no link between the public keys and the users which makes the signature scheme useless, or the link between the public keys and the users is only known to some designated people (verifiers). The conclusion is that a signature scheme with the plausible deniability probably is likely to be achieved with respect to a designated verifier.

A Designated Verifier Signature (DVS) scheme first proposed in [34] allows a Signer to convince a designated verifier that a signature is generated by himself and the Verifier is not able to transfer the conviction to others, while anyone can still believe that the signature is generated by one of them. A DVS is motivated by applications like signing personal health records, bank transactions, etc, to meet the privacy concerns of the Signer.

A. Our Contribution

We formally define the non-binding property for a designated verifier signature scheme that is *non-transferable* (see Definition 14). A DVS is *non-transferable* if the designated verifier $\mathcal{V}$ is able to perform the signature himself and therefore $\mathcal{V}$ is not able to convince a third party Judge that the Signer $\mathcal{S}$ has signed the message. Our definition is intended to consider all the cheating scenarios. Namely, $\mathcal{S}$ should be able to prove the validity of a signature generated by herself in a court even if $\mathcal{V}$ tries to mislead the Judge. However, $\mathcal{V}$ must be able to prevent $\mathcal{S}$ from convincing the Judge on an invalid signature without disclosing his secret information. And finally, if a valid signature is generated by $\mathcal{V}$ (for any unknown reason), $\mathcal{S}$ should not be able to claim it as her signature.

Then, we propose a designated verifier signature scheme (Protocol 1) that is *non-transferable* and *non-binding* with respect to our definition (Definition 14). Our scheme (Protocol 1) uses an IND-CCA secure key-encapsulation mechanism (KEM), a one-way public-key encryption (PKE), a message authentication code (MAC) and an existential unforgeable signature scheme (Sign). The Verifier $\mathcal{V}$ possesses two pairs $(\mathsf{pk}_v, \mathsf{sk}_v)$ and $(\mathsf{pk}_{\mathrm{Sign}}, \mathsf{sk}_{\mathrm{Sign}})$ generated by PKE and Sign, respectively. The Signer $\mathcal{S}$ possesses a pair $(\mathsf{pk}_s, \mathsf{sk}_s)$ generated by KEM.

The Verifier $\mathcal{V}$ generates a pair (c_s, k_s) using pk_s, computes a signature Σ on (pk_s, c_s) using $\mathsf{sk}_{\mathrm{Sign}}$ and make c_s and σ public. To sign a message m, the Signer checks if Σ is a valid signature for (pk_s, c_s) using $\mathsf{pk}_{\mathrm{Sign}}$. If it is not a valid signature, the Signer aborts. Otherwise, the Signer $\mathcal{S}$ decrypts c_s to get k_s, chooses a random value δ and encrypts it using pk_v to get a ciphertext c_v and finally it sends (m, c_s, c_v) and a tag obtained from MAC using the key $k_s \oplus \delta$. It is clear that $\mathcal{V}$ can verify the signature by decrypting c_v.

Our protocol is *non-transferable* because $\mathcal{V}$ can generate a signature by itself. Namely, he can choose a random value δ and encrypt it using pk_v to get a ciphertext c_v and finally it computes a tag on (m, c_s, c_v) obtained from MAC using the key $k_s \oplus \delta$.

Intuitively, $\mathcal{S}$ can convince a Judge on a signature generated by herself because PKE is one-way and $\mathcal{S}$ should be the generator of the signature if she knows the pre-image of c_v. Note that the signature Σ guarantees that c_s is generated by the veirfier and it is intended to be for the signer (since pk_s is signed along with c_s).

In a high-level, the one-time unforgeability holds since by the IND-CCA security of KEM, (c_s, k_s) is indistinguishable from $(c_s, k^{\$})$ for a randomly chosen $k^{\$}$. Then, if the adversary with the inputs (m, c_s, c_v) and a tag θ obtained from MAC using the key $k^{\$} \oplus \delta$ returns a forgery, it breaks the one-time unforgeability of MAC. (See Theorem 1 for more details.)

In addition, we propose an *instant messaging* protocol in Section VI. In a nutshell, instead of sending the plain message m, the Signer encrypts $m \oplus k_s$ along with a random value δ using pk_v to get a ciphertext c_v. Then it generates a tag θ on (c_s, c_v) using the key $k_s \oplus \delta$. Finally, it sends (c_v, θ) to the Verifier. The message m is hidden from an eavesdropper that is listening to the communication. The Verifier can reply similarly if we add two pairs of keys to the protocol. (See details in the Figure 2.)

Remark on Post-quantum Security. Even though we did not state our main theorem (Theorem 1) for a quantum polynomial-time adversary, if we use a post-quantum secure KEM [11], a post-quantum secure PKE [46] and a post-quantum secure MAC [9][1] in the Protocol 1, we can easily argue the post-quantum security of our signature scheme.

B. Related Works

Undeniable Signature. One may argue that an undeniable signature proposed in [18] might satisfy the non-binding property if the Signer does not participate in the verification. However, not participating in the verification is actually an issue for an undeniable signature since the Signer should not be able to disavow her own signature and this issue has been addressed in later works [12]. In other words, plausible

[1] This work proves the post-quantum security of MAC schemes in the superposition-access model which is a stronger level of security.

deniability is an issue for an undeniable signature but it is wanted in a *non-binding signature*.

Private Contract Signature. In [28, 29], authors introduce a type of signature called *private contract signature* (Definition 1 in [28]). Roughly, these are designated verifier signatures that can be converted into universally-verifiable signatures by either the signing party or a trusted third party appointed by the signing party. Even though one may use the universal verifiability of a *private contract signature* to show the non-binding property, we emphasize that their protocol needs a trusted third party to be executed. In contrast, we desire to achieve the non-binding property for a protocol executed between a Signer and a Verifier. In addition, their protocol (Section 4.1) is based on Diffie-Hellman decision problem and it heavily uses non-interactive proof of knowledge protocols.

Strong Designated Verifier. Vaguely speaking, it guarantees that even when the Verifier is honest and does not generate a fake signature, a third party can not distinguish a signature transcript generated by the Signer from a fake signature [34]. That is, anyone can generate a simulated signature indistinguishable from a valid signature generated by the Signer [32, 53].

Multi-designated Verifiers Signatures. In [41], authors propose a construction of multi-designated verifiers signatures where the signer chooses to sign a message for a fixed numbers of specific designated verifiers. The idea of such a protocol is to produce a signature which has the property that any verifier is convinced that this signature has been done by one member of a set of users, but is not able to determine which one.

Universal Designated-Verifier Signature. A UDVS scheme can function as a standard publicly-verifiable digital signature but has additional functionality which allows any holder of a signature (not necessarily the signer) to designate the signature to any desired designated-verifier (using the verifier's public-key). Given the designated-signature, the designated-verifier can verify that the message was signed by the signer, but is unable to convince anyone else of this fact [40, 49, 55, 56, 61].

Non-Delegatability. Briefly, in a non-delegatable DVS scheme, neither a signer nor a designated verifier can delegate the signing rights to any third party without revealing their secret-keys. This is achieved by the existence of an efficient knowledge extractor that can extract either Signer's secret-key or Verifier's secret-key, when given oracle access to an adversary who can create valid signatures with a high probability [43, 44].

Secure Disavowability: From the Section 1 in [44]: If the DVS scheme has a disavowal protocol, it must be the case that the Signer cannot disavow signatures, given by herself. By this representation from [44], a signature with secure disavowability can not be *non-binding*. However, there is another representation for the disavowability in the Section 6 of [44]: In some other schemes—that we call disavowable—Signer can prove that (a) she signed messages that she really signed, and (b) she has not signed signatures, simulated by the Verifier. This representation is similar to *non-binding* property that we propose in this paper with a difference that the Signer is able to disavow signatures given by herself in a *non-binding* signature.

Malleable Signatures. Some of the signature schemes allow a modification to the signature without effecting the verifiability. Few examples are Homomorphic Signature [35], Sanitizable Signatures [3], Structure-Preserving Signatures on Equivalence Classes [31], etc. We do not find an immediate connection to deploy the malleability in favor of constructing a non-binding signature. For instance a sanitizable signature allows authorized semi-trusted censors to modify – in a limited and controlled fashion – parts of a signed message without interacting with the original signer [3]. It seems that a sanitizable signature gives a flexibility to the signer to change his mind later and this results in a non-binding signature. However, we emphasize that once a pair (m, σ) is out by the signer, anyone can verify that this pair is generated by the signer using the associated public key. It is true that the signer (with the help of the censor) can modify this pair to a new pair (m', σ') that is publicly verifiable, but this does not affect the verifiability of (m, σ) that has produced earlier. In other words, the signer is not able to deny the origin of (m, σ).

II. Preliminaries

In this section, we present necessary preliminaries for our paper. More information regarding the definitions that have presented without a reference can be found in [38]. The function $\mathsf{negl}(\lambda)$ is any non-negative function that is smaller than the inverse of any non-negative polynomial $p(\lambda)$ for sufficiently large λ.

Definition 1. *A public-key encryption scheme* PKE *consists of three polynomial-time (in the security parameter λ) algorithms* $(\mathrm{PKE.Gen}, \mathrm{PKE.Enc}, \mathrm{PKE.Dec})$*, such that:*

1) $\mathrm{PKE.Gen}$*, the key generation algorithm, is a probabilistic algorithm which on input λ outputs a pair of keys,* $(pk, sk) \leftarrow \mathrm{PKE.Gen}(\lambda)$*, called the public-key and the secret-key for the encryption scheme, respectively.*
2) $\mathrm{PKE.Enc}$*, the encryption algorithm, is a probabilistic algorithm which takes as input a public-key pk and a message $m \in \mathtt{MSP}$ and outputs a ciphertext* $c \leftarrow \mathrm{PKE.Enc}_{pk}(m)$*. The message space, $\mathtt{MSP}$, may depend on pk.*
3) $\mathrm{PKE.Dec}$*, the decryption algorithm takes as input a secret-key sk and a ciphertext c and returns a message m. It is required that the decryption algorithm returns the original message with a high probability for every* $(\mathsf{pk}, \mathsf{sk})$ *generated by* $\mathrm{PKE.Gen}(\lambda)$ *and every $m \in \mathtt{MSP}$.*

The algorithm $\mathrm{PKE.Dec}$ *returns* $\perp$ *if a ciphertext* c *is not decryptable.*

Definition 2. *We say a public-key encryption scheme* $\mathrm{PKE} = (\mathrm{PKE.Gen}, \mathrm{PKE.Enc}, \mathrm{PKE.Dec})$ *is one-way if for any* PPT *adversary* $\mathcal{A}$*:*

$$\Pr[\mathcal{A}(\mathsf{pk}, c) = m : (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathrm{PKE.Gen}(\lambda), m \xleftarrow{\$} MSP, c \leftarrow \mathrm{PKE.Enc}_{\mathsf{pk}}(m)] \leq \textsf{\textit{negl}}(\lambda).$$

We define IND-CPA security for a public-key encryption scheme. We present the real-or-random security definition in which the adversary should not be able to distinguish an encryption of a chosen message from an encryption of a random message.

Definition 3. *We say a public-key encryption scheme* $\mathrm{PKE} = (\mathrm{PKE.Gen}, \mathrm{PKE.Enc}, \mathrm{PKE.Dec})$ *is is IND-CPA secure if for any* PPT *adversary* $\mathcal{A}$*:*

$$|\Pr[b = 1 : m \leftarrow \mathcal{A}(\mathsf{pk}), c^* \leftarrow \mathrm{PKE.Enc}_{\mathsf{pk}}(m), b \leftarrow \mathcal{A}(\mathsf{pk}, c^*)] - \Pr[b = 1 : m \leftarrow \mathcal{A}(\mathsf{pk}), m^{\$} \xleftarrow{\$} MSP, c^* \leftarrow \mathrm{PKE.Enc}_{\mathsf{pk}}(m^{\$}), b \leftarrow \mathcal{A}(\mathsf{pk}, c^*)]| \leq \textsf{\textit{negl}}(\lambda),$$

where $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathrm{PKE.Gen}(\lambda)$.

We define a message authentication code scheme below.

Definition 4 (Message Authentication Code (MAC))**.** *A message authentication code* MAC *consists of three (possibly randomized) algorithms* $\mathrm{MAC.\,Gen}$, $\mathrm{MAC.\,Tag}$, $\mathrm{MAC.\,Verif}$*:*

- *The algorithm* $\mathrm{MAC.\,Gen}$ *on input* λ *returns a key* k*.*
- *The algorithm* $\mathrm{MAC.\,Tag}$ *on inputs* m, k *returns a tag* θ*.*
- *The algorithm* $\mathrm{MAC.\,Verif}$ *on inputs* m, k, θ *returns* 1 *(accept) or* 0 *(reject).*

The MAC *should fulfill the correctness property, that is, the* $\mathrm{MAC.\,Verif}_k$ *returns accept on input of* (m, θ) *generated by* $\mathrm{MAC.\,Tag}_k$*.*

Definition 5 (One-time Unforgeability: MAC)**.** *We say a message authentication code* $\mathrm{MAC} = (\mathrm{MAC.\,Gen}, \mathrm{MAC.\,Tag}, \mathrm{MAC.\,Verif})$ *is one-time unforgeable if for any* PPT *adversary* $\mathcal{A}$*, for any message* m *and any* k *generated by* $\mathrm{MAC.\,Gen}$*, the following holds:*

$$\Pr[\mathrm{MAC.\,Verif}_k(m', \theta') = 1 \wedge (m', \theta') \neq (m, \theta) : \theta \leftarrow \mathrm{MAC.\,Tag}(m, k), (m', \theta') \leftarrow \mathcal{A}(m, \theta)] \leq \textsf{\textit{negl}}(\lambda).$$

Definition 6 (Signature Scheme)**.** *A signature scheme* Sign *consists of three (possibly randomized) algorithms* $\mathrm{S.\,Gen}$*,* $\mathrm{S.\,Sign}$*,* $\mathrm{S.\,Verif}$*:*

- *The algorithm* $\mathrm{S.\,Gen}$ *on input* λ *returns two key* pk, sk*.*
- *The algorithm* $\mathrm{S.\,Sign}$ *on inputs* m, sk *returns a signature* σ*.*
- *The algorithm* $\mathrm{S.\,Verif}$ *on inputs* m, pk, σ *returns* 1 *(accept) or* 0 *(reject).*

The Sign *should fulfill the correctness property, that is, the* $\mathrm{S.\,Verif}_{sk}$ *returns accept on input of* (m, σ) *generated by* $\mathrm{S.\,Sign}_{\mathsf{pk}}$ *where* pk, sk *are generated by* $\mathrm{S.\,Gen}$*.*

Definition 7 (Existential Unforgeability)**.** *A signature scheme* $\mathrm{Sign} := (\mathrm{S.\,Gen}, \mathrm{S.\,Sign}, \mathrm{S.\,Verif})$ *is existential unforgeable if for any* PPT *adversary* $\mathcal{A}$*, and any* $(\mathsf{pk}, \mathsf{sk})$ *generated by* $\mathrm{S.\,Gen}$*, the following holds:*

$$\Pr[\mathrm{S.\,Verif}_{\mathsf{pk}}(m, \sigma) = 1 \wedge (m, \sigma) \notin \mathbf{L} : (m, \sigma) \leftarrow \mathcal{A}^{\mathrm{S.Sign}_{\mathsf{sk}}}(\mathbf{L})] \leq \textsf{\textit{negl}}(\lambda),$$

where $\mathbf{L}$ *is a list to store the* $\mathcal{A}$*'s signature queries to* $\mathrm{Sign}_{\mathsf{sk}}$*.*

Definition 8 (Key Encapsulation Mechanism (KEM))**.** *A key encapsulation mechanism* KEM *consists of the following (possibly randomized) algorithms.*

- *A key generating algorithm* $\mathrm{KEM.\,Gen}$ *that on input* λ *returns a pair key* $(\mathsf{pk}, \mathsf{sk})$*.*
- *An encryption algorithm* $\mathrm{KEM.\,Enc}$ *that on input* λ *and a public-key* pk*, outputs a pair* (c, k)*, where* k *is a key and* c *is a ciphertext. (The algorithm* $\mathrm{KEM.\,Enc}$ *may need a random input* r *in each execution.)*
- *A decryption algorithm* $\mathrm{KEM.\,Dec}$ *that on input* λ*, a secret-key* sk*, a ciphertext* c*, outputs either a key* k *or* $\perp$*.*

The key encapsulation mechanism has to have the correctness property, that is, $\mathrm{KEM.\,Dec}_{\mathsf{sk}}(c) = k$ *with a high probability when* (c, k) *is obtained from* $\mathrm{KEM.\,Enc}$*.*

We say a key encapsulation mechanism is strongly correct if a PPT adversary is not able to maliciously generate a ciphertext that violates the correctness property.

Definition 9 (Strongly Correct KEM)**.** *A key encapsulation mechanism* $\mathrm{KEM} = (\mathrm{KEM.\,Gen}, \mathrm{KEM.\,Enc}, \mathrm{KEM.\,Dec})$ *is strongly correct if for any* PPT *adversary* $\mathcal{A}$*,*

$$|\Pr[k^* \neq k \ \wedge k^* \neq \perp \ \wedge \mathrm{KEM.\,Enc}_{\mathsf{pk}}(r) = (c, k) : (c, k, r) \leftarrow \mathcal{A}(\mathsf{pk}), k^* \leftarrow \mathrm{KEM.\,Dec}_{\mathsf{sk}}(c)] \leq \textsf{\textit{negl}}(\lambda),$$

where $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathrm{KEM.\,Gen}(\lambda)$*.*

Informally, a KEM is IND-CCA secure if the adversary is not able to distinguish between a pair (c^*, k^*) generated by $\mathrm{KEM.\,Enc}$ and $(c^*, k^{\$})$ (where $k^{\$}$ is chosen randomly from the key space), even with access to the decryption oracle (except for c^*.)

Definition 10 (IND-CCA)**.** *We say* $\mathrm{KEM} = (\mathrm{KEM.\,Gen}, \mathrm{KEM.\,Enc}, \mathrm{KEM.\,Dec})$ *is IND-CCA secure if for any* PPT *adversary* $\mathcal{A}$*,*

$$|\Pr[b = 1 : (c^*, k^*) \leftarrow \mathrm{KEM.\,Enc}(\mathsf{pk}), b \leftarrow \mathcal{A}^{\mathrm{KEM.Dec}_{c \neq c^*}}(c^*, k^*)] - \Pr[b = 1 : (c^*, k^*) \leftarrow \mathrm{KEM.\,Enc}(\mathsf{pk}), k^{\$} \xleftarrow{\$} \mathcal{K}, b \leftarrow \mathcal{A}^{\mathrm{KEM.Dec}_{c \neq c^*}}(c^*, k^{\$})]| \leq \textsf{\textit{negl}}(\lambda),$$

where $\mathcal{K}$ *is the set of all possible keys,* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathrm{KEM.\,Gen}(\lambda)$ *and* $\mathrm{KEM.\,Dec}_{c \neq c^*}$ *is the same as* $\mathrm{KEM.\,Dec}$ *except it does not decrypt* c^* *for the adversary.*

In the following, we define a designated verifier signature scheme.

Definition 11 (Designated Verifier Signature [44])**.** *A designated verifier signature scheme* DVS *with security parameter* λ *is defined by the following probabilistic algorithms:*

- *An algorithm* $\mathrm{DVS.\,SGen}$ *which takes* λ *as input, and outputs a pair of keys* $(\mathsf{pk}_s, \mathsf{sk}_s)$*.*
- *An algorithm* $\mathrm{DVS.\,VGen}$ *which takes* λ *as input, and outputs a pair of keys* $(\mathsf{pk}_v, \mathsf{sk}_v)$*.*
- *An algorithm* $\mathrm{DVS.\,VSetup}$ *which takes* λ *and* pk_s *as input and returns a public parameter* c_s *and a secret parameter* k_s*.*
- *A designated verifier signing algorithm* $\mathrm{DVS.\,Sign}$ *which takes a message* m*, a signing secret-key* sk_s*, a verifying public-key* pk_v *and a public parameter* c_s *as inputs and returns a value* σ*.*
- *A designated verifying algorithm* $\mathrm{DVs.\,Verif}$ *which takes a bit string* σ*, a signing public-key* pk_s*, a verifying secret-key* sk_v *and a pair public and secret parameter* (c_s, k_s) *as inputs, and returns a bit* b *(reject or accept).*
- *A* PPT *algorithm* Sim *that on inputs* $m, c_s, \mathsf{sk}_v, \mathsf{pk}_s$ *returns a value* σ*.*
- *A* PPT *algorithm* Judge *that on the inputs of the public parameters and some inputs from the Signer and Verifier either accepts (returns 1) or rejects (returns 0). Note that the description of this algorithm depends on the scheme.*

A designated verifier signature scheme DVS *should fulfill the correctness property in which* $\mathrm{DVs.\,Verif}$ *returns accept on a signature generated by* $\mathrm{DVS.\,Sign}$*.*

Intuitively, a DVS is one-time unforgeabile if any PPT adversary given a signature and public parameters (including c_s) is not able to forge a new signature.

Definition 12 (One-time Unforgeability (DVS))**.** *A designated verifier signature is one-time unforgeable if for any* PPT *adversary* $\mathcal{A}$*, for any message* m *and any* $(\mathsf{pk}_s, \mathsf{sk}_s)$ *and* $(\mathsf{pk}_v, \mathsf{sk}_v)$ *generated by* $\mathrm{DVS.\,SGen}$ *and* $\mathrm{DVS.\,VGen}$ *respectively, the following holds:*

$$\Pr[\mathrm{DVs.\,Verif}(m', c_s, \sigma') = 1 \wedge (m', \sigma') \neq (m, \sigma) : \\ (c_s, k_s) \leftarrow \mathrm{DVS.\,VSetup}(\mathsf{pk}_s), \sigma \leftarrow \mathrm{DVS.\,Sign}(m, \mathsf{sk}_s, c_s, \mathsf{pk}_v), \\ (m', \sigma') \leftarrow \mathcal{A}(m, c_s, \sigma, \mathsf{pk}_s, \mathsf{pk}_v)] \leq \textsf{\textit{negl}}(\lambda).$$

Informally speaking, a designated verifier signature scheme is *non-transferable* if the Verifier is able to produce a signature indistinguishable from a signature outputted by the Signer. This is shown by the existence of a PPT algorithm Sim that given the secret-key of the Verifier can simulate a valid signature indistinguishable from a signature outputted by the Signer.

Definition 13 (Non-transferability [44])**.** *We say a designated verifier signature scheme is non-transferable if there exists a* PPT *algorithm* Sim *such that for any* PPT *distinguisher* $\mathcal{D}$*, for any message* m *and any* $(\mathsf{pk}_s, \mathsf{sk}_s)$ *and* $(\mathsf{pk}_v, \mathsf{sk}_v)$ *generated by* $\mathrm{DVS.\,SGen}$ *and* $\mathrm{DVS.\,VGen}$ *respectively:*

$$|\Pr[b = 1 : (c_s, k_s) \leftarrow \mathrm{DVS.\,VSetup}(\mathsf{pk}_s), \\ \sigma \leftarrow \mathrm{DVS.\,Sign}(m, \mathsf{sk}_s, c_s, \mathsf{pk}_v), b \leftarrow \mathcal{D}(m, \sigma, c_s, \mathsf{pk}_s, \mathsf{pk}_v)] - \\ \Pr[b = 1 : \sigma \leftarrow \mathsf{Sim}(m, (c_s, k_s), \mathsf{sk}_v, \mathsf{pk}_s), \\ b \leftarrow \mathcal{D}(m, \sigma, c_s, \mathsf{pk}_s, \mathsf{pk}_v)]| \leq \textsf{\textit{negl}}(\lambda).$$

III. Non-binding Signature

The non-transferability property of a designated signature scheme guarantees that a designated verifier $\mathcal{V}$ is not able to convince a third party that a signer $\mathcal{S}$ has indeed performed the signature without revealing its secret information. This holds since a designated verifier is able to produce a signature that is indistinguishable from the Signer's signature. But this property causes a huge drawback. Namely, what if $\mathcal{V}$ denies the validity of a signature produced honestly by $\mathcal{S}$. Imagine that the Signer $\mathcal{S}$ is an employee that has signed a work contract with an employer $\mathcal{V}$ digitally using a non-transferable DVS. Then, the employer $\mathcal{V}$ is able to deny $\mathcal{S}$'s signature. To remedy this, we need a DVS scheme in which the Signer is able to legally prove to a Judge (without revealing its secret information) that she is the Signer of the work contract in case of $\mathcal{V}$'s denial.

We say a DVS is a *non-binding signature* if the Signer is able to prove to a Judge the validity of a signature of her own, but, she has the ability to disavow her own signatures if she wants (plausible deniability). The non-transferability property is helping to construct a *non-binding signature* since the Signer can simply claim that the signature is generated by the Verifier himself.

In the following, we formally define a *non-binding signature*. In the definition, the Judge is a PPT algorithm that on some inputs from the participants (Signer and Verifier) either accepts (returns 1) or rejects (returns 0). Note that the description of this algorithm depends on the scheme.

The Item 1 in the Definition 14 guarantees that the Signer is able to prove to a Judge the validity of a signature of her own. In the definition, the Signer should be able to provide some extra information $\mathbf{st_s}$ to convince the Judge. (It is desired that $\mathbf{st_s}$ does not contain the secret information of the Signer.) Even if the Verifier may generates c_s maliciously and is able to provide the Judge with some information $\mathbf{st_v}$ to mislead the Judge.

The Signer may try to convince the Judge on an invalid signature in the court. Of course the Verifier may be able to dispute her claim, however, we do not want that the Verifier displays his secret information (used in $\mathrm{DVs.\,Verif}$) to the Judge. We add the Item 2 to the definition and we denote the extra information provided by the Verifier with $\mathbf{st_v}$.

Finally, we prevent the Signer to convince the Judge on a signature outputted by the Verifier in the Item 3. The third case may be unlikely in real-world scenarios and we add it to the definition for completeness.

In a nutshell, the Judge on inputs $\mathbf{st_s}$, $\mathbf{st_v}$, m, σ, c_s, pk_s and pk_v returns a bit b. Here, σ is either generated by the

Signer or Sim. And the information $\mathbf{st_s}$ and $\mathbf{st_v}$ are provided by the Signer and the Verifier, respectively.

In each case, we differentiate the malicious entity with a star $*$.

Definition 14 (Non-Binding)**.** *A non-transferable DVS is non-binding if for any message m and any $(\mathsf{pk}_s, \mathsf{sk}_s)$ and $(\mathsf{pk}_v, \mathsf{sk}_v)$ generated by* $\mathrm{DVS.\,SGen}$ *and* $\mathrm{DVS.\,VGen}$ *respectively, there exists a* PPT *algorithm* Judge *such that the following three cases hold:*

1) *The Signer $\mathcal{S}$ is able to convince the* Judge *on the validity of her own signature without revealing its secret:*

$$\begin{aligned}\Pr[b = 1 \wedge b' = 1 : (c_s, k_s) &\leftarrow \mathrm{DVS.\,VSetup}^*,\\ (\mathbf{st_s}, \sigma) &\leftarrow \mathcal{S}(\mathsf{pk}_s, \mathsf{sk}_s, c_s, m, \mathsf{pk}_v),\\ b' &\leftarrow \mathrm{DVs.\,Verif}(m, \sigma, c_s, k_s, \mathsf{sk}_v),\\ \mathbf{st_v} &\leftarrow \mathcal{V}^*(\mathsf{pk}_s, c_s, k_s, \mathsf{sk}_v, m, \sigma)\\ b \leftarrow \mathsf{Judge}(\mathbf{st_s}, \mathbf{st_v}, m, \sigma, c_s, \mathsf{pk}_s, \mathsf{pk}_v)] &\geq 1 - \textit{negl}(\lambda).\end{aligned}$$

2) *The Signer $\mathcal{S}$ should not be able to convince the* Judge *on an invalid signature:*

$$\begin{aligned}\Pr[b = 1 \wedge b' = 0 : (c_s, k_s) &\leftarrow \mathrm{DVS.\,VSetup},\\ (\mathbf{st_s}, \sigma) &\leftarrow \mathcal{S}^*(\mathsf{pk}_s, \mathsf{sk}_s, c_s, m, \mathsf{pk}_v),\\ b' &\leftarrow \mathrm{DVs.\,Verif}(m, \sigma, c_s, k_s, \mathsf{sk}_v),\\ \mathbf{st_v} &\leftarrow \mathcal{V}(\mathsf{pk}_s, c_s, k_s, \mathsf{sk}_v, m, \sigma)\\ b \leftarrow \mathsf{Judge}(\mathbf{st_s}, \mathbf{st_v}, m, \sigma, c_s, \mathsf{pk}_s, \mathsf{pk}_v)] &\leq \textit{negl}(\lambda).\end{aligned}$$

3) *The Signer $\mathcal{S}$ should not be able to convince the* Judge *on a signature outputted by the verifier:*

$$\begin{aligned}\Pr[b = 1 : (c_s, k_s) &\leftarrow \mathrm{DVS.\,VSetup},\\ \sigma &\leftarrow \mathcal{V}(m, c_s, \mathsf{sk}_v, \mathsf{pk}_s),\\ \mathbf{st_s} &\leftarrow \mathcal{S}^*(\mathsf{pk}_s, \mathsf{sk}_s, m, c_s, \sigma, \mathsf{pk}_v),\\ \mathbf{st_v} &\leftarrow \mathcal{V}(\mathsf{pk}_s, c_s, \mathsf{sk}_v, m, \sigma)\\ b \leftarrow \mathsf{Judge}(\mathbf{st_s}, \mathbf{st_v}, m, \sigma, c_s, \mathsf{pk}_s, \mathsf{pk}_v)] &\leq \textit{negl}(\lambda).\end{aligned}$$

IV. Protocol

In this section, we define our protocol that is constructed from a key-encapsulation mechanism, a public-key encryption scheme and a message authentication code. We emphasize that using post-quantum version of these constructions will lead to a post-quantum DVS scheme.

Protocol 1 (Figure 1)**.** *The protocol uses a key-encapsulation mechanism* $\mathrm{KEM} := (\mathrm{KEM.\,Gen}, \mathrm{KEM.\,Enc}, \mathrm{KEM.\,Dec})$*, a public-key encryption scheme* $\mathrm{PKE} := (\mathrm{PKE.Gen}, \mathrm{PKE.Enc}, \mathrm{PKE.Dec})$*, a signature scheme* $\mathrm{Sign} := (\mathrm{S.\,Gen}, \mathrm{S.\,Sign}, \mathrm{S.\,Verif})$*, and a message authentication code* $\mathrm{MAC} := (\mathrm{MAC.\,Gen}, \mathrm{MAC.\,Tag}, \mathrm{MAC.\,Verif})$.

1) *The Signer invokes* $\mathrm{KEM.\,Gen}$ *on input λ to get a pair $(\mathsf{pk}_s, \mathsf{sk}_s)$. It makes pk_s public.*
2) *The Verifier invokes* $\mathrm{PKE.Gen}$ *and* $\mathrm{S.\,Gen}$ *on input λ to get a pair $(\mathsf{pk}_v, \mathsf{sk}_v)$ and $(\mathsf{pk}_{\mathrm{Sign}}, \mathsf{sk}_{\mathrm{Sign}})$. Then, it runs* $\mathrm{KEM.\,Enc}$ *on the input pk_s to get a pair (k_s, c_s). It makes $\mathsf{pk}_{\mathrm{Sign}}, \mathsf{pk}_v, c_s$ and $\Sigma := \mathrm{Sign}_{\mathsf{sk}_{sign}}(\mathsf{pk}_v, c_s)$ public.*
3) *The Signer checks if Σ is a valid signature for pk_v, c_s using $\mathsf{pk}_{\mathrm{Sign}}$. If no, it aborts. Otherwise, it invokes* $\mathrm{KEM.\,Dec}$ *on the inputs sk_s and c_s to get a value k_s. Then it executes* $\mathrm{PKE.Enc}$ *on the inputs pk_v and a random value δ to get a ciphertext c_v. To sign a message m, it computes $\theta = \mathrm{MAC}_{\delta \oplus k_s}(m, c_v, c_s)$ and sends $m, \sigma = (c_v, \theta)$ to the Verifier.*
4) *The Verifier invokes* $\mathrm{PKE.Dec}$ *on the inputs sk_v and c_v to get a value δ'. Then it returns the output of* $\mathrm{MAC.\,Verif}_{\delta' \oplus k_s}$ *on inputs (m, c_v, c_s) and θ.*
5) *The* PPT *algorithm* Judge *first checks if Σ is a valid signature for (pk_s, c_s). If this signature is not valid, it aborts and return 0. Otherwise, it is given $\mathbf{st_s} = (k_m, \delta, r_s)$ checks if $\mathrm{PKE.Enc}_{\mathsf{pk}_v}(\delta; r_s) = c_v$ and if the verification of* MAC *with the key k_m holds true. In case one of these two checks fails, the* Judge *aborts and returns 0 (reject). Otherwise, the* Judge *given $\mathbf{st_v} = (k_s, r_v)$, (if the verifier does not provide any information, $(k_s, r_v) := (0, 0)$ by* Judge*) verifies if $\mathrm{KEM.\,Enc}_{\mathsf{pk}_s}(r_v) = (c_s, k_s)$ and $k_s \neq k_m \oplus \delta$. If both checks holds true, it returns 0, otherwise, it returns 1 (accept).*

```
Setup:
$(\mathsf{pk}_{\mathrm{Sign}}, \mathsf{sk}_{\mathrm{Sign}}) \leftarrow \mathrm{S.\,Gen}(\lambda)$
$(\mathsf{pk}_s, \mathsf{sk}_s) \leftarrow \mathrm{KEM.\,Gen}(\lambda)$,
$(\mathsf{pk}_v, \mathsf{sk}_v) \leftarrow \mathrm{PKE.Gen}(\lambda)$
$r_v \overset{\$}{\leftarrow} \mathtt{R}$,
$\mathrm{KEM.\,Enc}_{\mathsf{pk}_s}(r_v) = (k_s, c_s)$
$\Sigma := \mathrm{Sign}_{\mathsf{sk}_{\mathrm{Sign}}(\mathsf{pk}_s, c_s)}$

Public Parameters: $(\mathsf{pk}_s, \mathsf{pk}_{\mathrm{Sign}}, \mathsf{pk}_v, \Sigma, c_s)$

Signer$(m, \mathsf{sk}_s)$                                    Verifier$(\mathsf{sk}_v, k_s)$
$b \leftarrow \mathrm{S.\,Verif}_{\mathsf{sk}_{\mathrm{Sign}}}(\mathsf{pk}_s, c_s, \Sigma)$
if $b = 0$ abort. Otherwise:
$k_s \leftarrow \mathrm{KEM.\,Dec}(\mathsf{sk}_s, c_s)$
$\delta \overset{\$}{\leftarrow} \mathtt{MSP}, r_s \overset{\$}{\leftarrow} \mathtt{R}$
$c_v = \mathrm{PKE.Enc}(\mathsf{pk}_v, \delta; r_s)$
$\theta = \mathrm{MAC}_{\delta \oplus k_s}(m, c_v, c_s)$

                 $\xrightarrow{m,\ \sigma = (c_v, \theta)}$

                              $\delta \leftarrow \mathrm{PKE.Dec}(\mathsf{sk}_v, c_v)$
                   $b \leftarrow \mathrm{MAC.\,Verif}_{\delta \oplus k_s}(\theta, m, c_v, c_s)$
                                                    RETURN $b$
```

Fig. 1. Non-Binding Designated Verifier Signature.

Remark. Since c_s is public and a signature σ is generated with respect to (k_s, c_s), our scheme is not a strong designated verifier signature.

Theorem 1. *The signature scheme in Protocol 1 is one-time unforgeable, non-transferable and non-binding if KEM is IND-CCA secure, PKE is one-way secure MAC is one-time unforgeable and* Sign *is existential unforgeable.*

Proof. Before getting to the details of the proof, we remark that the purpose of using a signature scheme Sign is to make the protocol non-interactive and to make sure that the setup phase is generated by the intended verifier. Assuming that the signature Σ is a valid signature on (c_s, pk_s), we continue to show the rest of the properties.

Non-transferability. We construct a simulator Sim that given $(k_s, c_s), \mathsf{sk}_v$ can simulate a signature perfectly. To sign a message m, the simulator executes $\mathrm{PKE.Enc}$ on the inputs pk_v and a random value δ to get a ciphertext c_v. Then, it returns (m, c_v, c_s) and $\theta = \mathrm{MAC}_{\delta \oplus k_s}(m, c_v, c_s)$ as the signature on m. It is clear that the distribution of the Sim's signature on m is equal to the one outputted in Protocol 1.

Non-binding: We show that the Protocol 1 is *non-binding*. In the following, we illustrate the PPT algorithm Judge more and show how it satisfies the non-binding property. The algorithm Judge on inputs m, c_s, pk_s, pk_v and $\sigma = (c_v, \theta)$ inquires from the Signer and the Verifier information $\mathbf{st_s}$ and $\mathbf{st_v}$, respectively. Recall that in the definition, there are three different cases and in each case it is determined which party is malicious. Note that when the Verifier is malicious, potentially, his input to the Judge is dishonestly generated or even no information is provided. Without loss of generality, we assume that in this case a malicious verifier provides malevolently information to mislead the Judge. This loss of generality comes from the description of the Judge algorithm below that is convinced by an honest signer without the need of the verfiier's participation. In other words, generating no information is equivalent to assuming that the input of the verifier is a zero bitstring (with a proper length).

An honest Signer has to provide the pre-image of c_v (both δ and r_s) and a verification key for MAC, that is, $\mathbf{st_s} = (k_m, \delta, r_s)$. And an honest Verifier has to provide the pre-image of c_s (both k'_s and r_v), that is, $\mathbf{st_v} = (k_s, r_v)$.

Then the Judge given $\mathbf{st_s} = (k_m, \delta, r_s)$ checks if $\mathrm{PKE.Enc}_{\mathsf{pk}_v}(\delta; r_s) = c_v$ and if the verification of MAC with the key k_m holds true. In case one of these two checks fails, the Judge aborts and returns 0 (reject). (At this point, if the Judge does not abort, he is convinced that the signer is honest unless the verifier maliciously alters his opinion.) Otherwise, the Judge given $\mathbf{st_v} = (k_s, r_v)$, (if the verifier does not provide any information, $(k_s, r_v) := (0, 0)$) verifies if $\mathrm{KEM.\,Enc}_{\mathsf{pk}_s}(r_v) = (c_s, k_s)$ and $k_s \neq k_m \oplus \delta$. If both checks holds true, it returns 0, otherwise, it returns 1 (accept).

1) In this item, we show that the Signer can convince the Judge on a signature generated by herself. For a signature $\sigma = (c_v, \theta = \mathrm{MAC}_{\delta \oplus k_s}(m, c_s, c_v))$ generated honestly by the Signer, the information $\mathbf{st_s} = (k_s \oplus \delta, \delta, r_s)$ can convince the Judge with a probability close to $1 - \mathsf{negl}(\lambda)$. Namely, the Judge checks if $\mathrm{PKE.Enc}_{\mathsf{pk}_v}(\delta; r_s) = c_v$ and if the verification of MAC with the key $\delta \oplus k_s$ holds true. Note that $\mathcal{V}$ can mislead the Judge if he finds a collision on c_s with a different k'_s. That is, if $\mathcal{V}$ is able to provide $\mathbf{st_v} = (k'_s, r'_v)$ such that $\mathrm{KEM.\,Enc}_{\mathsf{pk}_s}(r'_v) = (c_s, k'_s)$. But this holds with a negligible probability by the strongly correctness of KEM.
2) In this item, we show that the Signer is not able to convince the Judge on an invalid signature $\sigma = (c_v, \theta)$ on (m, c_s). Let $\mathbf{st_s} = (k_m, \delta, r_s)$ be the information provided by the Signer. Let us assume that $\mathrm{PKE.Enc}_{\mathsf{pk}_v}(\delta; r_s) = c_v$ and the verification of MAC with the key k_m holds true. (Otherwise, the Judge rejects the Signer's claim, anyway.) Note that c_s is generated by the honest Verifier, therefore, he can provide $\mathbf{st_v} = (k_s, r_v)$ such that $\mathrm{KEM.\,Enc}_{\mathsf{pk}_s}(r_v) = (c_s, k_s)$. Now since a signature $\sigma = (c_v, \theta)$ on (m, c_s) is not valid, the algorithm $\mathrm{DVs.\,Verif}$ returns 0. That is, the MAC verification of θ with the key $k_s \oplus \delta$ fails. Therefore, k_m should not be equal to $\delta \oplus k_s$ and consequently the Judge returns 0.
3) For a valid signature $\sigma = (c_v, \theta = \mathrm{MAC}_{\delta \oplus k_s}(m, c_s, c_v))$ generated by Sim, $\mathcal{S}$ is not able to convince the Judge by the one-wayness of PKE. ($\mathcal{S}$ is not able to return the pre-image of c_v.)

One-time Unforgeability. Let assume that the signature $\sigma = (c_v, \theta)$ where $\theta = \mathrm{MAC}_{\delta \oplus k_s}(m, c_s, c_v)$ is generated (honestly) by the Sign algorithm. We show that any PPT adversary $\mathcal{A}$ given m, c_s, c_v and θ is not able to forge a new signature. That is, the adversary $\mathcal{A}$ is not able to return a valid signature (m', c_s, c'_v, θ') where $(m', c'_v, \theta') \neq (m, c_v, \theta)$.

Let assume that a PPT adversary $\mathcal{A}$ is able to break the one-time unforgeability of the scheme with a non-negligible probability ϵ. We construct a reduction adversary $\mathcal{B}$ that breaks the IND-CCA security of KEM under the assumption that MAC is secure.[2]

The reduction adversary $\mathcal{B}$ when giving a challenge ciphertext (c^*_s, k^*_s), it runs $\mathrm{PKE.Gen}$ on inputs λ to get a pair $(\mathsf{pk}_v, \mathsf{sk}_v)$, it chooses a random value δ and computes $c_v \leftarrow \mathrm{PKE.Enc}_{\mathsf{pk}_v}(\delta)$ and $\theta^* = \mathrm{MAC}_{\delta \oplus k^*_s}(m, c^*_s, c_v)$ for a message m. Then, it runs $\mathcal{A}$ with the inputs $\lambda, \mathsf{pk}_s, \mathsf{pk}_v, m, c^*_s, c_v$ and θ^*. When the adversary $\mathcal{A}$ returns an output m', c^*_s, c'_v and θ', it runs the verification of MAC on $(m', c^*_s, c'_v, \theta')$ with the key $\delta \oplus k^*_s$. If the verification of MAC returns accept, $\mathcal{B}$ returns 1, otherwise it returns 0.

Note that when (c^*_s, k^*_s) is generated by $\mathrm{KEM.\,Enc}$, the values c^*_s, c_v, θ^* are generated as $\mathcal{A}$ expects, therefore, $\mathcal{B}$ returns 1 with the probability ϵ. However, when k^*_s is a randomly chosen key, c^*_s and k^*_s are irrelevant. This means that $k^*_s \oplus \delta$ is a random value as well that is only used to execute the MAC. Therefore, if the adversary returns a forgery m', c^*_s, c'_v, θ', it breaks the one-time unforgeability of MAC. Since MAC

[2] $p_1 \wedge p_2 \rightarrow q \iff \neg q \rightarrow \neg p_1 \vee \neg p_2 \iff \neg q \wedge p_1 \rightarrow \neg p_2$ where p_1, p_2 and q represent the security of MAC, KEM and DVS respectively in the theorem.

is one-time unforgeable, $\mathcal{B}$ returns 1 only with a negligible probability in this case and this finishes the proof. □

A. Discussion

We discuss some crucial points in favor of our signature scheme based on some scientific critics and reviews that the article has received in the previous submission.

- It may seem that anyone can generate a valid pair of message and signature using our scheme. Even though, the protocol allows any two parties communicate using their public keys, we emphasize that in the setup phase, the verifier generates a pair (k_s, c_s) using the public-key pk_s and makes c_s public. In addition, it publishes a signature Σ on these values. Then a valid signature is generated with respect to c_s. Therefore, a third party is not able to generate a valid signature without knowing k_s.
- **Connection with Identification Schemes.** Our protocol may seem to be an interactive protocol that is reminiscent of an identification scheme [26]. This is arguable since the verifier needs to generate (k_s, c_s) and then to broadcast $((c_s, \mathsf{pk}_s), \Sigma)$ over a public channel. In contrast, the most of identification schemes are in the form of a three-round protocol with three messages, commit, challenge and response. Finally, in our protocol, the signer identifies itself to the verifier (authenticity) and signs a message, but, an identification scheme is only used to identify a party. In addition, the anonymity of the signer is preserved against an external observer since the transcript of the protocol can be generated by the verifier.
- We emphasize that the one-time unforgeability is defined and proved with respect to the value c_s that is not costly to regenerate. In other words, the verifier and the signer do not need to generate new pairs $(\mathsf{sk}_s, \mathsf{pk}_s)$ and $(\mathsf{sk}_v, \mathsf{pk}_v)$ to sign a new message. Therefore, the one-time unforgeability is not limiting in our protocol.

V. Comparison

In this article, we investigate the non-binding property for a signature scheme that has not been introduced or studied in the literature so far. Beside that, our design approach is different from the previous DVS schemes and uses known cryptographic primitives.

Most of the previous DVS schemes are based on Discrete Logarithm Assumption [32, 40, 40, 41, 43, 44, 49, 53, 55, 56, 61] and are not post-quantum secure. Few Lattice-based DVS schemes are available [42, 45, 63]. A DVS based on Isogeny-based assumptions has been presented in [57] and later it has been concluded insecure in [27] by key reuse attacks.

In the Table I, we present few existing designated verifier signature schemes. Then we discuss which of them satisfy the Definition 14 (the *non-binding* definition) introduced in this paper. Since our definition is stated in the standard model, the schemes that are constructed in the random oracle model are not eligible to be verified by the Definition 14.[3] In addition, there exist signature and encryption schemes that are secure in the random oracle model, but for which any implementation of the random oracle results in insecure schemes [15]. Even though these separation examples are artificially invented, the cryptographic community prefer to use the random oracle model when there is no provable-secure construction in the standard model, or when the use of the random oracle model significantly improves the efficiency. For this reasons, we do not compare the protocols in the random oracle model with ours.

Protocol in [32]. In the SDVS protocol in [32], the Signer chooses a random value x and makes $\mathsf{pk}_s := g^x$ public. The Verifier chooses a random value y and makes $\mathsf{pk}_v := g^y$ public. To sign a message m, the Signer computes $k := \mathsf{pk}_v^x$ and sends $PRF_k(m)$ where PRF is a pseudo-random function. We argue that this protocol does not satisfy the *non-binding* property. In more details, the Signer on the input $\sigma := (m, PRF_k(m))$, is not able to provide convincing information $\mathbf{st_s}$ to the Judge that σ is generated by her. The reason is that since $\mathsf{pk}_s = g^x$ is public, the verifier is able to compute the key $k = g^{xy}$ as well and generate $\sigma := (m, PRF_k(m))$ for any message m. Therefore, if the Verifier claims that $(m, PRF_k(m))$ is generated by him, there is no way that the Signer can oppose this claim. In other words, there is no algorithm Judge that satisfies the non-binding property.

Protocol in [45]. The key generating algorithm of the SDVS protocol in [45] generates two pairs of keys $(\mathsf{pk}_s, \mathsf{sk}_s) := ((\mathbf{A}_s, \mathsf{pk}^{(s)}), (\mathbf{T}_s, \mathsf{sk}^{(s)}))$ and $(\mathsf{pk}_v, \mathsf{sk}_v) := ((\mathbf{A}_v, \mathsf{pk}^{(s)}), (\mathbf{T}_v, \mathsf{sk}^{(v)}))$ in which $\mathbf{A}_s$ and $\mathbf{A}_v$ are some matrices with the corresponding trapdoor matrices $\mathbf{T}_s$ and $\mathbf{T}_v$, respectively. And $(\mathsf{pk}^{(s)}, \mathsf{sk}^{(s)})$ and $(\mathsf{pk}^{(v)}, \mathsf{sk}^{(v)})$ are two pairs of keys generated by the key generating algorithm of a public-key encryption scheme. In order to sign a message m, the Signer on inputs $\mathsf{sk}_s, \mathsf{pk}_s, \mathsf{pk}_v$ chooses a randomness r and outputs a vector $\sigma = (\mathbf{v_s}, \mathbf{v_v}, \mathbf{c_s}, \mathbf{c_v})$ with the components are explained in the coming lines. The vector $\mathbf{v_s}$ is drawn from a distribution that depends on $H(m, r)$ (H is a hash function) and $\mathbf{A}_s$ while $\mathbf{v_v}$ is drawn from a distribution using a specific sampling algorithm. The values $\mathbf{c_s}$ and $\mathbf{c_v}$ are the encryption of r with the keys $\mathsf{pk}^{(s)}$ and $\mathsf{pk}^{(v)}$, respectively.

We argue that this protocol does not satisfies the *non-binding* property. Our reasoning is that the randomness r is encrypted with both keys $\mathsf{pk}^{(s)}$ and $\mathsf{pk}^{(v)}$. Consequently, the knowledge of r would not convince the Judge since both the Signer and the Verifier can obtain the randomness r from σ. The other two vectors $\mathbf{v_s}, \mathbf{v_v}$ are drawn from some

[3] Since in the random oracle model (ROM) all parties including the Judge have access to the random oracle, a non-binding definition in ROM would not be the same at the Definition 14. Furthermore, the security analysis in the quantum random oracle model [8] is needed to show the post-quantum security.

TABLE I

COMPARISON.

scheme	type	non-transfer	assumption	standard model	non-binding
[32]	SDVS	✓	DDH	✓	×
[32]	SDVS	✓	DL & GDH[4]	×	–
[44]	DVS	✓	DDH	×	–
[53]	SDVS	✓	DL	×	–
[40]	SDVS	✓	(C-D-G)BDH[5]	×	–
[14]	SDVS	✓	R-SIS	×	–
[45]	SDVS	✓	LWE & SIS	✓	×
Ours	DVS	✓	any	✓	✓

distributions in which the Signer may not be able to convince the Judge that she has invoked the sampling algorithms.

VI. APPLICATION

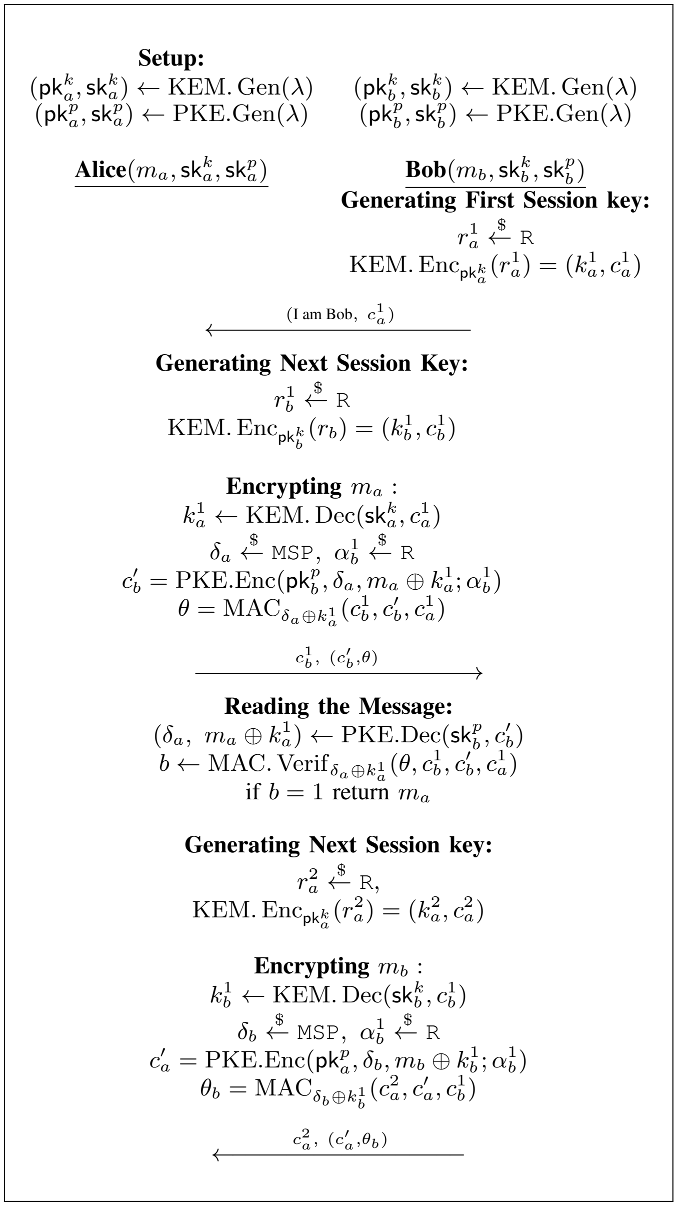
In this section, we discuss how our techniques and approach in this paper is useful to make a confidential and authentic online communication channel.

Note that the *off-the-record messaging* protocol in [10] uses Diffie-Hellman key exchange protocol. An eavesdropper Eve can store all the communication between Alice and Bob and later when a large-scale quantum computer is available she can decrypt all the messages exchanged between them using the Shor's algorithm [54].

We sketch a (post-quantum secure) *instant messaging* protocol obtained by modifying the Protocol 1. Vaguely speaking, the sender (the Signer in the Protocol 1) encrypts the message m XOR with k_s along with the value δ using an IND-CPA secure public encryption scheme resulting in a value c_v. Then, it sends this value c_v and a tag θ on it using the MAC scheme to the receiver. (In other words, the Signer does not send the plain message anymore.) Note that, if we add two more pair of keys to the protocol, the receiver can reply similarly. (See Figure 2.) We use an IND-CPA secure public encryption scheme to be sure that neither the message nor the session key would leak to Eve.

We depict a conversation between Alice and Bob in Figure 2. The first messages (I am Bob, c_a^1) is sent by Bob to share the first session key. We do not add a signature to the message because a man-in-the-middle attacker Eve can not gain by changing the content of the message. In more details, if Eve changes the value c_a^1 or the identity of Bob, neither she nor Bob can obtain the message sent by Alice later. This is considered a disruption attack and nothing more. And obviously, changing both values is only a disruption attack as well. The rest of session keys in Figure 2 are included inside of MAC to be sewn with the message.

As discussed by the designers of OTR in [10], if Alice encrypts her messages to Bob's public encryption key, and signs them with her own private signature key, an eavesdropper Eve that stores all the communication can later read all the messages sent by Alice if she manages to obtain Bob's private key. However, in our protocol, first, Eve should manage to obtain both Bob's private key sk_b^p and Alice's private key sk_a^k



Setup:

$(\mathsf{pk}_a^k, \mathsf{sk}_a^k) \leftarrow \mathrm{KEM.Gen}(\lambda)$ $(\mathsf{pk}_b^k, \mathsf{sk}_b^k) \leftarrow \mathrm{KEM.Gen}(\lambda)$

$(\mathsf{pk}_a^p, \mathsf{sk}_a^p) \leftarrow \mathrm{PKE.Gen}(\lambda)$ $(\mathsf{pk}_b^p, \mathsf{sk}_b^p) \leftarrow \mathrm{PKE.Gen}(\lambda)$

Alice$(m_a, \mathsf{sk}_a^k, \mathsf{sk}_a^p)$ **Bob**$(m_b, \mathsf{sk}_b^k, \mathsf{sk}_b^p)$

Generating First Session key:

$r_a^1 \xleftarrow{\$} \mathtt{R}$

$\mathrm{KEM.Enc}_{\mathsf{pk}_a^k}(r_a^1) = (k_a^1, c_a^1)$

$\xleftarrow{(\text{I am Bob},\ c_a^1)}$

Generating Next Session Key:

$r_b^1 \xleftarrow{\$} \mathtt{R}$

$\mathrm{KEM.Enc}_{\mathsf{pk}_b^k}(r_b) = (k_b^1, c_b^1)$

Encrypting m_a :

$k_a^1 \leftarrow \mathrm{KEM.Dec}(\mathsf{sk}_a^k, c_a^1)$

$\delta_a \xleftarrow{\$} \mathtt{MSP},\ \alpha_b^1 \xleftarrow{\$} \mathtt{R}$

$c_b' = \mathrm{PKE.Enc}(\mathsf{pk}_b^p, \delta_a, m_a \oplus k_a^1; \alpha_b^1)$

$\theta = \mathrm{MAC}_{\delta_a \oplus k_a^1}(c_b^1, c_b', c_a^1)$

$\xrightarrow{c_b^1,\ (c_b', \theta)}$

Reading the Message:

$(\delta_a,\ m_a \oplus k_a^1) \leftarrow \mathrm{PKE.Dec}(\mathsf{sk}_b^p, c_b')$

$b \leftarrow \mathrm{MAC.Verif}_{\delta_a \oplus k_a^1}(\theta, c_b^1, c_b', c_a^1)$

if $b = 1$ return m_a

Generating Next Session key:

$r_a^2 \xleftarrow{\$} \mathtt{R}$,

$\mathrm{KEM.Enc}_{\mathsf{pk}_a^k}(r_a^2) = (k_a^2, c_a^2)$

Encrypting m_b :

$k_b^1 \leftarrow \mathrm{KEM.Dec}(\mathsf{sk}_b^k, c_b^1)$

$\delta_b \xleftarrow{\$} \mathtt{MSP},\ \alpha_b^1 \xleftarrow{\$} \mathtt{R}$

$c_a' = \mathrm{PKE.Enc}(\mathsf{pk}_a^p, \delta_b, m_b \oplus k_b^1; \alpha_b^1)$

$\theta_b = \mathrm{MAC}_{\delta_b \oplus k_b^1}(c_a^2, c_a', c_b^1)$

$\xleftarrow{c_a^2,\ (c_a', \theta_b)}$

Fig. 2. Instant Messaging Protocol.

to read the message m_a[6]. Second, even if Eve manages to obtain both sk_b^p, sk_a^k and consequently reads m_a, she can not be convinced that the message m_a is sent by Alice. This is due to the *non-binding* property of the signature (c'_b, θ).

Our *instant messaging* protocol has an extra property that Alice can convince a third party Judge that she has sent the message m_a if it is needed. Note that if Alice and Bob are two agents for an organization that are conducting a confidential communication in which not sending the message m_a will have some legal consequences for Alice, an instant messaging protocol as OTR protocol [10] might not be the right choice for Alice to use.

VII. Conclusion and Open Problem

We define a non-binding property for a designated verifier signature scheme in this paper. We propose a DVS that is *non-transferable* and *non-binding*. In addition, we propose an *Instant Messaging* protocol by modifying our *non-binding* signature scheme. A formal security proof of our messaging protocol in the *Universal Composability Framework* (similar to [16]), is not in the scope of this project and remains an open question.

VIII. Acknowledgment

Ehsan Ebrahimi was supported by the Luxembourg National Research Fund under the Junior CORE project QSP (C22/IS/17272217/QSP/Ebrahimi). We thank Benjamin Dowling for a discussion about our Instant Messaging Protocol.

References

[1] Joël Alwen, Sandro Coretti, and Yevgeniy Dodis. The double ratchet: Security notions, proofs, and modularization for the signal protocol. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part I*, volume 11476 of *Lecture Notes in Computer Science*, pages 129–158. Springer, 2019.

[2] Joël Alwen, Sandro Coretti, Yevgeniy Dodis, and Yiannis Tselekounis. Security analysis and improvements for the IETF MLS standard for group messaging. In Daniele Micciancio and Thomas Ristenpart, editors, *Advances in Cryptology - CRYPTO 2020 - 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August 17-21, 2020, Proceedings, Part I*, volume 12170 of *Lecture Notes in Computer Science*, pages 248–277. Springer, 2020.

[3] Giuseppe Ateniese, Daniel H. Chou, Breno de Medeiros, and Gene Tsudik. Sanitizable signatures. In Sabrina De Capitani di Vimercati, Paul F. Syverson, and Dieter Gollmann, editors, *Computer Security - ESORICS 2005, 10th European Symposium on Research in Computer Security, Milan, Italy, September 12-14, 2005, Proceedings*, volume 3679 of *Lecture Notes in Computer Science*, pages 159–177. Springer, 2005.

[4] Fatih Balli, Paul Rösler, and Serge Vaudenay. Determining the core primitive for optimally secure ratcheting. In Shiho Moriai and Huaxiong Wang, editors, *Advances in Cryptology - ASIACRYPT 2020 - 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7-11, 2020, Proceedings, Part III*, volume 12493 of *Lecture Notes in Computer Science*, pages 621–650. Springer, 2020.

[5] Mihir Bellare, Asha Camper Singh, Joseph Jaeger, Maya Nyayapati, and Igors Stepanovs. Ratcheted encryption and key exchange: The security of messaging. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part III*, volume 10403 of *Lecture Notes in Computer Science*, pages 619–650. Springer, 2017.

[6] Alexander Bienstock, Yevgeniy Dodis, and Paul Rösler. On the price of concurrency in group ratcheting protocols. In Rafael Pass and Krzysztof Pietrzak, editors, *Theory of Cryptography - 18th International Conference, TCC 2020, Durham, NC, USA, November 16-19, 2020, Proceedings, Part II*, volume 12551 of *Lecture Notes in Computer Science*, pages 198–228. Springer, 2020.

[7] Olivier Blazy, Angèle Bossuat, Xavier Bultel, Pierre-Alain Fouque, Cristina Onete, and Elena Pagnin. SAID: reshaping signal into an identity-based asynchronous messaging protocol with authenticated ratcheting. In *IEEE European Symposium on Security and Privacy, EuroS&P 2019, Stockholm, Sweden, June 17-19, 2019*, pages 294–309. IEEE, 2019.

[8] Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. Random oracles in a quantum world. In Dong Hoon Lee and Xiaoyun Wang, editors, *Advances in Cryptology - ASIACRYPT 2011 - 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings*, volume 7073 of *Lecture Notes in Computer Science*, pages 41–69. Springer, 2011.

[9] Dan Boneh and Mark Zhandry. Quantum-secure message authentication codes. In Thomas Johansson and Phong Q. Nguyen, editors, *Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings*, volume 7881 of *Lecture Notes in Computer Science*, pages 592–608. Springer, 2013.

[10] Nikita Borisov, Ian Goldberg, and Eric A. Brewer. Off-the-record communication, or, why not to use PGP. In Vijay Atluri, Paul F. Syverson, and Sabrina De Capitani di Vimercati, editors, *Proceedings of the 2004 ACM Workshop on Privacy in the Electronic Society, WPES*

[6] Obtaining two secret-keys from two different individuals is harder.

2004, Washington, DC, USA, October 28, 2004, pages 77–84. ACM, 2004.

[11] Joppe W. Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehlé. CRYSTALS - kyber: A cca-secure module-lattice-based KEM. In *2018 IEEE European Symposium on Security and Privacy, EuroS&P 2018, London, United Kingdom, April 24-26, 2018*, pages 353–367. IEEE, 2018.

[12] Joan Boyar, David Chaum, Ivan Damgård, and Torben P. Pedersen. Convertible undeniable signatures. In Alfred Menezes and Scott A. Vanstone, editors, *Advances in Cryptology - CRYPTO '90, 10th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1990, Proceedings*, volume 537 of *Lecture Notes in Computer Science*, pages 189–205. Springer, 1990.

[13] Andrea Caforio, F. Betül Durak, and Serge Vaudenay. Beyond security and efficiency: On-demand ratcheting with security awareness. In Juan A. Garay, editor, *Public-Key Cryptography - PKC 2021 - 24th IACR International Conference on Practice and Theory of Public Key Cryptography, Virtual Event, May 10-13, 2021, Proceedings, Part II*, volume 12711 of *Lecture Notes in Computer Science*, pages 649–677. Springer, 2021.

[14] Jie Cai, Han Jiang, Pingyuan Zhang, Zhihua Zheng, Guangshi Lyu, and Qiuliang Xu. An efficient strong designated verifier signature based on -sis assumption. *IEEE Access*, 7:3938–3947, 2019.

[15] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited. *J. ACM*, 51(4):557–594, 2004.

[16] Ran Canetti, Palak Jain, Marika Swanberg, and Mayank Varia. Universally composable end-to-end secure messaging. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology - CRYPTO 2022 - 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15-18, 2022, Proceedings, Part II*, volume 13508 of *Lecture Notes in Computer Science*, pages 3–33. Springer, 2022.

[17] Melissa Chase, Trevor Perrin, and Greg Zaverucha. The signal private group system and anonymous credentials supporting efficient verifiable encryption. In Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna, editors, *CCS '20: 2020 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, USA, November 9-13, 2020*, pages 1445–1459. ACM, 2020.

[18] David Chaum and Hans Van Antwerpen. Undeniable signatures. In Gilles Brassard, editor, *Advances in Cryptology - CRYPTO '89, 9th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 1989, Proceedings*, volume 435 of *Lecture Notes in Computer Science*, pages 212–216. Springer, 1989.

[19] David Chaum and Eugène van Heyst. Group signatures. In Donald W. Davies, editor, *Advances in Cryptology - EUROCRYPT '91, Workshop on the Theory and Application of of Cryptographic Techniques, Brighton, UK, April 8-11, 1991, Proceedings*, volume 547 of *Lecture Notes in Computer Science*, pages 257–265. Springer, 1991.

[20] Kaiming Chen and Jiageng Chen. Anonymous end to end encryption group messaging protocol based on asynchronous ratchet tree. In Weizhi Meng, Dieter Gollmann, Christian Damsgaard Jensen, and Jianying Zhou, editors, *Information and Communications Security - 22nd International Conference, ICICS 2020, Copenhagen, Denmark, August 24-26, 2020, Proceedings*, volume 12282 of *Lecture Notes in Computer Science*, pages 588–605. Springer, 2020.

[21] Katriel Cohn-Gordon, Cas Cremers, Benjamin Dowling, Luke Garratt, and Douglas Stebila. A formal security analysis of the signal messaging protocol. *J. Cryptol.*, 33(4):1914–1983, 2020.

[22] Katriel Cohn-Gordon, Cas Cremers, and Luke Garratt. On post-compromise security. In *IEEE 29th Computer Security Foundations Symposium, CSF 2016, Lisbon, Portugal, June 27 - July 1, 2016*, pages 164–178. IEEE Computer Society, 2016.

[23] Katriel Cohn-Gordon, Cas Cremers, Luke Garratt, Jon Millican, and Kevin Milner. On ends-to-ends encryption: Asynchronous group messaging with strong security guarantees. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang, editors, *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, ON, Canada, October 15-19, 2018*, pages 1802–1819. ACM, 2018.

[24] Cas Cremers, Jaiden Fairoze, Benjamin Kiesl, and Aurora Naska. Clone detection in secure messaging: Improving post-compromise security in practice. In Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna, editors, *CCS '20: 2020 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, USA, November 9-13, 2020*, pages 1481–1495. ACM, 2020.

[25] F. Betül Durak and Serge Vaudenay. Bidirectional asynchronous ratcheted key agreement with linear complexity. In Nuttapong Attrapadung and Takeshi Yagi, editors, *Advances in Information and Computer Security - 14th International Workshop on Security, IWSEC 2019, Tokyo, Japan, August 28-30, 2019, Proceedings*, volume 11689 of *Lecture Notes in Computer Science*, pages 343–362. Springer, 2019.

[26] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *Advances in Cryptology - CRYPTO '86, Santa Barbara, California, USA, 1986, Proceedings*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer, 1986.

[27] Steven D. Galbraith, Christophe Petit, Barak Shani, and Yan Bo Ti. On the security of supersingular isogeny cryptosystems. In Jung Hee Cheon and Tsuyoshi Takagi,

editors, *Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part I*, volume 10031 of *Lecture Notes in Computer Science*, pages 63–91, 2016.

[28] Juan A. Garay, Markus Jakobsson, and Philip D. MacKenzie. Abuse-free optimistic contract signing. In Michael J. Wiener, editor, *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*, volume 1666 of *Lecture Notes in Computer Science*, pages 449–466. Springer, 1999.

[29] Juan A. Garay and Philip D. MacKenzie. Abuse-free multi-party contract signing. In Prasad Jayanti, editor, *Distributed Computing, 13th International Symposium, Bratislava, Slovak Republic, September 27-29, 1999, Proceedings*, volume 1693 of *Lecture Notes in Computer Science*, pages 151–165. Springer, 1999.

[30] Simson L. Garfinkel. *PGP - pretty good privacy: encryption for everyone (2. ed.)*. O'Reilly, 1995.

[31] Christian Hanser and Daniel Slamanig. Structure-preserving signatures on equivalence classes and their application to anonymous credentials. In Palash Sarkar and Tetsu Iwata, editors, *Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014. Proceedings, Part I*, volume 8873 of *Lecture Notes in Computer Science*, pages 491–511. Springer, 2014.

[32] Qiong Huang, Guomin Yang, Duncan S. Wong, and Willy Susilo. Efficient strong designated verifier signature schemes without random oracle or with non-delegatability. *Int. J. Inf. Sec.*, 10(6):373–385, 2011.

[33] Joseph Jaeger and Igors Stepanovs. Optimal channel security against fine-grained state compromise: The safety of messaging. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part I*, volume 10991 of *Lecture Notes in Computer Science*, pages 33–62. Springer, 2018.

[34] Markus Jakobsson, Kazue Sako, and Russell Impagliazzo. Designated verifier proofs and their applications. In Ueli M. Maurer, editor, *Advances in Cryptology - EUROCRYPT '96, International Conference on the Theory and Application of Cryptographic Techniques, Saragossa, Spain, May 12-16, 1996, Proceeding*, volume 1070 of *Lecture Notes in Computer Science*, pages 143–154. Springer, 1996.

[35] Robert Johnson, David Molnar, Dawn Xiaodong Song, and David A. Wagner. Homomorphic signature schemes. In Bart Preneel, editor, *Topics in Cryptology - CT-RSA 2002, The Cryptographer's Track at the RSA Conference, 2002, San Jose, CA, USA, February 18-22, 2002, Proceedings*, volume 2271 of *Lecture Notes in Computer Science*, pages 244–262. Springer, 2002.

[36] Daniel Jost, Ueli Maurer, and Marta Mularczyk. Efficient ratcheting: Almost-optimal guarantees for secure messaging. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part I*, volume 11476 of *Lecture Notes in Computer Science*, pages 159–188. Springer, 2019.

[37] Daniel Jost, Ueli Maurer, and Marta Mularczyk. A unified and composable take on ratcheting. In Dennis Hofheinz and Alon Rosen, editors, *Theory of Cryptography - 17th International Conference, TCC 2019, Nuremberg, Germany, December 1-5, 2019, Proceedings, Part II*, volume 11892 of *Lecture Notes in Computer Science*, pages 180–210. Springer, 2019.

[38] Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography*. Chapman and Hall/CRC Press, 2007.

[39] Marina Kolb. *The Memorandum of Understanding*, pages 141–162. Palgrave Macmillan UK, London, 2013.

[40] Fabien Laguillaumie and Damien Vergnaud. Designated verifier signatures: Anonymity and efficient construction from any bilinear map. In Carlo Blundo and Stelvio Cimato, editors, *Security in Communication Networks, 4th International Conference, SCN 2004, Amalfi, Italy, September 8-10, 2004, Revised Selected Papers*, volume 3352 of *Lecture Notes in Computer Science*, pages 105–119. Springer, 2004.

[41] Fabien Laguillaumie and Damien Vergnaud. Multi-designated verifiers signatures. In Javier López, Sihan Qing, and Eiji Okamoto, editors, *Information and Communications Security, 6th International Conference, ICICS 2004, Malaga, Spain, October 27-29, 2004, Proceedings*, volume 3269 of *Lecture Notes in Computer Science*, pages 495–507. Springer, 2004.

[42] BaoHong Li, YanZhi Liu, and Sai Yang. Lattice-based universal designated verifier signatures. In *15th IEEE International Conference on e-Business Engineering, ICEBE 2018, Xi'an, China, October 12-14, 2018*, pages 329–334. IEEE Computer Society, 2018.

[43] Yong Li, Helger Lipmaa, and Dingyi Pei. On delegatability of four designated verifier signatures. In Sihan Qing, Wenbo Mao, Javier López, and Guilin Wang, editors, *Information and Communications Security, 7th International Conference, ICICS 2005, Beijing, China, December 10-13, 2005, Proceedings*, volume 3783 of *Lecture Notes in Computer Science*, pages 61–71. Springer, 2005.

[44] Helger Lipmaa, Guilin Wang, and Feng Bao. Designated verifier signature schemes: Attacks, new security notions and a new construction. In Luís Caires, Giuseppe F. Italiano, Luís Monteiro, Catuscia Palamidessi, and Moti Yung, editors, *Automata, Languages and Programming, 32nd International Colloquium, ICALP 2005, Lisbon, Portugal, July 11-15, 2005, Proceedings*, volume 3580

of *Lecture Notes in Computer Science*, pages 459–471. Springer, 2005.

[45] Geontae Noh and Ik Rae Jeong. Strong designated verifier signature scheme from lattices in the standard model. *Secur. Commun. Networks*, 9(18):6202–6214, 2016.

[46] Chris Peikert. A decade of lattice cryptography. *IACR Cryptol. ePrint Arch.*, page 939, 2015.

[47] Bertram Poettering and Paul Rösler. Towards bidirectional ratcheted key exchange. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part I*, volume 10991 of *Lecture Notes in Computer Science*, pages 3–32. Springer, 2018.

[48] Mario Di Raimondo, Rosario Gennaro, and Hugo Krawczyk. Secure off-the-record messaging. In Vijay Atluri, Sabrina De Capitani di Vimercati, and Roger Dingledine, editors, *Proceedings of the 2005 ACM Workshop on Privacy in the Electronic Society, WPES 2005, Alexandria, VA, USA, November 7, 2005*, pages 81–89. ACM, 2005.

[49] Parvin Rastegari, Mehdi Berenjkoub, Mohammad Dakhilalian, and Willy Susilo. Universal designated verifier signature scheme with non-delegatability in the standard model. *Inf. Sci.*, 479:321–334, 2019.

[50] Ronald L. Rivest, Adi Shamir, and Yael Tauman. How to leak a secret. In Colin Boyd, editor, *Advances in Cryptology - ASIACRYPT 2001, 7th International Conference on the Theory and Application of Cryptology and Information Security, Gold Coast, Australia, December 9-13, 2001, Proceedings*, volume 2248 of *Lecture Notes in Computer Science*, pages 552–565. Springer, 2001.

[51] Paul Rösler, Christian Mainka, and Jörg Schwenk. More is less: On the end-to-end security of group chats in signal, whatsapp, and threema. In *2018 IEEE European Symposium on Security and Privacy, EuroS&P 2018, London, United Kingdom, April 24-26, 2018*, pages 415–429. IEEE, 2018.

[52] Lior Rotem and Gil Segev. Out-of-band authentication in group messaging: Computational, statistical, optimal. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part I*, volume 10991 of *Lecture Notes in Computer Science*, pages 63–89. Springer, 2018.

[53] Shahrokh Saeednia, Steve Kremer, and Olivier Markowitch. An efficient strong designated verifier signature scheme. In Jong In Lim and Dong Hoon Lee, editors, *Information Security and Cryptology - ICISC 2003, 6th International Conference, Seoul, Korea, November 27-28, 2003, Revised Papers*, volume 2971 of *Lecture Notes in Computer Science*, pages 40–54. Springer, 2003.

[54] Peter W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *35th Annual Symposium on Foundations of Computer Science, Santa Fe, New Mexico, USA, 20-22 November 1994*, pages 124–134. IEEE Computer Society, 1994.

[55] Ron Steinfeld, Laurence Bull, Huaxiong Wang, and Josef Pieprzyk. Universal designated-verifier signatures. In Chi-Sung Laih, editor, *Advances in Cryptology - ASIACRYPT 2003, 9th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, November 30 - December 4, 2003, Proceedings*, volume 2894 of *Lecture Notes in Computer Science*, pages 523–542. Springer, 2003.

[56] Ron Steinfeld, Huaxiong Wang, and Josef Pieprzyk. Efficient extension of standard schnorr/rsa signatures into universal designated-verifier signatures. In Feng Bao, Robert H. Deng, and Jianying Zhou, editors, *Public Key Cryptography - PKC 2004, 7th International Workshop on Theory and Practice in Public Key Cryptography, Singapore, March 1-4, 2004*, volume 2947 of *Lecture Notes in Computer Science*, pages 86–100. Springer, 2004.

[57] Xi Sun, Haibo Tian, and Yumin Wang. Toward quantum-resistant strong designated verifier signature from isogenies. In Fatos Xhafa, Leonard Barolli, Florin Pop, Xiaofeng Chen, and Valentin Cristea, editors, *2012 Fourth International Conference on Intelligent Networking and Collaborative Systems, INCoS 2012, Bucharest, Romania, September 19-21, 2012*, pages 292–296. IEEE, 2012.

[58] Nik Unger, Sergej Dechand, Joseph Bonneau, Sascha Fahl, Henning Perl, Ian Goldberg, and Matthew Smith. Sok: Secure messaging. In *2015 IEEE Symposium on Security and Privacy, SP 2015, San Jose, CA, USA, May 17-21, 2015*, pages 232–249. IEEE Computer Society, 2015.

[59] Nik Unger and Ian Goldberg. Deniable key exchanges for secure messaging. In Indrajit Ray, Ninghui Li, and Christopher Kruegel, editors, *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, October 12-16, 2015*, pages 1211–1223. ACM, 2015.

[60] Nik Unger and Ian Goldberg. Improved strongly deniable authenticated key exchanges for secure messaging. *Proc. Priv. Enhancing Technol.*, 2018(1):21–66, 2018.

[61] Damien Vergnaud. New extensions of pairing-based signatures into universal designated verifier signatures. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, *Automata, Languages and Programming, 33rd International Colloquium, ICALP 2006, Venice, Italy, July 10-14, 2006, Proceedings, Part II*, volume 4052 of *Lecture Notes in Computer Science*, pages 58–69. Springer, 2006.

[62] Hailun Yan and Serge Vaudenay. Symmetric asynchronous ratcheted communication with associated data. In Kazumaro Aoki and Akira Kanaoka, editors, *Advances in Information and Computer Security - 15th International Workshop on Security, IWSEC 2020, Fukui, Japan, September 2-4, 2020, Proceedings*, volume 12231

of *Lecture Notes in Computer Science*, pages 184–204. Springer, 2020.

[63] Yongqiang Zhang, Qiang Liu, Chengpei Tang, and Haibo Tian. A lattice-based designated verifier signature for cloud computing. *Int. J. High Perform. Comput. Netw.*, 8(2):135–143, 2015.