Efficient Deniable Authentication for Signatures
Application to Machine-Readable Travel Document

Jean Monnerat[1,⋆], Sylvain Pasini[2,⋆⋆], and Serge Vaudenay[2]

[1] SwissSign AG, Zurich, Switzerland
[2] EPFL, Lausanne, Switzerland

Abstract. Releasing a classical digital signature faces to privacy issues. Indeed, there are cases where the prover needs to authenticate some data without making it possible for any malicious verifier to transfer the proof to anyone else. It is for instance the case for e-passports where the signature from the national authority authenticates personal data. To solve this problem, we can prove knowledge of a valid signature without revealing it. This proof should be non-transferable.

We first study deniability for signature verification. Deniability is essentially a weaker form of non-transferability. It holds as soon as the protocol is finished (it is often called offline non-transferability).

We introduce Offline Non-Transferable Authentication Protocol (ONTAP) and we show that it can be built by using a classical signature scheme and a deniable zero-knowledge proof of knowledge. For that reason, we use a generic transform for Σ-protocols.

Finally, we give examples to upgrade signature standards based on RSA or ElGamal into an ONTAP. Our examples are well-suited for implementation in e-passports.

1 Introduction

Digital signature schemes are one of the most important primitives in cryptography. A digital signature on a document allows to bind this document with a public key (e.g. an identity). One drawback is the privacy issue. Indeed, assume Alice signed a message and sent both the message and the signature to Bob. With a standard digital signature scheme Bob can verify its validity, but in addition he is able to convince anyone else of its validity. In some situations this transferability leads to privacy issues.

Monnerat, Vaudenay, and Vuagnoux [34,47,48] studied the e-passport standards and identified the privacy issue from leaking of signatures on private data. For instance, data such as official name, true date of birth, citizenship and a facial image together with a digital signature could easily be released on the Internet or sold by a malicious verifier, allowing to convince anybody of their

⋆ Most of this work was done when the first author was at UCSD and supported by the Swiss National Foundation, PBEL2-116915.

⋆⋆ Supported by the Swiss National Science Foundation, 200021-113329.

authenticity. None of these data is really confidential. For instance, the *true* date of birth of some person can fairly be estimated or propagated by gossiping. The person can still claim that the gossiped date of birth is incorrect and keep it private. What is more sensitive is a *proof* that a date of birth is true because the person can no longer deny evidence that the proof is correct. For this reason, the authors suggested to use non-transferable proof of signature knowledge. So, the passport can convince the border patrol of the authenticity of the data without revealing the signature.

In our scenario, we have a signer (the national authority), a prover (the e-passport), and a verifier (the border patrol). Clearly, the passport should not know the signing key which is kept secret by the national authority.

Full non-transferability requires a public-key infrastructure (PKI) for verifiers. Certificate verification should be secured to avoid transfer attacks using rogue keys. While a PKI for border patrols is proposed in the EAC standard [36], this is not enough for the privacy issue we have in mind. Indeed, the key of verifiers in EAC will only be checked for verifiers in a country with agreements with the home country. That is, non-transferability would only be enforced in friendly countries but not in others. This seems pretty weird.

For this reason, we want to avoid PKI for verifiers and we will focus on a weaker form of non-transferability, which holds after the protocol is complete. That is, we enforce *offline non-transferability* which is equivalent to deniability (sometimes called self-simulatability). Zero-knowledge proofs are inherently deniable in the plain model while zero-knowledge protocols in the common reference string (CRS) model or the random oracle model (ROM) are not necessarily deniable. However, protocols in the CRS or ROM are more attractive for efficiency reasons. So, we have to consider the notion of deniable zero-knowledge [40,41].

The above reasons motivated the study of non-transferable proof of signature knowledge with a strong focus on the efficiency. The goal is to find a protocol which can be implemented on e-passports for proving the knowledge of a valid signature to a border patrol in a setting where there is no PKI for border patrols and these latters may be dishonest. The international e-passport standard [35] proposes the use of RSA and ElGamal-based signatures schemes. The EAC [36] extension suggest that passports could run ECDH protocols. So, there is a little place for public-key cryptography.

Related work. Non-transitive signatures [16,38] and deniable message authentication [17] also deal with transferability issues but do not immediately apply for a three-party settings where an intermediate player (e.g. the e-passport) shows to another one that some data was authenticated by an authority.

Invisible signatures (aka undeniable signatures) were invented by Chaum and van Antwerpen [11] and they subsequently were studied in [4,9,15,23,33,37]. They are not universally verifiable, i.e. they make it impossible to tell valid and invalid signatures apart while making the signer able to prove validity or invalidity through an interactive protocol. Undeniable signatures only consider a two-party setting. One issue for our case is that in order to confirm or deny a signature, the prover must know the secret key.

With undeniable signatures, the signer cooperation is essential. Indeed, without its cooperation the signature is useless. The latter issue motivated the introduction of designated confirmer signatures [10]. In short, designated confirmer signatures work as undeniable signatures but the verification may be shifted from the signer to the confirmer. They were designed mostly to protect the verifier from signers who would refuse to participate in verification protocols.

Asokan, Shoup, and Waidner [1,2] proposed a solution to cross-exchange signatures between two parties in a fair way (using a trusted third party in a fair way). For that, they propose a way to transform a signature scheme into a verifiable escrow scheme. A verifiable escrow scheme is based on a homomorphism and allows to produce an escrow signature from a signature. Then, given the escrow signature, one can verify that it is really a signature without obtaining the signature. Finally, someone can recover the signature from the escrow signature by using the secret key. One drawback for our application is that the escrow signature is verifiable, thus it is some kind of signature and not deniable.

Non-transferability was studied by Jakobsson *et al.* [31,7] and they introduced designated-verifier proofs. The idea is to designate the signature to a verifier (using its public-key) and then only the designated-verifier can be convinced on the validity of the signature. One drawback is that the verifier must be known at the signature time. Later, Steinfeld *et al.* [46] introduced Universal Designated-Verifier Signature (UDVS). This scheme applies to a three party setting: signer, designator, and verifier. *Universal* refers to that any designator who obtained a universally verifiable signature from the signer is able to designate it to a verifier. This relies on a PKI for verifiers. The method for having every verifier attached to a public key is an overkill. This motivated Baek *et al.* [3] to define a weaker notion of non-transferability and they published the Universal Designated Verifier Signature Proof (UDVSP). The primitive is similar to the concept of UDVS except that no signature is given to the verifier. The designator does not need to know the verifier, only a signature proof is given to the verifier. This primitive assumes that verifiers are honest. Clearly, our application scenario does not meet this assumption and this construction does not remain secure when the verifiers may be malicious [32,45]. Indeed, considering malicious verifiers leads to transferable proofs by using the Fiat-Shamir transform [22]. In addition, The proposed UDVS or UDVSP constructions rely on bilinear mappings which seems not very easy to implement in the case of e-passports.

Recently, Shahandashti, Safavi-Naini, and Baek [45] worked on Credential Ownership Proofs (COP). There have some similar features as non-transferable signature. However, COP allow users to copy/share the credits which is clearly not desirable is the case of e-passports. They also protect against “double spending” which is not necessary in our case.

In this paper. To motivate our constructions, we start in Section 2 with a short overview on e-passports. In Section 3, we introduce some preliminaries and in particular, we recall the concept of deniable zero-knowledge in the CRS and RO models. In Section 4, we introduce the definition of an offline non-transferable authentication protocol (ONTAP). We propose a generic transform of a signature scheme into an ONTAP by using a deniable ZK proof of knowledge. In order to build secure ONTAP, we study strong construction of proofs of knowledge in Section 5. In particular, we study a generic transform of Σ-protocols. In Section 6, we propose ONTAP protocols which can be efficiently executed in constrained environments such as e-passports. In particular, we give an example based on the Guillou-Quisquater protocol for RSA-based signatures and another example based on the Schnorr protocol for ElGamal-based signatures.

Our work compared to others. As the UDVSP of Baek et al. [3] our ONTAP definition requires no PKI for verifiers. UDVSP and ONTAP are conceptually equivalent but the security notions differ. The UDVSP security from [3] uses three definitions, restrict to known message attacks and honest verifiers while we use two definitions, chosen message attacks and malicious verifiers. In addition to this, our instantiations of ONTAP can be built efficiently on standard signature schemes and need no change for the signing algorithm. UDVSP does not apply directly to RSA and ElGamal-based schemes while our proposed ONTAP implementations do. Our proposed implementations just require a modular exponentiation for the prover while the proposed UDVSP constructions require the use of bilinear mappings as well as signature transforms.

Recently, Shahandashti and Safavi-Naini [44] presented a construction for UDVS. As we saw before, UDVS requires PKI for verifiers and thus is not adapted in our case. However, they present a way for a signature holder to prove his signature knowledge to a verifier. They define a signature class $\mathbb{C}$ for which signatures can be converted in a public and a private part. The private part is simulatable and there exists a proof of knowledge for the private part. The signature holder simply needs to convert its signature, to send the public part to the verifier, and finally to prove his knowledge of the private part. They use this definition to designate a signature to a verifier by using a Fiat-Shamir transform on the interactive proof. Except the transform, we use a similar idea, i.e. a signature in two part, one simulatable and the other provable. The authors does not give any security proof (since it is not their main contribution). They use a classical Σ-protocol and thus their proof of knowledge is HVZK only. As seen before, HVZK is clearly not enough for the application we have in mind. Here we strengthen the knowledge proofs, we give formal security proofs and examples of implementations. Finally the scheme of [44] may loose deniability if a malicious verifier registers a rogue key.

2 Passive Authentication for MRTD

E-passports, formally called *machine-readable travel documents* (MRTD), are now available in many countries [35]. They use an embedded RFID chip to show evidence of a traveler identity through wireless communication.

The memory of the chip is organized in standard files: several *data groups* and one *security object document* (SOD). There are only two mandatory data groups: DG1 includes basic information such as the name of the person, its gender, date of birth, citizenship, as well as the passport number and validity; and DG2 contains a facial picture of the owner. The SOD includes the digest of each data group and a digital signature of this list of digests issued by the national authority. Clearly, the SOD gives evidence of someone's true name, or true age, or true gender, or true citizenship, etc. Providing the data groups and the SOD is called *passive authentication.*

Since all data is readable, chip cloning is possible. To solve this issue, there is an optional *active authentication* (AA) protocol. In that case, the chip possesses a pair public/private key. The public key is stored in a DG (and thus authenticated) while the private key is in a secure part of the memory (and so not clonable). Following AA, the reader simply sends a challenge and the chip signs it and give it back.

Due to the wireless access, data could be captured without the agreement by the holder. Following the standard, access to the chip can be protected using *basic access control* (BAC). In short, it proves to the chip that the reader have an optical access to the first page. It is not a real access control since anyone can implement an e-passport reader and read any passport without being authorized by public authorities.

BAC is by far insufficient since the new generation of e-passport will contain more private information such as fingerprint, address, etc. For this, the European Union is now promoting an *extended access control* (EAC) [36] which is based on more elaborate cryptographic protocols (semi-static ECDH key agreement with certificate) and terminal authentication based on a specific PKI. This PKI is also known to suffer from weaknesses (namely, the unreliable revocation procedure). In addition to this, EAC is only meant to protect non-mandatory data groups since mandatory ones should still be accessible to countries with no agreement to read extra information. This means that the SOD is not protected by EAC so will still leak evidence that a given protected data group is correct. Clearly, an adversary can still distinguish a correct EAC-protected data group from an incorrect one without being authorized to read it.

For this reasons, we propose to have the signature part of the SOD hidden and passive authentication replaced by some deniable authentication protocol. Note that the chip is able to carry out some RSA computation in AA. Therefore, we can assume that chips are able to run one or two RSA computations in the deniable authentication protocol.

3 Preliminaries

Let S be a finite set. We write $s \in_{\mathsf{u}} S$ to say that s is picked uniformly from S.

Throughout this article the term "algorithm" stands for a probabilistic polynomial-time (PPT) Turing machine modeled by deterministic functions in terms of an input and random coins.

We denote by $\mathsf{prot}_{\mathbf{P}(\alpha),\mathbf{V}(\beta)}(\gamma)$ an instance of the protocol "prot" between $\mathbf{P}$ and $\mathbf{V}$. The element γ denotes the common input of all participants, e.g. public keys, while α (resp. β) describes the private input of $\mathbf{P}$ (resp. $\mathbf{V}$). Note that when the protocol is not known or is implicitly known, the interaction between the two parties can be noted by $\langle \mathbf{P}(\alpha), \mathbf{V}(\beta)\rangle(\gamma)$.

In some cases, we need to only describe the view of $\mathcal{B}$ and we denote it by $\mathsf{View}_{\mathcal{B}}(\mathsf{prot}_{\mathcal{A},\mathcal{B}}(\cdot))$. We call "the view of $\mathcal{B}$" all inputs known by $\mathcal{B}$ (including the random tape and messages received by $\mathcal{B}$). All other messages can be computed from the view and the $\mathcal{B}$ algorithm.

Classical Digital Signature (DS) Schemes. We denote by $\mathcal{M}$ and $\mathcal{S}$ the message space and the signature space respectively. A (classical) *digital signature* (DS) scheme is defined by the three following algorithms: The $(K_p, K_s) \leftarrow \mathsf{setup}(1^\lambda)$ algorithm generates a key pair from a security parameter λ. The $\sigma \leftarrow \mathsf{sign}(K_s, m)$ algorithm outputs a signature $\sigma \in \mathcal{S}$ of a message $m \in \mathcal{M}$. The $b = \mathsf{verify}(K_p, m, \sigma)$ tells whether the pair (m, σ) is valid $(b = 1)$ or not$(b = 0)$.

The scheme is *complete* if for any $(K_p, K_s) \leftarrow \mathsf{setup}(1^\lambda)$, any message m, and any $\sigma \leftarrow \mathsf{sign}(K_s, m)$, then $\mathsf{verify}(K_p, m, \sigma) = 1$. The standard security requirement for a DS is the *existential unforgeability against a chosen-message attack* (EF-CMA) put forth by Goldwasser *et al.* [28]. This property ensures that nobody except the signer $\mathbf{S}$ can output a valid signature $\widehat{\sigma}$ for any new message m with a non-negligible probability.

Definition 1 (Security of DS). *Consider an adversary $\mathcal{A}$ against S. $\mathcal{A}$ plays a game against a challenger $\mathcal{C}$ who can sign messages. $\mathcal{A}$ is allowed to make queries to a signing oracle. The goal of $\mathcal{A}$ is to yield a valid pair $(\widehat{m}, \widehat{\sigma})$ such that $\widehat{m}$ was never sent to the signing oracle. The signature scheme is said* EF-CMA-secure *if no* PPT *adversary $\mathcal{A}$ can win this game with non-negligible probability.*

Proof of Knowledge and Deniable Zero-Knowledge. Let $R \subseteq \{0,1\}^* \times \{0,1\}^*$ be a binary relation with a polynomial-size witness, i.e., for any $(x, w) \in R$ we have $|w| \leq \mathsf{poly}(|x|)$. Let L_R be a language related to the binary relation R. L_R is the set of all x such that there exists a witness w and (x, w) is in R, i.e. $L_R = \{x : \exists w \text{ s.t. } (x, w) \in R\}$.

Let $(\mathbf{P}, \mathbf{V})$ be a pair of interactive Turing machines. For a given $x \in L_R$, $\mathbf{P}$ wants to prove to $\mathbf{V}$ that he knows the corresponding witness w. For this, $\mathbf{P}$ and $\mathbf{V}$ will use an *interactive proof* with common input x noted $\mathsf{proof}_{\mathbf{P}(w),\mathbf{V}}(x)$. At the end, $\mathbf{P}$ should have convinced $\mathbf{V}$ that he knows w. $\mathbf{V}$ outputs accept or reject. In order to formalize the notion of proof of knowledge we need to introduce the concept of *knowledge extractor* Ext. The Ext algorithm gets input x and access to the prover, while he attempts to compute w such that $(x, w) \in R$.

Consider any proof of knowledge between a prover $\mathbf{P}$ and a verifier $\mathbf{V}$. *Zero-knowledge* means that no information leaks to the verifier except the validity of the statement. This concept was formalized by Goldwasser, Micali, and Rackoff [26,27]. The main idea behind zero-knowledge is that any verifier should be able to run the simulator by himself (instead of interacting with a prover). However in the CRS model, the simulator is able to choose crs while no verifier is able to do that in reality. For this reason, we use the concept of *deniability* [40,41].

Definition 2 (Deniable Zero-Knowledge Proof of Knowledge). *Let* crs *be any common reference string (CRS). Let* H *be a random oracle. Let* $\kappa(x)$ *be a real valued function.* $\mathsf{proof}_{\mathbf{P}^{\mathsf{H}}(w),\mathbf{V}^{\mathsf{H}}}(x,\mathsf{crs})$ *is a* proof of knowledge *for the relation* R *with soundness error* $\kappa(x)$ *if the following holds :*

– Efficiency: $\mathbf{P}$ *and* $\mathbf{V}$ *are polynomially bounded.*
– Completeness: *On common input* x,crs*, if* $\mathbf{P}$ *has a witness* w *such that* $(x,w) \in R$*, then* $\langle \mathbf{P}^{\mathsf{H}}(w), \mathbf{V}^{\mathsf{H}}\rangle(x,\mathsf{crs})$ *always outputs* accept*.*
– Soundness: *Given a Turing machine* ${\mathbf{P}^*}^{\mathsf{H}}$*,* crs *and* H*, let* $\varepsilon(x)$ *be the probability that* $\langle {\mathbf{P}^*}^{\mathsf{H}}, \mathbf{V}^{\mathsf{H}}\rangle(x,\mathsf{crs})$ *accepts. There exists an extractor* Ext *and a constant* k *such that for any* crs*, any* H*, any* $\mathbf{P}^*$*, and any* x*, if* $\varepsilon(x) > \kappa(x)$*, then* $\mathsf{Ext}^{\mathbf{P}^*}(x)$ *outputs a witness* w *such that* $(x,w) \in R$ *within expected time* $\mathcal{O}\left(\frac{|x|^k}{\varepsilon(x)-\kappa(x)}\right)$ *where an access to* $\mathbf{P}^*$ *only counts as one step.*

A proof of knowledge $\mathsf{proof}_{\mathbf{P}^{\mathsf{H}},\mathbf{V}^{\mathsf{H}}}(\cdot)$ *for a relation* R *is* deniable zero-knowledge *if for any* PPT $\mathbf{V}^*$*, there exists a* PPT *simulator* $\mathsf{Sim}^{\mathsf{H}}$ *such that*

$$\{\mathsf{crs},\mathsf{H},\mathsf{View}_{\mathbf{V}^*}(\mathsf{proof}_{\mathbf{P}^{\mathsf{H}}(w),{\mathbf{V}^*}^{\mathsf{H}}(z)}(x,\mathsf{crs}))\}_{z\in\{0,1\}^*,x\in L_R} \text{ for arbitrary } w \in R(x)$$

and

$$\{\mathsf{crs},\mathsf{H},\mathsf{Sim}^{\mathsf{H}}(x,z,\mathsf{crs})\}_{z\in\{0,1\}^*,x\in L_R}$$

are computationally indistinguishable.

When crs and H are constant and H is polynomially computable, we obtain the definition in the standard model. When crs is constant, we obtain the random oracle model. When H is constant and polynomially computable, we obtain the CRS model. When $\mathbf{V}^*$ is restricted by $\mathbf{V}$, i.e. $\mathbf{V}^* = \mathbf{V}$, we obtain the honest verifier zero-knowledge (HVZK) definition.

It seems that the need for deniablity makes the CRS model collapse down to the plain model (see Pass [41]). Indeed, there exists an efficient generic transformation of deniable zero-knowledge protocols from the CRS model into the plain model. However this transformation adds some more rounds which increases the round complexity. So, deniable ZK protocols in the CRS model may still be attractive in practice.

Commitment Schemes. We define a keyed commitment scheme by the two following algorithms: The $(K_p,K_s) \leftarrow \mathsf{setup}(1^\lambda,R)$ algorithm generates a pair public/private key given random coins R. The $c = \mathsf{com}(K_p,m,r)$ algorithm allows to compute the commit value c for a given message m by using the public key K_p and random coins r. Knowing both c, m, and r (and K_p), the commitment is checked by $c = \mathsf{com}(K_p,m,r)$. A commitment scheme should be perfectly *hiding*, meaning that for any K_p generated by setup, $c \leftarrow \mathsf{com}(K_p,m,r)$ has a distribution which is independent from m. We assume that it is uniform. It should also be computationally *binding*, meaning that for any PPT algorithm given a random K_p generated by setup, the probability that it finds r,r',m,m' such that $m \neq m'$ and $\mathsf{commit}(K_p,m,r) = \mathsf{commit}(K_p,m',r')$ is negligible.

Trapdoor commitment schemes were introduced by Brassard, Chaum, and Crépeau [6]. A trapdoor commitment scheme is a keyed commitment scheme extend by a third algorithm, equiv, which defeats the binding property by using the secret key K_s. For any K_p, K_s generated by setup, any m, any $\widehat{c}$, and any execution $\widehat{r} \leftarrow \mathsf{equiv}(K_s, m, \widehat{c})$, we have $\widehat{c} = \mathsf{com}(K_p, m, \widehat{r})$.

For instance, a trapdoor commitment based on the discrete logarithm problem was proposed by Boyar and Kurtz [5]. Another trapdoor commitment scheme was proposed by Catalano et al. [8] based on the Paillier's trapdoor permutation [39].

Random Oracle Commitment Scheme. In the RO model, we can use the *RO commitment scheme.* Let H be a random oracle. The com algorithm with input m simply returns $c = \mathsf{H}(m\|r)$. To check the validity of the commit value c, given the message m' and the used random coins r', it is enough to check $c = \mathsf{H}(m'\|r')$.

4 Offline Non-transferable Authentication Protocol

Definition 3 (ONTAP). *We define an* offline non-transferable authentication protocol *(ONTAP) by the two following algorithms and the interactive verification protocol:*

- *The* $(K_p, K_s) \leftarrow \mathsf{setup}(1^\lambda)$ *algorithm generates a key pair.*
- *The* $\sigma = (\sigma_p, \sigma_s) \leftarrow \mathsf{sign}(K_s, m)$ *algorithm outputs a signature* $\sigma \in \mathcal{S}$ *of a message* $m \in \mathcal{M}$. σ *is split in a public part* σ_p *and a private part* σ_s.
- *The* $\mathsf{iProof}_{\mathbf{P}(\sigma_s),\mathbf{V}}(K_p, m, \sigma_p)$ *protocol allows a prover* $\mathbf{P}$ *to convince a verifier* $\mathbf{V}$ *that he knows a* σ_s *to complete* σ_p *in a valid signature for* m. *At the end* $\mathbf{V}$ *accepts or rejects.*

The scheme is complete *if for any* $(K_p, K_s) \leftarrow \mathsf{setup}(1^\lambda)$, *any message* m, *and any* $(\sigma_p, \sigma_s) \leftarrow \mathsf{sign}(K_s, m)$, $\mathbf{V}$ *always accepts in* $\mathsf{iProof}_{\mathbf{P}(\sigma_s),\mathbf{V}}(K_p, m, \sigma_p)$.

The UDVSP [3] uses a KeyGen algorithm which is equivalent to our setup algorithm. The Sign algorithm outputs a classical signatures universally verifiable by using the Verify algorithm, there is a $\mathsf{Transform}$ algorithm which generates a modified signature (with a public and secret part) from the universally verifiable one. Our sign algorithm may be built with the Sign and $\mathsf{Transform}$ algorithms from the UDVSP and conversely. We removed the Verify algorithm since it is useless with our definition. Finally, there is an interactive proof $\mathsf{IVerify}$ as our iProof. So, the two definitions are conceptually equivalent. The main difference comes from the security requirements.

The ONTAP is secure if it satisfies the next two definitions.

Definition 4 (Offline Non-Transferability of ONTAP). *Consider an adversary* $\mathcal{A}$ *against the ONTAP.* $\mathcal{A}$ *plays a game with a challenger* $\mathcal{C}$. *The goal of* $\mathcal{A}$ *is to get evidence that some message* $\widehat{m}$ *was signed. During the training phase,* $\mathcal{A}$ *is allowed to query a sign oracle denoted* Sign. *After the training phase,* $\mathcal{A}$ *selects some* $\widehat{m}$, $\mathcal{C}$ *signs it and reveals* $\widehat{\sigma}_p$. *Then,* $\mathcal{A}$ *runs a session of* $\mathsf{iProof}_{\mathbf{P}(\widehat{\sigma}_s),\mathcal{A}}(K_p, \widehat{m}, \widehat{\sigma}_p)$ *protocol. At the end of the game,* $\mathcal{A}$ *outputs all input*

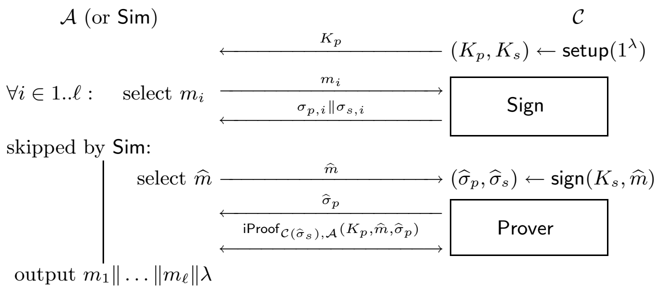
$\mathcal{A}$ (or Sim)
$\mathcal{C}$
K_p
$(K_p, K_s) \leftarrow \mathsf{setup}(1^\lambda)$
$\forall i \in 1..\ell$: select m_i
m_i
$\sigma_{p,i} \| \sigma_{s,i}$
Sign
skipped by Sim:
select $\widehat{m}$
$\widehat{m}$
$(\widehat{\sigma}_p, \widehat{\sigma}_s) \leftarrow \mathsf{sign}(K_s, \widehat{m})$
$\widehat{\sigma}_p$
$\mathsf{iProof}_{\mathcal{C}(\widehat{\sigma}_s),\mathcal{A}}(K_p, \widehat{m}, \widehat{\sigma}_p)$
Prover
output $m_1 \| \dots \| m_\ell \| \lambda$

Fig. 1. ONTAP Non-Transferable Game

queried to Sign *and its state* λ. *We introduce* Sim *which plays the same game but selects no* $\widehat{m}$ *and runs no* iProof *protocol.*

The ONTAP scheme is said offline non-transferable *if for any adversary* $\mathcal{A}$ *there exists a simulator* Sim *such that their output in the game of Fig. 1 are computationally indistinguishable.*

Definition 5 (Unforgeability of ONTAP). *Consider an adversary* $\mathcal{A}$ *against the ONTAP.* $\mathcal{A}$ *plays a game with a challenger* $\mathcal{C}$. *The goal of* $\mathcal{A}$ *is to convince* $\mathcal{C}$ *by running the* iProof *protocol that he knows* $\widehat{\sigma}_s$ *to complete* $\widehat{\sigma}_p$ *in a valid signature for* $\widehat{m}$. *During a training phase,* $\mathcal{A}$ *is allowed to query a sign oracle denoted* Sign. *On input message* m, Sign *answers the complete valid signature* (σ_p, σ_s). *After this training phase,* $\mathcal{A}$ *selects a* $\widehat{m}$ *and a* $\widehat{\sigma}_p$ *with* $\widehat{m}$ *not sent to* Sign. $\mathcal{A}$ *simulates a prover to a honest verifier (see Fig. 2).*

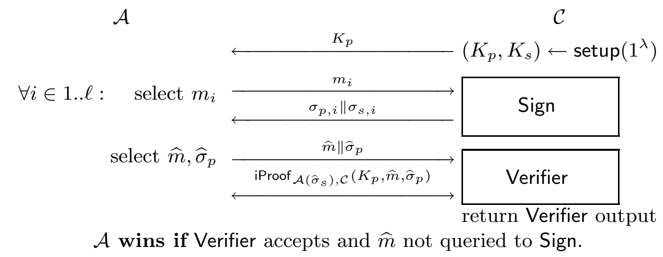
$\mathcal{A}$
$\mathcal{C}$
K_p
$(K_p, K_s) \leftarrow \mathsf{setup}(1^\lambda)$
$\forall i \in 1..\ell$: select m_i
m_i
$\sigma_{p,i} \| \sigma_{s,i}$
Sign
select $\widehat{m}, \widehat{\sigma}_p$
$\widehat{m} \| \widehat{\sigma}_p$
$\mathsf{iProof}_{\mathcal{A}(\widehat{\sigma}_s),\mathcal{C}}(K_p, \widehat{m}, \widehat{\sigma}_p)$
Verifier
return Verifier output
$\mathcal{A}$ **wins if** Verifier accepts and $\widehat{m}$ not queried to Sign.

Fig. 2. ONTAP Unforgeability Game

The ONTAP scheme is said unforgeable *if no* PPT *adversary* $\mathcal{A}$ *can make the honest verifier accepting the game of Fig. 2 with non-negligible probability.*

Clearly, Def. 5 implies unforgeability in the sense of Def. 1 since anyone able to forge a signature is also able to win the game of Fig. 2.

In Def. 5, we could give access to non-concurrent prover oracles to the adversary $\mathcal{A}^*$. Suppose it is the case and we denote them by Prover_j's. These oracles simulate the behaviour of an honest prover $\mathbf{P}$ in $\mathsf{iProof}_{\mathbf{P}(\sigma^*_{s,j}),\mathcal{A}^*}(K_p, m^*_j, \sigma^*_{p,j})$. Each oracle Prover_j is setup with a given message m^*_j and several iProof executions can be requested for the same m^*_j and some signature (σ_p, σ_s). Executions to the same Prover_j cannot be performed concurrently. This definition of unforgeability can be reduced to Def. 5 which uses no prover oracle. Suppose $\mathcal{A}^*$ is limited to m Prover oracles. We split $\mathcal{A}^*$ in several adversaries $\mathcal{A}_1^*$ to $\mathcal{A}_m^*$ playing modified games. Each $\mathcal{A}_i^*$ play with $\mathcal{C}$ where all Prover_j for $j \neq i$ are replaced by a query to the sign oracle and a simulation for the iProof protocol. So, only the Prover_i in the game with the adversary $\mathcal{A}_i$ uses a Prover oracle. Clearly, $\Pr[\mathcal{A}^* \text{ succeeds}] \leq \sum_{i=1}^{m} \Pr[\mathcal{A}_i^* \text{ succeeds}]$. Now we define adversaries A_i': each one plays the same game than $\mathcal{A}_i^*$ except than Prover_i is simulated by Sim as defined in Def. 4. By using the offline non-transferability property, the state of adversary $\mathcal{A}_i^*$ is computationally indistinguishable from the one of adversary $\mathcal{A}_i'$, so $\Pr[\mathcal{A}^* \text{ succeeds}] \leq \sum_{i=1}^{m} \Pr[\mathcal{A}_i' \text{ succeeds}] + \mathsf{negl}$. This proves that introducing a Prover oracle in Def. 5 does not strengthen our unforgeability notion when offline non-transferability is granted.

Theorem 6 (ONTAP construction). *Let $\mathcal{S}$ be a classical digital signature scheme in which the* sign *algorithm outputs a signature splittable in two parts: a public part σ_p and a private part σ_s. We assume there exists an algorithm* $\mathsf{simulate}$ *such that $\sigma_p \leftarrow \mathsf{simulate}(K_p, m)$ is computationally indistinguishable from the one generated by $\mathsf{sign}(K_s, m)$. Let* iProof *be a deniable zero-knowledge proof of knowledge for witness σ_s in the relation*

$$R(K_p \| m \| \sigma_p, \sigma_s) \iff \mathsf{verify}(K_p, m, \sigma_p \| \sigma_s) \ .$$

If $\mathcal{S}$ is EF-CMA-secure, then the ONTAP $(\mathsf{setup}, \mathsf{sign}, \mathsf{iProof})$ *is secure.*

The required signature scheme $\mathcal{S}$ should be in the class $\mathbb{C}$ defined in [44] which includes many signature schemes. Note that there exists a Σ-protocol for any signature scheme since any NP relation has one [44]. However, such a protocol is in general not efficient. Thanks to the next section, we transform Σ-protocols into a denial ZK proof of knowledge.

Proof. We start with the constructed ONTAP scheme and consider the ONTAP security games. Assuming that $\mathcal{S}$ is EF-CMA-secure, the public signature is simulatable, and iProof is deniable ZK, we want to show that the ONTAP is unforgeable and offline non-transferable.

Unforgeability: We consider an adversary $\mathcal{A}$ playing the ONTAP unforgeability game with a challenger $\mathcal{C}$ as depicted on Fig. 2. $\mathcal{A}$ is bounded by a complexity T and is limited by ℓ queries to the oracle Sign. We split $\mathcal{A}$ in two parts: $\mathcal{A}_1$, which represents the three first moves of $\mathcal{A}$ on Fig. 2 and outputs a state λ, and $\mathcal{A}_2(\lambda)$, which represents the last two moves of $\mathcal{A}$. Thanks to the soundness, Ext fed with $\mathcal{A}_2(\lambda)$ produces $\widehat{\sigma}_s$ such that $\mathsf{verify}(K_p, \widehat{m}, \widehat{\sigma}_p, \widehat{\sigma}_s)$ holds. Hence, running $\lambda \leftarrow \mathcal{A}_1^{\mathsf{Sign}}$, then $\mathsf{Ext}^{\mathcal{A}_2(\lambda)}$ wins in the EF-CMA game which is not possible.

Offline Non-Transferability: We construct Sim by running $\mathcal{A}$ until $\widehat{m}$ is submitted. Then, Sim runs $\widehat{\sigma}_p' \leftarrow \mathsf{simulate}(K_p, \widehat{m})$ and continues to simulate $\mathcal{A}$ by feeding it with $\widehat{\sigma}_p'$. Clearly, $\mathcal{A}$ with the simulated $\widehat{\sigma}_p'$ reaches a state which is indistinguishable from $\mathcal{A}$ with a true signature $\widehat{\sigma}_p$. Then, we use the simulator for iProof to simulate the final state (and output) from $\mathcal{A}$. □

5 Deniable ZK from Σ-Protocols

The notion of Σ-protocol represents an important tool for the design of zero-knowledge protocols. Below, we first briefly recall the required material and refer to Damgård [14] for a detailed treatment. Then, we present a generic transform from any Σ-protocol into a deniable zero-knowledge proof of knowledge.

A Σ-protocol is a special 3-move honest-verifier zero-knowledge proof of knowledge for a relation R. We recall that for a pair $(x, w) \in R$, x is a common input for $\mathbf{P}$ and $\mathbf{V}$ and w is a private input for $\mathbf{P}$. We usually denote the transcript (i.e., the three exchanged messages) by (a, e, z) and call the transcript "accepting" if an honest verifier $\mathbf{V}$ would accept the corresponding interactive proof execution. In Σ-protocols, e is a random bit-string which is (for the honest verifier) independent from a.

To fully characterize a Σ-protocol, we specify the algorithms which generate a and z, the domain of e, and the verifying algorithm executed by the verifier at the end. Let us denote them by PR_1, PR_2, $\{0,1\}^t$, and VER respectively. Finally, a Σ-protocol can be described formally as depicted on Fig. 3 where the notation ϖ_{P} (resp. ϖ_{V}) represents the random tape of the prover $\mathbf{P}$ (resp. verifier $\mathbf{V}$).

$$\begin{array}{rcl}
\mathbf{P}(w; \varpi_{\mathrm{P}}) & (x) & \mathbf{V}(\cdot; \varpi_{\mathrm{V}}) \\
a = \mathsf{PR}_1(x, w; \varpi_{\mathrm{P}}) & \xrightarrow{\quad a \quad} & \\
 & \xleftarrow{\quad e \quad} & e = \mathsf{trunc}_t(\varpi_{\mathrm{V}}) \\
z = \mathsf{PR}_2(x, w, e; \varpi_{\mathrm{P}}) & \xrightarrow{\quad z \quad} & b = \mathsf{VER}(x, a, e, z)
\end{array}$$

Fig. 3. A Generic Σ-protocol

In addition to the above restrictions, a Σ-protocol must achieve efficiency and completeness following Def. 2, and must satisfy the two following conditions:
Special Soundness. For any $x \in L_R$ and any two accepting transcripts on input x, (a, e, z), (a, e', z') with $e \neq e'$, there exists a polynomial-time extractor $\mathsf{Ext}(x, a, e, e', z, z')$ which outputs a bit-string w such that $(x, w) \in R$.
Special HVZK. There exists a polynomial-time simulator Sim which for any x and a random e outputs a and z such that (a, e, z) has an identical probability distribution to the transcript generated by $\mathbf{P}$ and $\mathbf{V}$ on input x.

The special soundness (resp. special HVZK) guarantees that a Σ-protocol is sound (resp. HVZK). We define a weaker notion as follows.

Definition 7 ($\kappa(x)$-weak Σ-protocol). *Let κ be a real function. A $\kappa(x)$*-weak *Σ*-protocol *is a Σ-protocol with the special soundness property modified as follows:*

For any $x \in L_R$, any a, any $e \in \{0,1\}^t$, there exists a unique z such that $\mathsf{VER}(x, a, e, z) = 1$. *Denote $z = \mathsf{Resp}(x, a, e)$.*
There exists a polynomial-time algorithm Ext *such that for any $x \in L_R$, any a, and any $e \in \{0,1\}^t$, we have*

$$\Pr_{e' \in_{\mathsf{u}} \{0,1\}^t}[(x, \mathsf{Ext}(x, a, e, e', \mathsf{Resp}(x, a, e), \mathsf{Resp}(x, a, e'))) \in R] \geq 1 - \kappa(x)$$

Special soundness is achieved for $\kappa(x) = 2^{-t}$. $\kappa(x)$-weak Σ-protocols are sound with soundness error $\kappa(x)$. This comes from a simplified version of the proof of Th. 9 below.

Here we give two examples of $\kappa(x)$-weak Σ-protocols. The first example is the Guillou-Quisquater (GQ) protocol [30,29]. Let $N = pq$, e, and d be respectively an RSA modulus, the public and private exponents. For simplicity we assume that e is prime. (In practice, RSA keys use $e = 3$ or $e = 65537$). The GQ protocol allows to prove the knowledge of x such that $X = x^e \bmod N$, see Fig. 4a. The second example is from Schnorr [42,43]. Let g be the generator of a group G of prime order q. The Schnorr protocol allows to prove the knowledge of the discrete logarithm x in G of the element $X = g^x$, see Fig. 4b.

$$\begin{array}{rcl} \mathbf{P}(x) & (N,e,X) & \mathbf{V} \\ \text{pick } y \in_{\mathsf{u}} \mathbb{Z}_N^* & & \\ Y = y^e & \xrightarrow{\quad Y \quad} & \\ & \xleftarrow{\quad r \quad} & \text{pick } r \in_{\mathsf{u}} \{0,1\}^t \\ z = yx^r & \xrightarrow{\quad z \quad} & \text{check } z^e \overset{?}{=} YX^r \end{array}$$

(a)

$$\begin{array}{rcl} \mathbf{P}(x) & (g,q,X) & \mathbf{V} \\ \text{pick } y \in_{\mathsf{u}} \mathbb{Z}_q & & \\ Y = g^y & \xrightarrow{\quad Y \quad} & \\ & \xleftarrow{\quad r \quad} & \text{pick } r \in_{\mathsf{u}} \{0,1\}^t \\ z = y + rx \bmod q & \xrightarrow{\quad z \quad} & \text{check } g^z \overset{?}{=} YX^r \end{array}$$

(b)

Fig. 4. The GQ (a) and Schnorr (b) Protocols

Theorem 8. *Let t be the bit-length of the second move. The GQ protocol with prime exponent e is a $\frac{\lceil \frac{2^t}{e} \rceil}{2^t}$-weak Σ-protocol. The Schnorr protocol in a group of prime order is a 2^{-t}-weak Σ-protocol.*

Proof. The case of the Schnorr protocol is well known, so we concentrate on the GQ protocol.

Clearly we can define PR_1, PR_2, VER. Special HVZK is straightforward. Here we only need to prove that it is $\kappa(x)$-weak. Note that given the two first moves (Y,r), there exists a unique third move (z) for which $\mathbf{V}$ will accepts. It remains to prove the soundness and for that we should build an extractor Ext which outputs the witness given any $(I, Y, r_1, \mathsf{Resp}(I, Y, r_1))$ and a random $(r_2, \mathsf{Resp}(I, Y, r_2))$ with $I = (N, e, X)$.

Given (N, e, X), Y, r_1, r_2, z_1, z_2 such that $z_1^e = YX^{r_1} \pmod N$ and $z_2^e = YX^{r_2} \pmod N$ if $\gcd(r_1 - r_2, e) = 1$ we can find some integers a and b such that $ae + b(r_1 - r_2) = 1$ by using the Extended Euclid algorithm and then can compute $x = X^a z_1^b z_2^{-b} \bmod N$ which satisfies

$$x^e = X^{ae}(z_1^e)^b(z_2^e)^{-b} = X^{ae}(YX^{r_1})^b(YX^{r_2})^{-b} = X^{ae+b(r_1-r_2)} = X \pmod N$$

so a valid witness is extracted. Clearly, the GQ protocol is $\kappa(x)$-weak where $\kappa(x) = \max_{r_1} \Pr_{r_2}[\mathsf{gcd}(r_1 - r_2, e) \neq 1]$ and we find that $\kappa(x) \leq \frac{\lceil \frac{2^t}{e} \rceil}{2^t}$ since

$$\kappa(x) = \sum_{k=0}^{2^t-1} 1_{\mathsf{gcd}(r_1-k,e)\neq 1} \Pr[r_2 = k] = \frac{1}{2^t}\#\{\text{multiples of } e \text{ in } [r_1, r_1 + 2^t - 1]\}.$$

$\square$

We transform a HVZK protocol into a deniable ZK protocol by adding a commitment step. This idea was proposed by Goldreich-Micali-Wigderson [25] and then reused by Goldreich-Kahan [24]. They prove that it is possible to achieve ZK in the standard model with a polynomial round complexity. Here, we want to prove that it is possible to achieve deniable ZK in the CRS or RO models with only 4 moves. At the same time, we achieve ZK in the standard model with one extra move. The extra move is necessary for sending the fresh public-key which replaces the common reference string. Note that Cramer-Damgård-MacKenzie [12] proposed a transform to achieve ZK but with a bigger round complexity while Damgård [13] proposed an efficient construction but without deniability. Clearly, for our application, deniability is mandatory in the CRS and RO models.

$$\begin{array}{rcl}
\mathbf{P}(w;\varpi_{\mathrm{P}}) & (x) & \mathbf{V}(\cdot;\varpi_{\mathrm{V}}) \\
R \leftarrow \mathsf{random}(\varpi_{\mathrm{P}}) & & \\
(K_p, K_s) \leftarrow \mathsf{setup}(1^\lambda, R) & \xrightarrow{\quad K_p \quad} & e \in_{\mathsf{u}} \{0,1\}^t,\ r \leftarrow \mathsf{random}(\varpi_{\mathrm{V}}) \\
 & \xleftarrow{\quad c \quad} & c = \mathsf{com}(K_p, e, r) \quad (\text{resp. } c = H(e,r)) \\
a = \mathsf{PR}_1(x, w; \varpi_{\mathrm{P}}) & \xrightarrow{\quad a \quad} & \\
\left(\text{resp. } c \overset{?}{=} H(e,r)\right) \quad c \overset{?}{=} \mathsf{com}(K_p, e, r) & \xleftarrow{\quad e\|r \quad} & \\
z = \mathsf{PR}_2(x, w, e; \varpi_{\mathrm{P}}) & \xrightarrow{\quad z\|R \quad} & (\widehat{K}_p, \widehat{K}_s) \leftarrow \mathsf{setup}(1^\lambda, R) \\
 & & \widehat{K}_p \overset{?}{=} K_p \\
 & & b = \mathsf{VER}(x, a, e, z)
\end{array}$$

Fig. 5. A Generic Transform of Σ-protocol

The protocol in the standard model is depicted on Fig. 5. Clearly, the prover should be ensured that nobody knows the trapdoor K_s. Consequently, the prover generates the key pair himself, he gives the public key on the first (extra) move and the private key on the last one.

Theorem 9. *Let $\mathcal{C}$ be a trapdoor commitment scheme, π be a $\kappa(x)$-weak Σ-protocol, and π' be its generic transform as depicted on Fig. 5.*

For any arbitrary large integer k, π' is a zero-knowledge proof of knowledge in the standard model with soundness error $\kappa'(x) = \max\left(\kappa(x), 1/|x|^k\right)$.

Clearly, if the trapdoor K_s is known by the verifier, the protocol remains honest-verifier zero-knowledge (this is essentially the Σ-protocol) but loses deniability. Indeed, a malicious verifier could take $e = \mathsf{OW}(a)$ and open c to e. The response z would become a transferable proof following the Fiat-Shamir paradigm [22] to transform interactive proofs into non-interactive ones. Thus, the K_p key should be trusted by the prover who believes that the verifier does not know the corresponding private key. In practice, a costless pragmatic solution could consist of using a hash function at the place of the commitment. This is essentially the instantiated variant with RO commitment scheme.

Theorem 10. *Let $\mathcal{C}$ be a trapdoor commitment scheme (resp. the RO commitment scheme). Let π be a $\kappa(x)$-weak Σ-protocol and let π' be its generic transform as depicted on Fig. 5 where K_p is the public key as setup in the commitment scheme (resp. where H is a random oracle).*

1. Ext' picks ϖ_{P} and set up $\mathbf{P}^*$ with ϖ_{P}.
2. Ext' plays the role of the verifier and runs a complete protocol with $\mathbf{P}^*$ who gives the trapdoor at the end. If this protocol does not fail (event A_1), this defines a first transcript $c, a, e_1\|r_1, z_1\|K_s$ such that $\mathsf{VER}(x, a_1, e_1, z_1, \varpi_p)$ outputs 1.
3. Ext' picks e_2 and computes $r_2 \leftarrow \mathit{equivocate}(K_s, c, e_2)$. Ext' runs another complete protocol with $\mathbf{P}^*$ set up with the same ϖ_p and uses messages c and $e_2\|r_2$. If this protocol does not fail (event A_2), this defines a second transcript $c, a, e_2\|r_2, z_2$ such that $\mathsf{VER}(x, a, e_2, z_2)$ outputs 1.
4. If one of the two protocols failed, Ext' aborts. Otherwise, using Ext with inputs (a, e_1, z_1) and (a, e_2, z_2), Ext' recovers w such that $(x, w) \in R$.

Fig. 6. The Knowledge Extractor Ext

For any arbitrary large integer k, π' is a deniable zero-knowledge proof of knowledge in the CRS model (resp. in the RO model) with soundness error $\kappa'(x) = \max\left(\kappa(x), 1/|x|^k\right)$.

Proof (Th. 9 and Th. 10). Efficiency and completeness of the protocols of Fig. 5 are trivial. So, we concentrate in proving the properties of soundness and deniable zero-knowledge.

Soundness. Let k be an arbitrary positive large integer and let $\mathbf{P}^*$ be any malicious prover which passes the protocol π' for x with an honest verifier $\mathbf{V}$ with probability $\varepsilon(x)$. Let $\kappa'(x)$ be the soundness error of the protocol π'. By Def. 2 it is assumed that $\varepsilon(x) > \kappa'(x)$.

Recall that for any $x \in L_R$ given two random accepting transcripts (a, e_1, z_1) and (a, e_2, z_2) there exists a polynomial-time extractor Ext which outputs the witness w such that $(x, w) \in R$ with probability $1 - \kappa(x)$ (over e_2).

We construct the extractor Ext' as described on Fig. 6. Thanks to the property of equiv, the extractor Ext' simulates perfectly a honest verifier for the malicious prover $\mathbf{P}^*$. Given ϖ_{P} and c, both protocols are independent and succeed with the same probability, let us denote it by $\Pr[A_j|\varpi_{\mathrm{P}}, c] = p_{\varpi_{\mathrm{P}},c}$ for $j = 1, 2$. The expected value of $p_{\varpi_{\mathrm{P}},c}$ over the random choice of ϖ_{P} and c is $\varepsilon(x)$. No matter whether A_j holds, let z_j be the unique z such that $\mathsf{VER}(x, a, e_j, z_j) = 1$. Let B the event that $\mathsf{Ext}(a, e_1, e_2, z_1, z_2)$ succeeds, i.e. $\Pr[\neg B] \leq \kappa(x)$. Furthermore, $\Pr[\neg B|A_1] \leq \kappa(x)$. We want to compute $\Pr[A_1 \wedge A_2 \wedge B]$ and we have :

$$\Pr[A_1 \wedge A_2 \wedge B|\varpi_{\mathrm{P}}, c] = \Pr[A_1 \wedge A_2|\varpi_{\mathrm{P}}, c] - \Pr[A_1 \wedge A_2 \wedge \neg B|\varpi_{\mathrm{P}}, c] \ .$$

Focusing on the right term, we have $\Pr[A_1 \wedge A_2 \wedge \neg B|\varpi_{\mathrm{P}}, c] \leq \Pr[A_1 \wedge \neg B|\varpi_{\mathrm{P}}, c] \leq p_{\varpi_{\mathrm{P}},c}\kappa(x)$, while the other term is $\Pr[A_1 \wedge A_2|\varpi_{\mathrm{P}}, c] = p^2_{\varpi_{\mathrm{P}},c}$, and we obtain $\Pr[A_1 \wedge A_2 \wedge B|\varpi_{\mathrm{P}}, c] \geq p_{\varpi_{\mathrm{P}},c}(p_{\varpi_{\mathrm{P}},c} - \kappa(x))$. Finally, we compute the expected value over the ϖ's and c's and using the Jensen's inequality on the function $x \mapsto x^2$, we obtain $\Pr[A_1 \wedge A_2 \wedge B] \geq \varepsilon(x)(\varepsilon(x) - \kappa(x))$. We conclude that the average number of running time of Ext' before it succeeds is $\frac{1/\varepsilon(x)}{\varepsilon(x)-\kappa(x)}$. Following Def. 2, it suffices to prove that $\frac{1/\varepsilon(x)}{\varepsilon(x)-\kappa(x)} \leq \frac{|x|^k}{\varepsilon(x)-\kappa'(x)}$ for any $\varepsilon(x) > \kappa'(x)$. It is the case when $\kappa'(x) = \max\left(\kappa(x), 1/|x|^k\right)$.

In the case of the CRS model, the extractor Ext' can be assumed to *know the trapdoor of the commitment.* In the standard model (Fig. 5), Ext' learns the

1. Sim launches $\mathbf{V}^*$ with a fresh random tape ϖ_V and receives c from $\mathbf{V}^*$.
2. Sim picks a random a using the same distribution than $\mathsf{PR}_1(\cdot)$. Thanks to the special HVZK property, this can be simulated by using Sim' and obtaining (a, e_0, z_0). Sim then gives a to $\mathbf{V}^*$.
3. Sim receives e and r from $\mathbf{V}^*$ and checks $c = \mathsf{commit}(K_p, e, r)$.
   - If c is not valid, Sim stops the simulation and releases the transcript $(\varpi_\mathrm{V}, K_p, x, a)$.
   - Otherwise, i.e., if c is valid,
     (a) Sim rewinds $\mathbf{V}^*$ with the same random tape ϖ_V and receives the same c from $\mathbf{V}^*$ since ϖ_V is unchanged. Sim can thus guess that c will open to e.
     (b) Sim gives x and e to the simulator Sim' described above in order to obtain a "good" transcript (a', e, z'). Sim sends a' to $\mathbf{V}^*$.
     (c) Sim receives e' and r' from $\mathbf{V}^*$ and checks $c = \mathsf{com}(K_p, e', r')$.
       - If c is not valid, Sim goes back to step 3a.
       - Otherwise, i.e., if c is valid
         i. If $e \neq e'$ (double opening of c), Sim aborts.
         ii. Sim finishes by yielding $(\varpi_\mathrm{V}, K_p, x, a', z')$ of the last interaction with $\mathbf{V}^*$.

Fig. 7. The Simulator Sim

trapdoor when event A_1 holds. In the case of the RO commitment, the proof is essentially the same: Ext' creates two entries $\mathsf{H}(e_1, r_1) = \mathsf{H}(e_2, r_2) = c$ in the H table and executes both protocols by using only one entry. If by any chance $\mathbf{P}^*$ queries H with the other, the extraction fails. But this happens with negligible probability.

Deniable Zero-Knowledge. First, note that in the standard model deniable zero-knowledge (dZK) and zero-knowledge (ZK) are equivalent. So, in this proof we will show that all three protocols are dZK. This will imply that the protocol of Fig. 5 is ZK in the standard model.

We need to build a simulator able to simulate the interactions with any verifier $\mathbf{V}^*$ as described in Def. 2. Note that in the CRS model, the simulator Sim is not allowed to generate the common reference string. Let $K_p = \mathsf{crs}$ be any uniformly distributed random string. K_p is given to both, to Sim and to $\mathbf{V}^*$.

Recall that given any $x \in L_R$ and a random e there exists a polynomial-time simulator Sim' which outputs a transcript (a, e, z) which has identical probability distribution than a transcript generated by $\mathbf{P}$ and $\mathbf{V}$ on input x.

We construct the simulator Sim as depicted on Fig. 7. Clearly, Sim always returns a complete protocol view from $\mathbf{V}^*$. It is either of type I $(\varpi_\mathrm{V}, K_p, x, a)$ or of type II $(\varpi_\mathrm{V}, K_p, x, a', z')$. The ϖ_V distribution is perfect as well as the view of type I. Let $A_{\varpi_\mathrm{V}, K_p, x}$ be the set of all a such that $\mathbf{V}^*(\varpi_\mathrm{V}, K_p, x, a)$ returns a valid c. The distribution of a' is the marginal distribution from PR_1 conditioned to set $A_{\varpi_\mathrm{V}, K_p, x}$. So, it is perfectly simulated as well. Finally, the unique z' is well simulated (the negligible probability of breaking the commitment has a negligible influence on the distribution) so we have a computationally indistinguishable simulator.

We still have to show that the average number of rewindings is polynomial. Let ϖ_V be a fix random tape of $\mathbf{V}^*$. Given $x \in L_R$, for w s.t. $(x, w) \in R$ we consider $\mathbf{V}^*_{\varpi_\mathrm{V}}$ interacting with $\mathbf{P}(x, w)$. We denote by $p_{\varpi_\mathrm{V}}(x)$ the probability that the commitment is incorrectly opened to $\mathbf{P}$. Since the distribution of a can be simulated, $p_{\varpi_\mathrm{V}}(x)$ does not depend on w. The number of executions of $\mathbf{P}$ is 1 with probability $1 - p_{\varpi_\mathrm{V}}(x)$ and $1 + \frac{1}{p_{\varpi_\mathrm{V}}(x)}$ with probability $p_{\varpi_\mathrm{V}}(x)$, so it is 2 on average. This proves that the simulator runs in expected polynomial time. $\square$

6 ONTAP in Practice

6.1 ONTAP with Generic RSA Signature

We propose ONTAP-RSA: an ONTAP scheme which is generic for RSA-based signatures. It is based on a zero-knowledge variant of the GQ protocol.

Consider $h \leftarrow \mathsf{H}_{\mathsf{seed}}(m)$ be a formatting function and $b = \mathsf{V}(h, m)$ be a 0/1 check function returning 1 iff the formatted h is consistent with m.

Definition 11 (Generic RSA). *A generic RSA signature works in a group $\mathbb{Z}_N^*$ with $N \leftarrow pq$ and p, q are two $\frac{k}{2}$-bit random prime numbers. Let e, d such that $ed \equiv 1 \pmod{\varphi(N)}$ and e is prime (since several variant are commonly used we do not specify further the generation algorithm). The private key is $K_s \leftarrow (N, d)$ and the corresponding public key is $K_p \leftarrow (N, e)$*

The signature of message m consists of the tuple $\sigma = (\sigma_s, \sigma_p)$. The algorithm picks a random seed*, computes the formatted message $\sigma_p \leftarrow \mathsf{H}_{\mathsf{seed}}(m)$, the signature $\sigma_s = \sigma_p^d \bmod N$ (using the private key K_s), and outputs σ_p and σ_s.*

There exists a verification algorithm $\mathsf{verify}(K_p, m, \sigma_p, \sigma_s)$ *which outputs 1 if the signature is valid, i.e. $\mathsf{V}(\sigma_p, m) = 1$ and $\sigma_s^e \bmod N = \sigma_p$, and 0 otherwise.*

Clearly, the PKCS#1v1.5, ISO/IEC 9796, RSA-PSS standards all fit into this category.

The iProof protocol works as depicted on Fig. 8a. Note that K_p is the public-key related to the signature scheme while crs is the one related to the commitment scheme. The way to adapt to the plain model or random oracle model is straightforward.

Theorem 12. *Assume that the RSA-based signature is EF-CMA-secure and that the* $\mathsf{com}(\cdot)$ *is a trapdoor commitment scheme in the CRS model (resp. RO commitment scheme). The digital signature scheme added to the signature proof* iProof *of Fig. 8a forms a ONTAP scheme as defined in Def. 3 in the CRS model (resp. RO model). The soundness error is $\lceil \frac{2^t}{e} \rceil / 2^t$.*

With an extra round we obtain an ONTAP in the standard model (see Fig. 5).

Proof. Clearly, there exists an algorithm $\mathsf{simulate}(K_p, m)$ which outputs a σ_p computationally indistinguishable from the one generated by $\mathsf{sign}(K_s, m)$, i.e. $\sigma_p \leftarrow \mathsf{H}_{\mathsf{seed}}(m)$. Thanks to Th. 6, we only need to prove that the signature scheme is unforgeable and the protocol is deniable zero-knowledge. Unforgeability is already assumed. Efficiency and completeness of iProof are trivial. Soundness and deniable zero-knowledge properties of iProof are proven by Th. 8 and 9. □

6.2 ONTAP with Generic ElGamal Signature

Definition 13 (Generic ElGamal). *A* generic *ElGamal signature scheme works in a group $\mathbb{G}$ with a generator $g \in \mathbb{G}$ with order q. The private key is $K_s = x \in_{\mathsf{u}} \mathbb{Z}_q$ and the corresponding public key is $K_p = y = g^x$.*

$$\begin{array}{rcl}
\mathbf{P}(\sigma_s) & (\mathsf{crs}, K_p, m) & \mathbf{V} \\
\sigma_p = \sigma_s^e \bmod N & & \text{pick } \tilde{r} \\
 & & \text{pick } r \in_{\mathsf{u}} \{0,1\}^t \\
\text{pick } y \in_{\mathsf{u}} \mathbb{Z}_N^* & \xleftarrow{\quad c \quad} & c = \mathsf{com}(\mathsf{crs}, r, \tilde{r}) \\
Y = y^e \bmod N & \xrightarrow{\; Y \| \sigma_p \;} & \mathsf{V}(\sigma_p, m) \overset{?}{=} 1 \\
c \overset{?}{=} \mathsf{com}(\mathsf{crs}, r, \tilde{r}) & \xleftarrow{\quad r \| \tilde{r} \quad} & \\
z \leftarrow y \sigma_s^r \bmod N & \xrightarrow{\quad z \quad} & z^e \overset{?}{=} Y \sigma_p^r \pmod N
\end{array}$$

(a)

$$\begin{array}{rcl}
\mathbf{P}(\sigma_p, \sigma_s) & (\mathsf{crs}, K_p, m) & \mathbf{V} \\
 & & \text{pick } \tilde{r} \\
 & & \text{pick } r \in_{\mathsf{u}} \{0,1\}^t \\
\text{pick } \ell \in_{\mathsf{u}} [0, q-1] & \xleftarrow{\quad c \quad} & c = \mathsf{com}(\mathsf{crs}, r, \tilde{r}) \\
a = u^\ell \bmod p & \xrightarrow{\; a \| \sigma_p \;} & \mathsf{ver}(K_p, m, \sigma_p) \overset{?}{=} 1 \\
c \overset{?}{=} \mathsf{com}(\mathsf{crs}, r, \tilde{r}) & \xleftarrow{\quad r \| \tilde{r} \quad} & \sigma_p = (u, v, \xi) \\
z = \ell + r \cdot s \bmod q & \xrightarrow{\quad z \quad} & u^z \overset{?}{=} a v^r \pmod p
\end{array}$$

(b)

Fig. 8. The iProof Protocols for ONTAP-RSA (a) and ONTAP-ElGamal (b)

The signature of a message m consists of the tuple $\sigma = (u, v, \xi, s) \leftarrow \mathsf{sign}(K_s, m)$. This tuple is split in two parts: a part $\sigma_p = (u, v, \xi)$ which can be perfectly simulated without K_s and a part $\sigma_s = s$.

There exists a verification algorithm $\mathsf{verify}(K_p, m, \sigma_p, \sigma_s)$ which outputs 1 if $u^s = v \ \wedge\ \mathsf{ver}(K_p, m, \sigma_p) = 1$ and 0 otherwise for some function ver.

ElGamal [21] (with a group of prime order), Schnorr [42,43], DSA [19,18], and ECDSA [20] signatures all meet the *generic* ElGamal requirements. Clearly, all of them respect the parameter and key generation. We give briefly four examples:

- The plain ElGamal signature is $u = g^k \bmod p$ for some random k, $v = g^{h(m)} y^{-u} \bmod p$, $\xi = \emptyset$, $s = \frac{h(m) - x\sigma_r}{k} \pmod q$ and the ver algorithm consists in checking $v \overset{?}{=} g^{h(m)} y^{-u} \pmod p$.
- The Schnorr signature is $u = g$, $v = g^s \bmod p$, $\xi = h(g^k \bmod p \| m)$, $s = k + x\xi \bmod q$ and the ver algorithm consists in checking $h(vy^{-\xi} \bmod p \| m) \overset{?}{=} \xi$
- The DSA signature is $u = g^k$ for some random k, $v = g^{h(m)} y^u$, $\xi = \emptyset$, $s = \frac{h(m) + xu}{k} \bmod q$ and the ver algorithm consists in checking $v \overset{?}{=} g^{h(m)} y^u \pmod p$.
- The ECDSA signature works over an elliptic curve with prime order n, with generator G, and with keys $K_s = d \in_{\mathsf{u}} [1, n-1]$ and $K_P = Q = dG$. The ECDSA signature is $u = (u_x, u_y) = kG$, $v = h(m)G + \bar{u}_x Q$, $\xi = \emptyset$, $s = \frac{h(m) + d\bar{u}_x}{k} \bmod n$ and the ver algorithm consists in checking $v \overset{?}{=} h(m)G + \bar{u}_x Q \pmod n$. (Here we use additive notations and the $u_x \mapsto \bar{x}_x$ mapping is defined in [20]).

In order to build a non-transferable signature, instead of revealing the private part of the signature, we will prove that we know it. Clearly, we will use a zero-knowledge proof as before. The required proof of knowledge should allow $\mathbf{P}$ to prove to $\mathbf{V}$ that he knows s such that $u^s = v$. Note that this is the proof of the knowledge of the discrete logarithm. The identification protocol from Schnorr [42,43] is a Σ-protocol when q is prime proving exactly that. Consequently, we applied our generic transform from Th. 9 and we obtain the verification protocol of Fig. 8b which is deniable zero-knowledge in the CRS model. We thus obtain several schemes: ONTAP-ElGamal, ONTAP-Schnorr, ONTAP-DSA, ONTAP-ECDSA, and so on.

Theorem 14. *Assume that the ElGamal-based signature is EF-CMA-secure and that the* $\mathsf{com}(\cdot)$ *is a trapdoor commitment scheme in the CRS model (resp. RO commitment scheme). The digital signature scheme added to the signature proof* iProof *of Fig. 8b forms a ONTAP scheme in the CRS model (resp. RO model). The soundness error is* 2^{-t}.

See proof of Th. 12.

7 Conclusion

We studied the deniability notion in the case of digital signature verification. We proposed a new primitive called ONTAP as an offline non-transferable proof for holding a valid signature. As example, we presented an efficient signature proof for RSA-based and ElGamal-based signatures. Our protocol offers an adequate solution for private data authentication especially in the context of e-passports. It is compatible with all standard signature schemes.

References

1. Asokan, N., Shoup, V., Waidner, M.: Optimistic fair exchange of digital signatures (extended abstract). In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 591–606. Springer, Heidelberg (1998)
2. Asokan, N., Shoup, V., Waidner, M.: Optimistic fair exchange of digital signatures (extended abstract). IEEE Journal on Selected Areas in Communications 18(4), 593–610 (2000)
3. Baek, J., Safavi-Naini, R., Susilo, W.: Universal Designated Verifier Signature Proof (or How to Efficiently Prove Knowledge of a Signature). In: Roy, B. (ed.) ASIACRYPT 2005. LNCS, vol. 3788, pp. 644–661. Springer, Heidelberg (2005)
4. Boyar, J.F., Chaum, D., Damgård, I., Pedersen, T.P.: Convertible Undeniable Signatures. In: Menezes, A., Vanstone, S.A. (eds.) CRYPTO 1990. LNCS, vol. 537, pp. 189–205. Springer, Heidelberg (1991)
5. Boyar, J.F., Kurtz, S.A., Krentel, M.W.: A discrete logarithm implementation of perfect zero-knowledge blobs. Journal of Cryptology 2(2), 63–76 (1990)
6. Brassard, G., Chaum, D., Crépeau, C.: Minimum disclosure proofs of knowledge. Journal of Computer and System Sciences 37(2), 156–189 (1988)
7. Camenisch, J., Michels, M.: Confirmer Signature Schemes Secure against Adaptive Adversaries. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 243–258. Springer, Heidelberg (2000)
8. Catalano, D., Gennaro, R., Howgrave-Graham, N., Nguyen, P.Q.: Paillier's cryptosystem revisited. In: CCS 2001, pp. 206–214. ACM Press, New York (2001)
9. Chaum, D.: Zero-Knowledge Undeniable Signatures. In: Damgård, I.B. (ed.) EUROCRYPT 1990. LNCS, vol. 473, pp. 458–464. Springer, Heidelberg (1991)
10. Chaum, D.: Designated Confirmer Signatures. In: De Santis, A. (ed.) EUROCRYPT 1994. LNCS, vol. 950, pp. 86–91. Springer, Heidelberg (1995)
11. Chaum, D., van Antwerpen, H.: Undeniable Signatures. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 212–217. Springer, Heidelberg (1990)

12. Cramer, R., Damgård, I., MacKenzie, P.D.: Efficient Zero-Knowledge Proofs of Knowledge Without Intractability Assumptions. In: Imai, H., Zheng, Y. (eds.) PKC 2000. LNCS, vol. 1751, pp. 354–372. Springer, Heidelberg (2000)
13. Damgård, I.: Efficient Concurrent Zero-Knowledge in the Auxiliary String Model. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 418–430. Springer, Heidelberg (2000)
14. Damgård, I.: On Σ-protocols. Lecture Notes (2005)
15. Damgård, I., Pedersen, T.P.: New Convertible Undeniable Signature Schemes. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 372–386. Springer, Heidelberg (1996)
16. Desmedt, Y.: Subliminal-Free Authentication and Signature (Extended Abstract). In: Günther, C.G. (ed.) EUROCRYPT 1988. LNCS, vol. 330, pp. 23–33. Springer, Heidelberg (1988)
17. Dolev, D., Dwork, C., Naor, M.: Nonmalleable cryptography. SIAM Review 45(4), 727–784 (2003)
18. Digital signature standard (DSS). Federal Information Processing Standard, Publication 186-2, U.S. Department of Commerce, National Institute of Standards and Technology (2000)
19. Digital signature standard (DSS). Federal Information Processing Standard, Publication 186, U.S. Department of Commerce, National Institute of Standards and Technology (1994)
20. ANSI X9.62. Public Key Cryptography for the Financial Service Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA). American National Standard Institute. American Bankers Associtaion (1998)
21. ElGamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Transactions on Information Theory 31(4), 469–472 (1985)
22. Fiat, A., Shamir, A.: How to Prove Yourself: Practical Solutions to Identification and Signature Problems. In: Odlyzko, A.M. (ed.) CRYPTO 1986. LNCS, vol. 263, pp. 186–194. Springer, Heidelberg (1987)
23. Gennaro, R., Krawczyk, H., Rabin, T.: RSA-Based Undeniable Signatures. Journal of Cryptology 13(4), 397–416 (2000)
24. Goldreich, O., Kahan, A.: How To Construct Constant-Round Zero-Knowledge Proof Systems for NP. Journal of Cryptology 9(3), 167–189 (1996)
25. Goldreich, O., Micali, S., Wigderson, A.: Proofs that Yield Nothing But Their Validity or All Languages in NP Have Zero-Knowledge Proof Systems. Journal of the ACM 38(1), 691–729 (1991)
26. Goldwasser, S., Micali, S., Rackoff, C.: The Knowledge Complexity of Interactive Proof-Systems. In: STOC 1985, pp. 291–304. ACM Press, New York (1985)
27. Goldwasser, S., Micali, S., Rackoff, C.: The Knowledge Complexity of Interactive Proof Systems. SIAM Journal on Computing 18(1), 186–208 (1989)
28. Goldwasser, S., Micali, S., Rivest, R.L.: A digital signature scheme secure against adaptive chosen-message attacks. SIAM Journal on Computing 17(2), 281–308 (1988)
29. Guillou, L.C., Quisquater, J.-J.: A Practical Zero-Knowledge Protocol Fitted to Security Microprocessor Minimizing Both Trasmission and Memory. In: Günther, C.G. (ed.) EUROCRYPT 1988. LNCS, vol. 330, pp. 123–128. Springer, Heidelberg (1988)
30. Guillou, L.C., Quisquater, J.-J.: A "Paradoxical" Identity-Based Signature Scheme Resulting from Zero-Knowledge. In: Goldwasser, S. (ed.) CRYPTO 1988. LNCS, vol. 403, pp. 216–231. Springer, Heidelberg (1990)

31. Jakobsson, M., Sako, K., Impagliazzo, R.: Designated Verifier Proofs and Their Applications. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 143–154. Springer, Heidelberg (1996)
32. Li, J., Wang, Y.: Universal Designated Verifier Ring Signature (Proof) Without Random Oracles. In: Zhou, X., Sokolsky, O., Yan, L., Jung, E.-S., Shao, Z., Mu, Y., Lee, D.C., Kim, D.Y., Jeong, Y.-S., Xu, C.-Z. (eds.) EUC Workshops 2006. LNCS, vol. 4097, pp. 332–341. Springer, Heidelberg (2006)
33. Monnerat, J., Vaudenay, S.: Generic Homomorphic Undeniable Signatures. In: Lee, P.J. (ed.) ASIACRYPT 2004. LNCS, vol. 3329, pp. 354–371. Springer, Heidelberg (2004)
34. Monnerat, J., Vaudenay, S., Vuagnoux, M.: About Machine-Readable Travel Documents – Privacy Enhancement Using (Weakly) Non-Transferable Data Authentication. In: RFIDSEC 2007 (2007)
35. Machine Readable Travel Documents. Development of a Logical Data Structure — LDS For Optional Capacity Expansion Technologies. Version 1.7 (2004), `http://www.icao.int/mrtd/download/technical.cfm`
36. Machine Readable Travel Documents. PKI for Machine Readable Travel Documents offering ICC Read-Only Access. Version 1.1 (2004), `http://www.icao.int/mrtd/download/technical.cfm`
37. Ogata, W., Kurosawa, K., Heng, S.-H.: The Security of the FDH Variant of Chaum's Undeniable Signature Scheme. In: Vaudenay, S. (ed.) PKC 2005. LNCS, vol. 3386, pp. 328–345. Springer, Heidelberg (2005)
38. Okamoto, T., Ohta, K.: How to Utilize the Randomness of Zero-Knowledge Proofs. In: Menezes, A., Vanstone, S.A. (eds.) CRYPTO 1990. LNCS, vol. 537, pp. 456–475. Springer, Heidelberg (1991)
39. Paillier, P.: Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 223–238. Springer, Heidelberg (1999)
40. Pass, R.: On Deniability in the Common Reference String and Random Oracle Model. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 316–337. Springer, Heidelberg (2003)
41. Pass, R.: Alternative Variants of Zero-Knowledge Proofs. Licentiate Thesis (2004)
42. Schnorr, C.-P.: Efficient Identification and Signatures for Smart Cards. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 239–252. Springer, Heidelberg (1990)
43. Schnorr, C.-P.: Efficient Signature Generation by Smart Cards. Journal of Cryptology 4(3), 161–174 (1991)
44. Shahandashti, S.F., Safavi-Naini, R.: Construction of Universal Designated-Verifier Signatures and Identity-Based Signatures from Standard Signatures. In: Cramer, R. (ed.) PKC 2008. LNCS, vol. 4939, pp. 121–140. Springer, Heidelberg (2008)
45. Shahandashti, S.F., Safavi-Naini, R., Baek, J.: Concurrently-secure credential ownership proofs. In: ASIACCS 2007, pp. 161–172. ACM Press, New York (2007)
46. Steinfeld, R., Bull, L., Wang, H., Pieprzyk, J.: Universal Designated-Verifier Signatures. In: Laih, C.-S. (ed.) ASIACRYPT 2003. LNCS, vol. 2894, pp. 523–542. Springer, Heidelberg (2003)
47. Vaudenay, S.: E-Passport Threats. IEEE Security and Privacy Magazine 5(6), 61–64 (2007)
48. Vaudenay, S., Vuagnoux, M.: About Machine-Readable Travel Documents. In: ICS 2007. LNCS. Springer, Heidelberg (2007)