Efficient and Non-Malleable Proofs of Plaintext Knowledge and Applications*

(Extended Abstract)

Jonathan Katz

Dept. of Computer Science, University of Maryland, College Park, MD
(Work done while at Columbia University)
`jkatz@cs.umd.edu`

Abstract. We describe efficient protocols for *non-malleable* (interactive) proofs of plaintext knowledge for the RSA, Rabin, Paillier, and El Gamal encryption schemes. We also highlight some important applications of these protocols:

– *Chosen-ciphertext-secure, interactive encryption.* In settings where both parties are on-line, an interactive encryption protocol may be used. We construct chosen-ciphertext-secure interactive encryption schemes based on any of the schemes above. In each case, the improved scheme requires only a small overhead beyond the original, semantically-secure scheme.

– *Password-based authenticated key exchange.* We derive efficient protocols for password-based key exchange in the public-key model [28, 5] whose security may be based on any of the cryptosystems mentioned above.

– *Deniable authentication.* Our techniques give the first efficient constructions of deniable authentication protocols based on, e.g., the RSA or computational Diffie-Hellman assumption.

Of independent interest, we consider the concurrent composition of *proofs of knowledge*; this is essential to prove security of our protocols when run in an asynchronous, concurrent environment.

1 Introduction

Given an instance of an encryption scheme with public key pk and secret key sk, a *proof of plaintext knowledge* (PPK) allows a sender $\mathcal{S}$ to prove knowledge of the contents m of some ciphertext $C = \mathcal{E}_{pk}(m)$ to a receiver $\mathcal{R}$ (a formal definition appears in Section 2). To be useful, a PPK should also ensure that no information about m is revealed, either to the receiver — in case the receiver does not have sk — or to an eavesdropper. As we show here, PPKs have applications to chosen-ciphertext-secure (IND-CCA2) public-key encryption schemes [31, 35], password-based authentication and key exchange (password-AKE) protocols in the public-key model [28, 5], and deniable authentication [15, 17].

* The full version of this work appears in [29].

Of course, PPKs may be achieved using generic zero-knowledge (ZK) proofs of knowledge [20, 25, 3]; similarly [21, 12], non-interactive PPKs are possible assuming appropriate public parameters are included with *pk*. For the Rabin, RSA, Paillier, or El-Gamal [34, 36, 33, 19] encryption schemes, the well-known Σ-protocols [7] for these schemes (e.g., [32, 26, 8, 38]) may be adapted to give PPKs, although modifications are needed to ensure security against a cheating verifier. For the applications listed above, however, these solutions are not sufficient; the following considerations additionally need to be taken into account:

- Non-malleability. An active adversary $\mathcal{M}$ may be controlling all communication between the honest parties in a classic "man-in-the-middle" attack. We then need to ensure that the adversary cannot divert the proof of knowledge being given by $\mathcal{S}$ to $\mathcal{R}$. For example, $\mathcal{S}$ may be giving a PPK of C', yet $\mathcal{M}$ might be able to change this to a PPK of some C *even though* $\mathcal{M}$ *has no knowledge of the real decryption of* C.
- Concurrency. A receiver may be interacting asynchronously and concurrently with many senders who are simultaneously giving PPKs for different ciphertexts. The protocol should remain secure even in this environment.

Generic solutions to the above problems exist (cf. Section 1.2). However, these solutions — particularly in the case of non-malleability — are extremely inefficient. Our main contribution is to give very efficient non-malleable PPKs for the commonly-used cryptosystems mentioned above; namely, Rabin, RSA, Paillier, and El Gamal. As discussed in the following section, we furthermore show how these PPKs yield efficient protocols (even in a concurrent setting) for a number of applications.

1.1 Applications and Results

We describe some applications of our non-malleable PPKs, and also the notion of *concurrent proofs of knowledge* which arises when proving security for these applications.

Interactive public-key encryption. When a sender and receiver are both on-line, it may be perfectly acceptable to use an interactive encryption protocol.[1] Known *non-interactive* IND-CCA2 encryption schemes are either impractical [15, 37] or are based on specific, *decisional* assumptions [10, 11]. It is therefore reasonable to look for efficient constructions of *interactive* IND-CCA2 encryption schemes based on (potentially weaker) *computational* assumptions.

Interactive encryption schemes based on PPKs have been proposed previously [22, 27, 24]; these, however, achieve only non-adaptive CCA1 security. Interactive encryption was also considered by [15], who give a generic and relatively efficient IND-CCA2 scheme. This scheme requires a signature from the receiver making it unsuitable for use in some applications (see below). Moreover, their protocol

[1] Note that interaction may be taking place already (e.g., to establish a TCP connection) as part of the larger protocol in which encryption is taking place.

requires the receiver — for each encrypted message — to (1) compute an existentially unforgeable signature and (2) run the key-generation algorithm for a public-key encryption scheme. Our protocols, optimized for particular number-theoretic assumptions, are more efficient still.

Using the non-malleable PPKs presented here, we construct interactive encryption schemes secure against chosen-ciphertext attacks in the standard model. Their efficiency is comparable to known schemes [10, 11]; in fact, our protocols require only a small computational overhead beyond a basic, semantically-secure scheme. The security of our protocols may be based on a variety of computational assumptions, such as RSA, the composite residuosity assumption [33], or the hardness of factoring.

Password-based authentication and key exchange. Interactive encryption becomes an even more appealing solution when encryption is used within a larger, already-interactive protocol. For example, consider password-based authentication and key exchange in the public-key model [28, 5]. In this setting, a client and a server share a weak password which they use to authenticate each other and to derive a key for securing future communication; additionally, the client knows the server's public key. Since the password is short, off-line dictionary attacks must be explicitly prevented. Previous work [28, 5] gives elegant, interactive[2] protocols for securely realizing this task using any IND-CCA2 public-key cryptosystem; our techniques allow the first efficient realization of these protocols based on, e.g., the factoring or RSA assumptions.

Deniable authentication. A deniable authentication protocol [15, 17, 18, 16] allows a prover $\mathcal{P}$ (who has a public key) to authenticate a message to a verifier $\mathcal{V}$ such that the transcript of the interaction cannot be used as evidence that $\mathcal{P}$ took part in the protocol (i.e., $\mathcal{P}$ can later *deny* that the authentication took place). In addition to deniability, we require (informally) that an adversary who interacts with the prover — who authenticates messages $m_1, \ldots, m_\ell$ of the adversary's choice — should be unable to forge the authentication of any message $m' \notin \{m_1, \ldots, m_\ell\}$ for an honest verifier.

Constructions of deniable authentication protocols based on any IND-CCA2 encryption scheme are known [17, 18, 16]. However, these protocols are not secure (in general) when an IND-CCA2 *interactive* encryption scheme is used. For example, the scheme of [15] requires a signature from the prover and hence the resulting authentication protocol is no longer deniable; this problem is pointed out explicitly in [17].

The only previously-known protocol which is both practical and also satisfies the strongest notion of deniability uses the construction of [17] instantiated with the encryption scheme of [10]; security is based on the DDH assumption. We present here the first efficient protocols based on factoring or other assumptions which are secure under the strongest notion of deniability. We also show (in the full version) a deniable authentication protocol based on the CDH assumption

[2] Interaction is *essential* in any authentication protocol to prevent replay attacks.

which is computationally more efficient and requires a shorter public key than the previous best-known solution.

Recently, Naor [30] has suggested applications of the protocols presented here to the problem of deniable ring authentication.

Concurrent proofs of knowledge. When using our PPKs for the applications described above, we must ensure that the protocols remain secure even when run in an asynchronous, concurrent environment. In the context of proofs of knowledge, this requires that witness extraction be possible even when multiple provers are concurrently interacting with a single verifier. Although the issue of concurrency in the context of zero-knowledge proofs has been investigated extensively (following [17]), concurrent *proofs of knowledge* have received much less attention (we are only aware of [23]). We believe the notion of concurrent proofs of knowledge is of independent interest, and hope our work motivates future research on this topic.

1.2 Related Work

Proofs of plaintext knowledge are explicitly considered by Aumann and Rabin [1], who provide a generic solution for *any* public-key encryption scheme. Our solutions differ from theirs in many respects: (1) by working with specific, number-theoretic assumptions we achieve much better efficiency and round complexity; (2) we explicitly consider malleability and ensure that our solutions are non-malleable; (3) our protocols are secure even against a dishonest verifier; and (4) we explicitly handle concurrency and our protocols remain provably-secure under asynchronous, concurrent composition.

Non-malleable zero-knowledge (interactive) proofs were defined and constructed by Dolev, Dwork, and Naor [15]; recently, Barak [2] has given constructions with $O(1)$ round complexity. Sahai [37] and De Santis, et al. [13] provide definitions and constructions for non-malleable NIZK proofs and proofs of knowledge.[3] These solutions are all based on general assumptions and are impractical for giving proofs related to number-theoretic problems of interest.

Non-malleable PPKs were considered, *inter alia*, by Cramer, et al. [8] in the context of communication-efficient multi-party computation; they also present an efficient construction suitable for their application (no definitions of non-malleable PPKs are given in [8]). Here, in addition to constructions, we give formal definitions and also show applications to a number of other cryptographic protocols. Furthermore, we note some important differences between our approaches. First, their solution relies in an essential way upon the fact that the set of participants (i.e., their number and their identities) is fixed and publicly known. Our protocols do not require any notion of user identities and we assume no bound on the number of potential participants. Second, their work considers synchronous communication; here, we allow for asynchronous (concurrent) communication which is more realistic in the context of, e.g., public-key encryption.

[3] Interestingly, ours is the first work to explicitly consider non-malleable *interactive* proofs *of knowledge*.

Universal composability (UC) [6] has been proposed as a general framework in which to analyze the security and composition of protocols; a UC protocol is, in particular, non-malleable. Although the PPKs given here are *not* universally-composable, they are proven secure in alternate — yet standard — models of security. The resulting protocols are more efficient than known UC proofs of knowledge. We note also that security requirements for certain applications (deniable authentication in particular) do not seem to fit readily into the UC framework; for such applications, definitions of the sort used here seem more appropriate.

1.3 Outline of the Paper

In Section 2 we present a definition of PPKs and non-malleable PPKs, and in Section 3 we show and prove secure a construction of a non-malleable PPK for the RSA cryptosystem. (We have obtained similar results for the Rabin, Paillier, and El Gamal cryptosystems [29], but we omit further details in the present abstract.) In Section 4 we briefly sketch some applications of our non-malleable PPKs. Definitions, details, and complete proofs appear in the full version [29].

2 Definitions and Preliminaries

The definitions given here focus on proofs of plaintext knowledge, yet they may be easily extended to proofs of knowledge for general NP-relations. Let $(\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a non-interactive public-key encryption scheme which need not be semantically secure. The encryption of message m under public key pk using randomness r to give ciphertext C is denoted as $C := \mathcal{E}_{pk}(m; r)$, and we say that (m, r) is a *witness to the decryption of* C *under* pk. For convenience, we assume $|pk| = k$, the security parameter. We let $\langle A(a), B(b)\rangle(c)$ be the random variable denoting the output of B following an execution of an interactive protocol between A (with private input a) and B (with private input b) on joint input c, where A and B have uniformly-distributed random tapes.

In a PPK protocol, sender $\mathcal{S}$ proves knowledge to receiver $\mathcal{R}$ of a witness to the decryption of some ciphertext C under the known public key. Both $\mathcal{S}$ and $\mathcal{R}$ have an additional joint input σ; in practice, this may be published along with the public key pk.[4] Our definitions build on the standard one for proofs of knowledge [3, 24], except that our protocols are technically *arguments* of knowledge and we therefore restrict ourselves to consideration of provers running in probabilistic, polynomial time.

To ensure that no information about m is revealed, a PPK is required to be "perfect zero-knowledge" in the following sense: Let the joint input σ be generated by some algorithm $\mathcal{G}(pk)$. We require the existence of a simulator $\mathcal{SIM}$ which takes pk as input and outputs parameters σ' whose distribution is equivalent to the output of $\mathcal{G}(pk)$. Then, given any ciphertext C (but no witness to its decryption), $\mathcal{SIM}$ should be able to perfectly simulate a PPK of C with any (malicious) receiver $\mathcal{R}'$ using parameters σ'.

[4] In all our applications, there is no incentive to cheat when generating σ.

Definition 1. *Let $\Pi = (\mathcal{G}, \mathcal{S}, \mathcal{R})$ be a tuple of* PPT *algorithms. Π is a* proof of plaintext knowledge (PPK) *for encryption scheme $(\mathcal{K}, \mathcal{E}, \mathcal{D})$ if the following conditions hold:*

(Completeness) *For all pk output by $\mathcal{K}(1^k)$, all σ output by $\mathcal{G}(pk)$, and all C with witness w to the decryption of C under pk we have $\langle \mathcal{S}(w), \mathcal{R}\rangle(pk, \sigma, C) = 1$. (When $\mathcal{R}$ outputs 1 we say it* accepts*.)*

(Perfect zero-knowledge) *There exists a* PPT simulator *$\mathcal{SIM}$ such that, for all pk output by $\mathcal{K}(1^k)$, all $\mathcal{R}'$, and all m, r, the following distributions are equivalent:*

$$\{\sigma \leftarrow \mathcal{G}(pk); C := \mathcal{E}_{pk}(m; r) : \langle \mathcal{S}(m, r), \mathcal{R}'\rangle(pk, \sigma, C)\}$$

$$\{(\sigma, s) \leftarrow \mathcal{SIM}_1(pk); C := \mathcal{E}_{pk}(m; r) : \langle \mathcal{SIM}_2(s), \mathcal{R}'\rangle(pk, \sigma, C)\}.$$

(Note that $\mathcal{SIM}_2$ does not need to rewind $\mathcal{R}'$.)

(Witness extraction) *There exists a function $\kappa : \{0,1\}^* \to [0,1]$, a negligible function $\varepsilon(\cdot)$, and an expected polynomial-time* knowledge extractor *$\mathcal{KE}$ such that, for all* PPT *algorithms $\mathcal{S}'$, with all but negligible probability over pk, σ, r, machine $\mathcal{KE}$ satisfies the following:*

> *Denote by $p_{pk,\sigma,r}$ the probability that $\mathcal{R}$ accepts when interacting with $\mathcal{S}'$ (using random tape r) on joint input pk, σ, C (where C is chosen by $\mathcal{S}'$). On input pk, σ, and access to $\mathcal{S}'_r$, the probability that $\mathcal{KE}$ outputs a witness to the decryption of C under pk is at least:*
>
> $$p_{pk,\sigma,r} - \kappa(pk) - \varepsilon(|pk|).$$

Our definition of non-malleability ensures that "anything proven by a man-in-the-middle adversary $\mathcal{M}$ is known by $\mathcal{M}$ (unless $\mathcal{M}$ simply copies a proof)." To formalize this, we allow $\mathcal{M}$ to interact with a simulator[5] (cf. Definition 1) while simultaneously interacting with an honest receiver $\mathcal{R}$. The goal of $\mathcal{M}$ is to successfully complete a PPK of C to $\mathcal{R}$ while the simulator is executing a PPK of C' to $\mathcal{M}$ (where C', C are chosen adaptively by $\mathcal{M}$). The following definition states (informally) that if $\mathcal{R}$ accepts $\mathcal{M}$'s proof — yet the transcripts of the two proofs are different — then a knowledge extractor $\mathcal{KE}^*$ can extract a witness to the decryption of C.

Definition 2. *PPK $(\mathcal{G}, \mathcal{S}, \mathcal{R})$ is* non-malleable *if there exists a simulator $\mathcal{SIM}$ (as in Definition 1), a function $\kappa^* : \{0,1\}^* \to [0,1]$, a negligible function $\varepsilon^*(\cdot)$, and an expected polynomial-time* knowledge extractor *$\mathcal{KE}^*$ such that, for all* PPT *algorithms $\mathcal{M}$, with all but negligible probability over choice of pk, σ, s, r, r', machine $\mathcal{KE}^*$ satisfies the following:*

> *Assume $\mathcal{M}$ (using random tape r') acts as a receiver with $\mathcal{SIM}_2(s; r)$ on joint input pk, σ, C' and simultaneously as a sender with $\mathcal{R}$ on joint*

[5] Note that, by the perfect zero-knowledge property, $\mathcal{M}$'s probability of convincing $\mathcal{R}$ remains unchanged whether $\mathcal{M}$ interacts with the simulator or a real sender.

input pk, σ, C (where C', C are chosen by $\mathcal{M}$). Let the transcripts of these two interactions be π' and π. Denote by p^* the probability (taken over random coins of $\mathcal{R}$) that $\mathcal{R}$ accepts in the above interaction and $\pi' \neq \pi$. On input s, r, pk, σ, and access to $\mathcal{M}_{r'}$, the probability that $\mathcal{KE}^*$ outputs a witness to the decryption of C under pk is at least:

$$p^* - \kappa^*(pk) - \varepsilon^*(|pk|).$$

Our definitions of zero-knowledge (in Definition 1) and non-malleability (in Definition 2) both consider the single-theorem case. The definitions may be modified for the multi-theorem case; however, the present definitions suffice for our intended applications.

2.1 A Note on Complexity Assumptions

Our complexity assumptions are with respect to adversaries permitted to run in *expected polynomial time*. For example, we assume that RSA inverses cannot be computed with more than negligible probability by any expected polynomial-time algorithm. The reason for this is our reliance on (efficient) constant-round proofs of knowledge, for which only expected polynomial-time knowledge extractors are known.

3 A Non-Malleable PPK for the RSA Cryptosystem

We begin with an overview of our technique. Recall the parameter σ which is used as a common input during execution of the PPK. To allow simulation, a first attempt is to have a PPK for ciphertext C consist of a witness-indistinguishable proof of knowledge of *either* a witness to the decryption of C *or* a witness x corresponding in some way to σ (using the known techniques for constructing such proofs [9]). Notice that a simulator who knows x can easily simulate a PPK for any ciphertext; on the other hand (informally), the protocol is sound since a PPT adversary does not know x.

This approach does not suffice to achieve non-malleability. To see why, consider a simulator interacting with man-in-the-middle $\mathcal{M}$ while $\mathcal{M}$ simultaneously interacts with receiver $\mathcal{R}$. Using the approach sketched above, the simulator gives a proof of "w or x" while $\mathcal{M}$ gives a proof of "w' or x", where w (resp. w') is a witness to the decryption of ciphertext C (resp. C'). The idea is now to rewind $\mathcal{M}$ and thus (hopefully) extract w'. But since the simulator must know x (since w is unknown; recall that C is chosen by $\mathcal{M}$), there is no contradiction is extracting x (and not w') from $\mathcal{M}$! Indeed, a more careful approach is needed.

To overcome this obstacle, we adapt a technique used previously in the context of non-malleable commitment [14]. The prover will choose a value α and prove knowledge of "w or x_α", where x_α corresponds to some function of σ and α. Thus, in the man-in-the-middle attack above, the simulator will be proving knowledge of "w or x_α" while $\mathcal{M}$ proves knowledge of "w' or $x_{\alpha'}$"; in particular, rewinding $\mathcal{M}$ will extract either the desired value w' or $x_{\alpha'}$. The desired witness will be extracted (and hence the PPK will be non-malleable) if the following conditions hold: (1) the simulator can know the witness x_α; yet (2) extracting $x_{\alpha'}$ for *any* $\alpha' \neq \alpha$ results in a contradiction; furthermore, (3) $\mathcal{M}$ cannot duplicate the value α used by the simulator, so that $\alpha' \neq \alpha$. Details follow in the remainder of this section.

The PPK we describe here will be for the following encryption scheme for ℓ-bit messages [4], which is semantically-secure under the RSA assumption: The modulus N is chosen as a product of two random $k/2$-bit primes, and e is a prime number such that $|e| = O(k)$.[6] The public key is (N, e). Let $hc(\cdot)$ be a hard-core bit [24] for the RSA permutation, and define $hc^*(r) \stackrel{\text{def}}{=} hc(r^{e^{\ell-1}}) \circ \cdots \circ hc(r^e) \circ hc(r)$. Encryption of ℓ-bit message m is done by choosing random $r \in \mathbb{Z}_N^*$, computing $C = r^{e^\ell} \bmod N$, and sending $\langle C, c \stackrel{\text{def}}{=} hc^*(r) \oplus m\rangle$. (Clearly, we could also extract more than a single hard-core bit per application of the RSA permutation.)

The PPK builds on the following Σ-protocol for proving knowledge of e^ℓ-th roots, based on [26]: To prove knowledge of $r = C^{1/e^\ell}$, the prover chooses a random element $r_1 \in \mathbb{Z}_N^*$ and sends $A = r_1^{e^\ell}$ to the verifier. The verifier replies with a random challenge $q \in \mathbb{Z}_e$. The prover responds with $R = r^q r_1$ and the receiver verifies that $R^{e^\ell} \stackrel{?}{=} C^q A$. To see that special soundness holds, consider two accepting conversations (A, q, R) and (A, q', R'). Since $R^{e^\ell} = C^q A$ and $(R')^{e^\ell} = C^{q'} A$ we have $(R/R')^{e^\ell} = C^{q-q'}$. Noting that $|q - q'|$ is relatively prime to e^ℓ, standard techniques may be used to compute the desired witness C^{1/e^ℓ} [26]. Special honest-verifier zero knowledge is demonstrated by the simulator which, on input C and a "target" challenge q, chooses random $R \in \mathbb{Z}_N^*$, computes $A = R^{e^\ell}/C^q$, and outputs the transcript (A, q, R).

We now describe the non-malleable PPK (cf. Figure 1). Parameters σ are generated by selecting two random elements $g, h \in \mathbb{Z}_N^*$ and a random function $H : \{0,1\}^* \to \mathbb{Z}_e$ from a family of universal one-way hash functions. Given σ, a PPK for ciphertext $\langle C, c\rangle$ proceeds as follows: first, a key-generation algorithm for a one-time signature scheme is run to yield verification key VK and signing key SK, and $\alpha = H(\mathsf{VK})$ is computed. The PPK will be a witness-indistinguishable proof of knowledge (using techniques of [9]) of *either* $r = C^{1/e^\ell}$ *or* $x_\alpha \stackrel{\text{def}}{=} (g^\alpha h)^{1/e}$. In more detail, the sender chooses random $r_1, R_2 \in \mathbb{Z}_N^*$ and $q_2 \in \mathbb{Z}_e$, and then computes $A_1 = r_1^{e^\ell}$ and $A_2 = R_2^e/(g^\alpha h)^{q_2}$. These values are sent (along with VK, C, c) as the first message. The receiver sends challenge $q \in \mathbb{Z}_e$. The sender responds with $q_1 = q - q_2 \bmod e$, and also $R_1 = r^{q_1} r_1$ (completing the "real" proof of knowledge with "challenge" q_1) and R_2 (completing the "simulated" proof of knowledge with "challenge" q_2). To complete the proof, the sender signs a transcript of the entire execution (including C, c) using SK and sends the signature to the receiver. The receiver verifies correctness of the

[6] For efficiency, the protocol may be modified for small e (e.g., $e = 3$); see [29].

Public key: N; prime e

$\sigma: \ g, h \in \mathbb{Z}_N^*; \ H: \{0,1\}^* \to \mathbb{Z}_e$

$\mathcal{S}$ (input $m \in \{0,1\}^\ell$)		$\mathcal{R}$
$(\mathsf{VK}, \mathsf{SK}) \leftarrow \mathsf{SigGen}(1^k)$		
$r, r_1, R_2 \leftarrow \mathbb{Z}_N^*; q_2 \leftarrow \mathbb{Z}_e$		
$C := r^{e^\ell}; c := hc^*(r) \oplus m$		
$\alpha := H(\mathsf{VK})$		
$A_1 := r_1^{e^\ell}$		
$A_2 := R_2^e / (g^\alpha h)^{q_2}$	$\xrightarrow{\mathsf{VK}, C, c, A_1, A_2}$	
	$\xleftarrow{\ q\ }$	$q \leftarrow \mathbb{Z}_e$
$q_1 := q - q_2 \bmod e$		
$R_1 := r^{q_1} r_1$		
$s \leftarrow \mathsf{Sign}_{\mathsf{SK}}(\text{transcript})$	$\xrightarrow{q_1, R_1, R_2, s}$	
		Verify: $R_1^{e^\ell} \stackrel{?}{=} C^{q_1} A_1$
		$R_2^e \stackrel{?}{=} \left(g^{H(\mathsf{VK})} h\right)^{(q - q_1 \bmod e)} A_2$
		$\mathsf{Vrfy}_{\mathsf{VK}}(\text{transcript}, s) \stackrel{?}{=} 1$

Fig. 1. Non-malleable PPK for the RSA cryptosystem.

proofs by checking that $R_1^{e^\ell} \stackrel{?}{=} C^{q_1} A_1$ and $R_2^e \stackrel{?}{=} (g^\alpha h)^{(q - q_1 \bmod e)} A_2$. Finally, the receiver verifies the correctness of the signature on the transcript.

Theorem 1. *Under the RSA assumption for expected poly-time algorithms, the protocol of Figure 1 is a non-malleable PPK (with $\kappa^*(pk) = 1/e$) for the RSA cryptosystem.*

The proof appears in Appendix B.

Concurrent composition. In many applications, it may be necessary to consider the case where multiple, concurrent proofs are conducted and witness extraction is required from each of them (and extraction is required as soon as the relevant proof is completed). If arbitrary interleaving of the proofs is allowed, a knowledge extractor operating in the obvious way may require exponential time; a similar problem is encountered in simulation of concurrent zero-knowledge proofs [17]. (In the context of concurrent zero-knowledge, the difficulty is to ensure that the concurrent interaction of a single prover with multiple verifiers is simulatable; for concurrent proofs of knowledge, the difficulty is to ensure that the concurrent interaction of a single verifier with multiple *provers* still allows *witness extraction*.) To ensure that our protocols remain proofs of knowledge in a concurrent setting, we introduce timing constraints [17]; essentially, these constraints prevent "bad" interleavings and allow extraction of all relevant witnesses. We describe this in more detail in the following section; further details and full proofs of security appear in the full version [29].

4 Applications

In this section, we briefly discuss applications of our non-malleable PPKs to the construction of protocols for (1) interactive IND-CCA2 encryption, (2) password-AKE in the public-key model, and (3) strong deniable authentication. The protocols described here may be based on any of the PPKs given in the full version of this work; security of these protocols may therefore be based on a variety of assumptions (e.g., RSA, hardness of factoring, or composite residuosity). All our applications are assumed to run in an asynchronous, concurrent environment; thus, in each case we augment our PPKs with timing constraints as discussed in the previous section. Further details and proofs of all theorems stated here appear in the full version [29].

Chosen-ciphertext-secure, interactive encryption. We give a definition of chosen-ciphertext security for interactive encryption in Appendix A. The non-malleable PPK of Figure 1 immediately yields an interactive encryption scheme: The receiver generates N, e, σ as above and sets the public key $pk = (N, e, \sigma)$ (the secret key is as usual for RSA). To encrypt message m under public key pk, the sender computes $C = r^{e^\ell}$ and $c = hc^*(r) \oplus m$, sends $\langle C, c\rangle$ to the receiver, and then executes the PPK using parameters σ. To decrypt, the receiver first determines whether to accept or reject the proof; if the receiver accepts, the receiver decrypts $\langle C, c\rangle$ as in the standard RSA scheme. If the proof is rejected, the receiver outputs $\perp$.

This scheme above is secure against adaptive chosen-ciphertext attacks when the adversary is given sequential access to the decryption oracle. To ensure security against an adversary given *concurrent* access to the decryption oracle, timing constraints are necessary. In particular, we require that the sender respond to the challenge within time α from when the challenge is sent. If the response is not received in time, the proof is rejected. Additionally, an acknowledgment is sent from the receiver to the sender upon completion of the protocol; this message is ack if the sender's proof was accepted and $\perp$ otherwise. The receiver delays sending the acknowledgment until time β has elapsed from when the second message of the protocol was sent (with $\beta > \alpha$).

Theorem 2. *Under the RSA assumption for expected poly-time algorithms, the protocol of Figure 1 (with $|e| = \Theta(k)$) is secure against sequential chosen-ciphertext attacks. If timing constraints are enforced, the protocol is secure against concurrent chosen-ciphertext attacks.*

Password-based authentication and key exchange. Protocols for password-based authentication may be constructed from any IND-CCA2 encryption scheme as follows [28, 5]: Let pw be the user's password. To authenticate the user, the server sends a nonce n and the user replies with an encryption of $pw \circ n$ (this brief description suppresses details which are unimportant here; see [28, 5]). When non-interactive encryption is used, the nonce is needed to prevent replay attacks. The server decrypts and verifies correctness of the password and the nonce. (Previous work [28, 5] proved security when non-interactive IND-CCA2 encryption was used, but the proofs extend for the case of interactive encryption.) If desired, a key K may be exchanged by encrypting $K \circ pw \circ n$.

Our interactive IND-CCA2 encryption schemes allow the first efficient implementations of these protocols based on assumptions other than DDH (e.g., we may base security on the RSA or factoring assumptions). Furthermore, since interaction is essential (to prevent replay attacks), using interactive encryption is not a serious drawback. Finally, if interactive encryption is used, the nonce is not needed if the probability that a server repeats its messages is negligible. Thus, password-AKE protocols constructed from our PPKs ultimately require only one more round than previous constructions.

Deniable authentication. Definitions of security for deniable authentication appear in Appendix A. Our non-malleable PPKs may be adapted to give deniable authentication protocols whose security is based on the *one-wayness* (rather than semantic security) of an underlying encryption scheme. This results in protocols with improved efficiency and with security based on potentially weaker assumptions (recall that previous efficient constructions achieving strong deniability require the DDH assumption).

We present a generic paradigm for constructing a deniable authentication protocol based on our non-malleable PPKs. The basic idea is for $\mathcal{V}$ to encrypt a random value, send the resulting ciphertext to the prover, and then execute a non-malleable PPK for the ciphertext (here, $\mathcal{V}$ acts as the prover in the PPK). To "bind" the protocol to a message m to be authenticated, m is included in the transcript and signed along with everything else. Assuming $\mathcal{V}$'s proof succeeds, $\mathcal{P}$ authenticates the message by decrypting the ciphertext and sending back the resulting value.

Figure 2 shows an example of this approach applied to the PPK of Figure 1. The public key of the prover $\mathcal{P}$ is an RSA modulus N, a prime e (with $|e| = \Theta(k)$), elements $g, h \in \mathbb{Z}_N^*$, and a hash function H chosen randomly from a family of universal one-way hash functions. Additionally, $\mathcal{P}$ has secret key d such that $de = 1 \bmod \varphi(N)$. To have m authenticated by $\mathcal{P}$, the verifier chooses a random $y \in \mathbb{Z}_N^*$, computes $C = y^e$, and then performs a non-malleable PPK for C. Additionally, the message m is sent as part of the first message of the protocol, and is signed along withe the rest of the transcript. If the verifier's proof succeeds, the prover computes C^d (i.e., y) and sends this value to the verifier. Otherwise, the prover simply replies with $\perp$.

As before, timing constraints are needed when concurrent access to the prover is allowed. In this case, we require that the verifier respond to the challenge within time α from when the challenge is sent (the proof is rejected otherwise).

Public key: N; prime e; $g, h \in \mathbb{Z}_N^*$; $H : \{0,1\}^* \to \mathbb{Z}_e$

$\mathcal{V}$ (input $m \in \{0,1\}^*$)		$\mathcal{P}$ (input d)
$(\mathsf{VK}, \mathsf{SK}) \leftarrow \mathsf{SigGen}(1^k)$		
$y, r_1, R_2 \leftarrow \mathbb{Z}_N^*; q_2 \leftarrow \mathbb{Z}_e$		
$C := y^e; \alpha := H(\mathsf{VK})$		
$A_1 := r_1^e$		
$A_2 := R_2^e / (g^\alpha h)^{q_2}$	$\xrightarrow{\mathsf{VK}, m, C, A_1, A_2}$	
	$\xleftarrow{q}$	$q \leftarrow \mathbb{Z}_e$
$q_1 := q - q_2 \bmod e$		
$R_1 := y^{q_1} r_1$		
$s \leftarrow \mathsf{Sign}_{\mathsf{SK}}(\text{transcript})$	$\xrightarrow{q_1, R_1, R_2, s}$	Verify: $R_1^e \stackrel{?}{=} C^{q_1} A_1$
		$R_2^e \stackrel{?}{=} \left(g^{H(\mathsf{VK})} h\right)^{(q - q_1 \bmod e)} A_2$
	$\xleftarrow{C^d}$	$\mathsf{Vrfy}_{\mathsf{VK}}(\text{transcript}, s) \stackrel{?}{=} 1$

Fig. 2. A deniable authentication protocol based on RSA.

Additionally, the final message of the protocol is not sent by the prover until at least time β has elapsed since sending the challenge (with $\beta > \alpha$).

Theorem 3. *Under the RSA assumption for expected poly-time algorithms, the protocol of Figure 2 (with $|e| = \Theta(k)$) is a strong deniable authentication protocol for adversaries given sequential access to the prover. If timing constraints are enforced, the protocol is a strong ε-deniable authentication protocol for adversaries given concurrent access to the prover.*

We stress that the resulting deniable authentication protocols are quite practical. For example, the full version of this work shows a deniable authentication protocol based on the CDH assumption which has the same round-complexity, requires fewer exponentiations, has a shorter public key, and is based on a weaker assumption than the most efficient previously-known protocol for strong deniable authentication. Furthermore, no previous efficient protocols were known based on the RSA, factoring, or composite residuosity assumptions. We also remark that interactive IND-CCA2 encryption schemes do *not*, in general, yield deniable authentication protocols using the paradigm sketched above. For example, when using the IND-CCA2 interactive encryption scheme of [15], two additional rounds are necessary just to achieve *weak* deniable authentication (the terms "strong" and "weak" are explained in Appendix A).

Public key: N; prime e; $g, h \in \mathbb{Z}_N^*$

$\mathcal{V}$ (input $m \in M_k \subset \mathbb{Z}_e$)		$\mathcal{P}$ (input d)
$y, r_1, R_2 \leftarrow \mathbb{Z}_N^*; q_2 \leftarrow \mathbb{Z}_e$		
$C := y^e; A_1 := r_1^e$		
$A_2 := R_2^e / (g^m h)^{q_2}$	$\xrightarrow{m, C, A_1, A_2}$	
	$\xleftarrow{q}$	$q \leftarrow \mathbb{Z}_e$
$q_1 := q - q_2 \bmod e$		
$R_1 := y^{q_1} r_1$	$\xrightarrow{q_1, R_1, R_2}$	Verify: $R_1^e \stackrel{?}{=} C^{q_1} A_1$
	$\xleftarrow{C^d}$	$R_2^e \stackrel{?}{=} (g^m h)^{(q - q_1 \bmod e)} A_2$

Fig. 3. A deniable authentication protocol with improved efficiency.

Further efficiency improvements (see Figure 3) result from the observation that, in the context of deniable authentication, it is not necessary to generate and send a signature key VK and sign the transcript (recall this is done to force the adversary to use $\alpha' = H(\mathsf{VK}') \neq H(\mathsf{VK}) = \alpha$; see Section 3). Instead, we may simply use m (i.e., the message to be authenticated) as our α: informally, this works since the adversary must choose $m' \neq m$ in order to falsely authenticate a *new* message m which was never authenticated by the prover. On the other hand, because the adversary chooses m (and this value must be guessed by the simulator in advance), the scheme is only provably secure for polynomial-size message spaces M_k. We refer to [29] for further details.

Acknowledgments. I am grateful to Moti Yung and Rafail Ostrovsky for their many helpful comments and suggestions regarding the work described here.

References

1. Y. Aumann and M.O. Rabin. A Proof of Plaintext Knowledge Protocol and Applications. Manuscript. June, 2001.
2. B. Barak. Constant-Round Coin Tossing with a Man in the Middle or Realizing the Shared Random String Model. *Proceedings of the 43rd Annual Symposium on Foundations of Computer Science*, IEEE, 2002, pp. 345–355.
3. M. Bellare and O. Goldreich. On Defining Proofs of Knowledge. *Advances in Cryptology — Crypto '92*, LNCS vol. 740, E. Brickell, ed., Springer-Verlag, 1992, pp. 390–420.
4. M. Blum and S. Goldwasser. An Efficient Probabilistic Public-Key Encryption Scheme which Hides All Partial Information. *Advances in Cryptology — Crypto '84*, LNCS vol. 196, G. Blakley and D. Chaum, eds., Springer-Verlag, pp. 289–302.

5. M. Boyarsky. Public-Key Cryptography and Password Protocols: the Multi-User Case. *ACM Conference on Computer and Communications Security*, 1999, pp. 63–72.
6. R. Canetti. Universally Composable Security: A New Paradigm for Cryptographic Protocols. *Proceedings of the 42nd Annual Symposium on Foundations of Computer Science*, IEEE, 2001, pp. 136–145.
7. R. Cramer. Modular Design of Secure Yet Practical Cryptographic Protocols. PhD Thesis, CWI and U. Amsterdam, 1996.
8. R. Cramer, I. Damgård, and J.B. Nielsen. Multiparty Computation from Threshold Homomorphic Encryption. *Advances in Cryptology — Eurocrypt 2001*, LNCS vol. 2045, B. Pfitzmann, ed., Springer-Verlag, 2001, pp. 280–299.
9. R. Cramer, I. Damgård, and B. Schoenmakers. Proofs of Partial Knowledge and Simplified Design of Witness Hiding Protocols. *Advances in Cryptology — Crypto '94*, LNCS vol. 839, Y. Desmedt, ed., Springer-Verlag, 1994, pp. 174–187.
10. R. Cramer and V. Shoup. A Practical Public-Key Cryptosystem Provably Secure against Adaptive Chosen Ciphertext Attack. *Advances in Cryptology — Crypto '98*, LNCS vol. 1462, H. Krawczyk, ed., Springer-Verlag, 1998, pp. 13–25.
11. R. Cramer and V. Shoup. Universal Hash Proofs and a Paradigm for Adaptive Chosen-Ciphertext-Secure Public-Key Encryption. *Advances in Cryptology — Eurocrypt 2002*, LNCS vol. 2332, L. Knudsen, ed., Springer-Verlag, 2002, pp. 45–64.
12. A. De Santis and G. Persiano. Zero-Knowledge Proofs of Knowledge Without Interaction. *Proceedings of the 43rd Annual Symposium on Foundations of Computer Science*, IEEE, 1992, pp. 427–436.
13. A. De Santis, G. Di Crescenzo, R. Ostrovsky, G. Persiano, and A. Sahai. Robust Non-Interactive Zero Knowledge. *Advances in Cryptology — Crypto 2001*, LNCS vol. 2139, J. Kilian, ed., Springer-Verlag, 2001, pp. 566–598.
14. G. Di Crescenzo, J. Katz, R. Ostrovsky, and A. Smith. Efficient and Non-Interactive Non-Malleable Commitment. *Advances in Cryptology — Eurocrypt 2001*, LNCS vol. 2045, B. Pfitzmann, ed., Springer-Verlag, 2001, pp. 40–59
15. D. Dolev, C. Dwork, and M. Naor. Non-Malleable Cryptography. *SIAM J. Computing* 30(2): 391–437 (2000).
16. C. Dwork and M. Naor. Zaps and Their Applications. *Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, IEEE, 2000, pp. 283–293.
17. C. Dwork, M. Naor, and A. Sahai. Concurrent Zero-Knowledge. *Proceedings of the 30th Annual Symposium on Theory of Computing*, ACM, 1998, pp. 409–418.
18. C. Dwork and A. Sahai. Concurrent Zero-Knowledge: Reducing the Need for Timing Constraints. *Advances in Cryptology — Crypto '98*, LNCS vol. 1462, H. Krawczyk, ed., Springer-Verlag, 1998, pp. 442–457.
19. T. El Gamal. A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. *IEEE Transactions on Information Theory* 31(4): 469–472 (1985).
20. U. Feige, A. Fiat, and A. Shamir. Zero-Knowledge Proofs of Identity. *Journal of Cryptology* 1(2): 77–94 (1988).
21. U. Feige, D. Lapidot, and A. Shamir. Multiple Non-Interactive Zero-Knowledge Proofs Based on a Single Random String. *Proceedings of the 31st Annual Symposium on Foundations of Computer Science*, IEEE, 1990, pp. 308–317.
22. Z. Galil, S. Haber, and M. Yung. Symmetric Public-Key Encryption. *Advances in Cryptology — Crypto '85*, LNCS vol. 218, H.C. Williams, ed., Springer-Verlag, 1985, pp. 128–137.

23. J. Garay and P. MacKenzie. Concurrent Oblivious Transfer. *Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, IEEE, 2000, pp. 314–324.
24. O. Goldreich. *Foundations of Cryptography: Basic Tools*. Cambridge University Press, Cambridge, UK, 2001.
25. O. Goldreich, S. Micali, and A. Wigderson. Proofs that Yield Nothing but Their Validity or: All Languages in NP Have Zero-Knowledge Proof Systems. *JACM* 38(3): 691–729 (1991).
26. L.C. Guillou and J.-J. Quisquater. A Practical Zero-Knowledge Protocol Fitted to Security Microprocessors Minimizing Both Transmission and Memory. *Advances in Cryptology — Eurocrypt '88*, LNCS vol. 330, C.G. Günther, ed., Springer-Verlag, 1988, pp. 123–128.
27. S. Haber. Multi-Party Cryptographic Computations: Techniques and Applications. PhD Thesis, Columbia University, 1987.
28. S. Halevi and H. Krawczyk. Public-Key Cryptography and Password Protocols. *ACM Transactions on Information and System Security* 2(3): 230–268 (1999).
29. J. Katz. Efficient Cryptographic Protocols Preventing "Man-in-the-Middle" Attacks. PhD Thesis, Columbia University, 2002.
30. M. Naor. Deniable Ring Authentication. *Advances in Cryptology — Crypto 2002*, LNCS vol. 2442, M. Yung, ed., Springer-Verlag, 2002, pp. 481–498.
31. M. Naor and M. Yung. Public-Key Cryptosystems Provably Secure Against Chosen-Ciphertext Attack. *Proceedings of the 22th Annual Symposium on Theory of Computing*, ACM, 1990, pp. 427–437.
32. H. Ong and C.P. Schnorr. Fast Signature Generation With a Fiat-Shamir-Like Scheme. *Advances in Cryptology — Eurocrypt '90*, LNCS vol. 473, I. Damgård, ed., Springer-Verlag, 1990, pp. 432–440.
33. P. Paillier. Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. *Advances in Cryptology — Eurocrypt '99*, LNCS vol. 1592, J. Stern, ed., Springer-Verlag, 1999, pp. 223–238.
34. M. Rabin. Digitalized Signatures and Public-Key Functions as Intractable as Factorization. Technical Report MIT/LCS/TR-212, MIT Laboratory for Computer Science, January 1979.
35. C. Rackoff and D. Simon. Non-Interactive Zero-Knowledge Proof of Knowledge and Chosen-Ciphertext Attack. *Advances in Cryptology — Crypto '91*, LNCS vol. 576, J. Feigenbaum, ed., Springer-Verlag, 1991, pp. 433–444.
36. R. Rivest, A. Shamir, and L.M. Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Communications of the ACM* 21(2): 120–126 (1978).
37. A. Sahai. Non-Malleable Non-Interactive Zero-Knowledge and Adaptive Chosen-Ciphertext Security. *Proceedings of the 40th Annual Symposium on Foundations of Computer Science*, IEEE, 1999, pp. 543–553.
38. C.P. Schnorr. Efficient Identification and Signatures for Smart Cards. *Advances in Cryptology — Crypto '89*, LNCS vol. 435, G. Brassard, ed., Springer-Verlag, 1989, pp. 239–252.

A Additional Definitions

Interactive encryption. A number of approaches for defining chosen-ciphertext security in the interactive setting are possible; we sketch one such definition here.

We have a sender, a receiver, and an adversary $\mathcal{M}$ who controls all communication between them. To model this, we define an *encryption oracle* $\mathcal{E}_{b,pk}$ (playing the role of the sender) and a *decryption oracle* $\mathcal{D}_{sk}$ (player the role of the receiver) to which $\mathcal{M}$ is given access. The adversary may interact with $\mathcal{E}_{b,pk}$ multiple times, and may arbitrarily interleave requests to this oracle with requests to the decryption oracle. At the outset of the experiment, a bit b is chosen at random. An instance of the adversary's interaction with the encryption oracle proceeds as follows: first, the adversary sends two messages m_0, m_1 to $\mathcal{E}_{b,pk}$. The oracle then executes the encryption protocol for message m_b; the adversary, however, need not act as an honest receiver. The oracle maintains state between the adversary's oracle calls, and the adversary may have multiple concurrent interactions with the oracle. When $\mathcal{E}_{b,pk}$ sends the final message for a given instance of its execution, we say that instance is *completed*.

$\mathcal{D}_{sk}$ also maintains state between oracle calls, and the adversary may again have multiple concurrent interactions with this oracle. Furthermore, the adversary need not act as an honest sender. Each time a given decryption-instance is completed, the decryption oracle decrypts and returns the result (i.e., a message or $\perp$) to the adversary.

The adversary succeeds if it can guess b. Clearly, some limitations must be placed on the adversary's access to $\mathcal{D}_{sk}$ or else the adversary may simply forward messages between $\mathcal{E}_{b,pk}$ and $\mathcal{D}_{sk}$ and thereby trivially determine b. At any point during the adversary's execution, the set of transcripts of completed encryption-instances of $\mathcal{E}_{b,pk}$ is well defined. Upon completing a decryption-instance, let $S = \{\pi_1, \ldots, \pi_\ell\}$ denote the transcripts of all completed encryption-instances. We allow the adversary to receive the decryption corresponding to a decryption-instance with transcript π' only if $\pi' \notin S$.

Definition 3. *Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an interactive, public-key encryption scheme. Π is* CCA2-secure *if, for any* PPT *adversary A, the following is negligible:*

$$\left|\Pr\left[(sk, pk) \leftarrow \mathcal{K}(1^k); b \leftarrow \{0,1\} : A^{\mathcal{E}_{b,pk}, \mathcal{D}_{sk}}(1^k, pk) = b\right] - 1/2\right|,$$

where A's access to $\mathcal{D}_{sk}$ is restricted as discussed above.

Deniable authentication. We review the definitions of [17, 16]. We have a prover $\mathcal{P}$ who is willing to authenticate messages to a verifier $\mathcal{V}$; however, $\mathcal{P}$ is *not* willing to allow the verifier to convince a third party (after the fact) that $\mathcal{P}$ authenticated anything. This is formalized by ensuring that any transcript of an execution of the protocol can be simulated by a verifier alone (without any access to $\mathcal{P}$). Furthermore, an adversary $\mathcal{M}$ (acting as man-in-the-middle between $\mathcal{P}$ and a verifier) should not be able to authenticate a message m to the verifier which $\mathcal{P}$ does not authenticate for $\mathcal{M}$. More formally, a *strong* deniable authentication protocol satisfies the following (in addition to a standard completeness requirement):

– SOUNDNESS. Assume $\mathcal{P}$ concurrently authenticates messages $m_1, m_2, \ldots$ chosen adaptively by a PPT adversary $\mathcal{M}$. Then $\mathcal{M}$ will succeed with at most negligible probability in authenticating a message $m \notin \{m_1, \ldots\}$ to an honest verifier.

– Strong deniability. Assume $\mathcal{P}$ concurrently authenticates polynomially-many messages chosen adaptively by a PPT adversary $\mathcal{V}'$. Then there exists an expected-polynomial-time simulator that, given black-box access to $\mathcal{V}'$, can output a transcript indistinguishable from a transcript of a real execution between $\mathcal{V}'$ and $\mathcal{P}$.

A relaxation of the above definition [17] allows the simulator to have access to $\mathcal{P}$ when producing the simulated transcript, but $\mathcal{P}$ authenticates some fixed sequence of messages independent of those chosen by $\mathcal{V}'$. We call this *weak* deniable authentication. In practice, weak deniability may not be acceptable because the protocol then leaves an undeniable record that $\mathcal{P}$ authenticated *something* (even if not revealing *what*). However, $\mathcal{P}$ may want to deny that any such interaction ever took place.

The notion of *ε-deniability* [17] requires that for any given $\varepsilon > 0$, there exists a simulator whose expected running time is polynomial in k and $1/\varepsilon$ and which outputs a simulated transcript such that the advantage of any poly-time algorithm in distinguishing real transcripts from simulated transcripts is negligibly close to ε.

B Proof of Theorem 1

A sketch of the proof is given here; full details can be found in [29]. It is easy to show that the protocol is a PPK. To prove non-malleability, consider the following simulator: $\mathcal{SIM}_1(N, e)$ chooses random hash function H, runs the key-generation algorithm for the one-time signature scheme to generate $(\mathsf{VK}', \mathsf{SK}')$, and computes $\alpha' = H(\mathsf{VK}')$. Random elements $g, x \in \mathbb{Z}_N^*$ are chosen, and h is set equal to $g^{-\alpha'} x^e$. Finally, $\sigma \stackrel{\text{def}}{=} \langle g, h, H \rangle$ is output along with state information $\mathsf{state} = \langle \mathsf{VK}', \mathsf{SK}, x \rangle$. Note that σ output by $\mathcal{SIM}_1$ has the correct distribution. Furthermore, given state, $\mathcal{SIM}_2$ can simulate the proof of Figure 1 for any ciphertext: simply use verification key VK' and then the witness $x = \left(g^{\alpha'} h\right)^{1/e}$ is known. The resulting simulation is perfect and is achieved without rewinding the (potentially) dishonest verifier.

Fix pk, σ, state, and randomness r for $\mathcal{SIM}_2$. We are given adversary $\mathcal{M}$ using (unknown) random tape r' who interacts with both $\mathcal{SIM}_2(\mathsf{state}; r)$ and honest receiver $\mathcal{R}$. Once the challenge q of $\mathcal{R}$ is fixed, the entire interaction is completely determined (recall that ciphertext $\langle C', c' \rangle$ for which $\mathcal{SIM}_2$ will be required to prove a witness, is chosen adaptively by $\mathcal{M}$). Define $\pi'(q)$ as the transcript of the conversation between $\mathcal{SIM}_2(\mathsf{state}; r)$ and $\mathcal{M}_{r'}$ when q is the challenge sent by $\mathcal{R}$; analogously, define $\pi(q)$ as the transcript of the conversation between $\mathcal{M}_{r'}$ and $\mathcal{R}$ when q is the challenge of $\mathcal{R}$.

The knowledge extractor $\mathcal{KE}^*$ is given $pk, \sigma, \mathsf{state}, r$, and access to $\mathcal{M}_{r'}$. When we say that $\mathcal{KE}^*$ *runs* $\mathcal{M}_{r'}$ *with challenge* q we mean that $\mathcal{KE}^*$ interacts with $\mathcal{M}_{r'}$ by running algorithm $\mathcal{SIM}_2(\mathsf{state}; r)$ and sending challenge q for $\mathcal{R}$. We stress that interleaving of messages (i.e., scheduling of messages to/from $\mathcal{R}$ and $\mathcal{SIM}_2$) is completely determined by $\mathcal{M}_{r'}$.

$\mathcal{KE}^*$ first picks a random value $q^1 \in \mathbb{Z}_e$ and runs $\mathcal{M}_{r'}$ with challenge q^1. If $\pi(q^1)$ is not accepting, or if $\pi(q^1) = \pi'(q^1)$, stop and output $\perp$. Otherwise, let VK denote the verification key used by $\mathcal{M}$, and let $\alpha = H(\mathsf{VK})$. Using standard rewinding techniques (see [29]), $\mathcal{KE}^*$ extracts either (1) a witness to the decryption of the ciphertext $\langle C, c\rangle$ sent by $\mathcal{M}$ or (2) the value $y \stackrel{\text{def}}{=} (g^\alpha h)^{1/e}$ (or possibly both). Further, $\mathcal{KE}^*$ runs in expected polynomial time.

To complete the proof, we need to argue that extraction of y occurs with negligible probability (and thus $\mathcal{KE}^*$ extracts a witness to the decryption of $\langle C, c\rangle$ with all but negligible probability). We first note that, w.h.p., $H(\mathsf{VK}) \neq H(\mathsf{VK}')$; this follows from the security of the one-time signature scheme (so $\mathsf{VK} \neq \mathsf{VK}'$) and the universal one-way hash function. But then $\Delta \stackrel{\text{def}}{=} \alpha - \alpha' \neq 0$ and

$$y \stackrel{\text{def}}{=} (g^\alpha h)^{1/e} = \left(g^\Delta x^e\right)^{1/e} = \left(g^\Delta\right)^{1/e} x,$$

and therefore $\tilde{y} \stackrel{\text{def}}{=} y/x$ satisfies $\tilde{y}^e = g^\Delta$. Note that $|\Delta|$ and e are relatively prime since $\Delta \in (-e, e)$. Standard techniques allow efficient computation of $g^{1/e}$.

The above shows that extraction of y enables $\mathcal{KE}^*$ to compute $g^{1/e}$ and hence invert a given RSA instance $\langle N, e, g\rangle$ (note that $\mathcal{KE}^*$ needs no secret information about N, e, or g to run). So, under the RSA assumption for expected polynomial-time algorithms, extraction of y occurs with negligible probability.