Construction of a Non-Malleable Encryption Scheme from Any Semantically Secure One

Rafael Pass[1], abhi shelat[2], and Vinod Vaikuntanathan[3]

[1] Cornell University
[2] IBM ZRL
[3] MIT

Abstract. There are several candidate semantically secure encryption schemes, yet in many applications *non-malleability* of encryptions is crucial. We show how to transform *any* semantically secure encryption scheme into one that is non-malleable for arbitrarily many messages.

Keywords. Public-key Encryption, Semantic Security, Non-malleability, Non-interactive Zero-knowledge Proofs.

1 Introduction

The most basic goal of an encryption scheme is to guarantee the privacy of data. In the case of public-key encryption, the most universally accepted formalization of such privacy is the notion of *semantic security* as defined by Goldwasser and Micali [GM84]. Intuitively, semantic security guarantees that whatever a polynomial-time machine can learn about a message given its encryption, it can learn even without the encryption.

Non-malleability, as defined by Dolev, Dwork and Naor [DDN00], is a stronger notion of security for encryption schemes. In addition to the privacy guarantee, non-malleability of an encryption scheme guarantees that it is infeasible to *modify* ciphertexts $\alpha_1, \ldots, \alpha_n$ into one, or many, other ciphertexts of messages related to the decryption of $\alpha_1, \ldots, \alpha_n$. At first one might wonder whether it is possible to violate non-malleability, i.e., modify ciphertexts into ciphertexts of related messages, without violating semantic security, i.e. without learning anything about the original message.

It turns out, however, that many semantically secure encryption schemes, including the original one proposed in [GM84], are easily malleable. Thus, non-malleability is a strictly stronger requirement than semantic security. Moreover, non-malleability is often times indispensable in practical applications. For example, no one would consider secure an electronic "sealed-bid" auction in which an adversary can consistently bid exactly one more dollar than the previous bidders. The importance of non-malleability raises an important question:

> Is it possible to *immunize* any semantically secure encryption scheme against malleability attacks?

The work of Dolev, Dwork and Naor partially answers this question affirmatively. They show how to perform such an immunization (for the even stronger chosen-ciphertext attack) *assuming the existence of enhanced trapdoor permutations*. Subsequently, several other constructions of non-malleable encryption schemes have been presented under various number-theoretic assumptions such as decisional Diffie-Hellman [CS98] and quadratic-residuosity [CS02]. Nevertheless, there exist some notable computational assumptions, such as computational Diffie-Hellman, and the worst-case hardness of various lattice-problems [AD97, Reg05], under which semantically secure encryption schemes exist, yet no non-malleable encryption schemes are known.

Another partial answer comes from a *single-message* non-malleable encryption scheme based on any semantically secure scheme called DDNLite [Nao04, Dwo99]. By single-message, we mean that the DDNLite scheme is secure only when the adversary receives *one* ciphertext.[4] As was stated in [DDN00], many-message security is a *sine qua non* of non-malleable encryption, and indeed the classic motivating examples for non-malleability (such as auctions) require it.

In this paper, our main result is to fully address the posed question. We show how to immunize *any* semantically secure encryption scheme into one that is non-malleable for any number of messages *without any further computational assumptions.*

Main Theorem 1 (Informal) *Assume the existence of an encryption scheme that is semantically secure against chosen-plaintext attacks (CPA). Then there exists an encryption scheme that is many-message non-malleable against CPA.*

As an additional contribution, we show that even in the case of single-message security, the previous definitions of non-malleability and also the DDN-Lite scheme allow a subtle class of malleability attacks. We address this issue by presenting a stronger definition of non-malleability and emphasize that our construction meets this stronger notion.

1.1 Definitional Contributions

Our main conceptual contribution is to *strengthen* the definition of non-malleable encryption. Our definition has the advantage of being both technically simple and providing a *natural* explanation of non-malleability. In one sentence, our definition technically captures the following requirement:

> "No matter what encryptions the adversary receives, the *decryption* of his output will be indistinguishable."

Recall the indistinguishability-based definition of secrecy: an adversary cannot *distinguish* between the ciphertexts of any two messages of his choosing. Another way of phrasing this definition is to say that no matter which encryption an adversary receives, his *output* will be indistinguishable.

[4] As pointed out by Genaro and Lindell [GL03], the DDNLite scheme is in fact *malleable* when the adversary receives multiple ciphertexts.

By requiring the even the *decryption* of his output be indistinguishable, we capture the property that also the plaintexts corresponding to the ciphertexts output by the adversary are (computationally) *independent* of the messages he receives encryptions of.

Notice how in the context of auctions this requirement directly implies that an adversary's bid is independent of the previous one. In general, it naturally captures that an adversary cannot *maul* a given encryption into another one that is related—otherwise, the decryption of his output could be distinguished. Except for some simplifications, our definition is very similar to the formal description of IND-PA0 from [BS99]. However, there is some ambiguity regarding whether or not the IND-PA0 definition allows the adversary to succeed by outputting an invalid ciphertext.[5] As we discuss in the next section, this issue of invalid ciphertext can become a serious one.

The Problem of Invalid Ciphertexts Our definition highlights a subtle technical weakness in previous definitions of non-malleability relating to how one treats an adversary who produces *invalid* ciphertexts as part of its output.

In the original definition from [DDN00], and in subsequent equivalent definitions [BDPR98, BS99], an adversary who produces any invalid ciphertexts as part of its output is considered to have lost the non-malleability game.

THE DEVIL IS IN THE DETAILS While seemingly harmless, this small detail unleashes a subtle but devastating class of attacks against encryption schemes which only meet the [DDN00] definition.

For example, consider a "quorum voting application" in which participants encrypt their vote, either YES or NO, under the pollmaster's public key. The pollmaster decrypts all votes, counts those which decrypt to either YES or NO as *valid* votes, ignores all others (e.g., discards the "hanging chads"), and announces whether (a) there was as a *quorum* of valid votes, and if so (b) whether the "yeahs" have a majority of the valid votes. (This voting process roughly resembles how voting works in shareholder meetings of a corporation or condo association meetings.)

Now suppose a "swing voter" Alice submits an encryption of her vote x, and Bob is able to produce a ciphertext for the vote NO if $x =$NO and $\perp$ otherwise (in Appendix A, we show exactly such an attack against the DDNLite cryptosystem). In a tight election, such an attack enhances Bob's ability to cause the measure to fail since his vote is either NO when the swing vote is NO, or his vote undermines the legitimacy of the election by preventing the establishment of the necessary quorum. This attack is better than simply voting NO, because a NO vote by itself might be the one which establishes quorum and allows the measure to pass.

[5] Specifically, the formal IND-PA0 definition states one thing but the motivating discussion states another. It also appears that the main result in [BS99] is sensitive to this issue. We further discuss these issues in a forthcoming note.

Under the original definition, one can verify that Bob does not formally win the non-malleability game, and thus an encryption scheme which allows such an attack would still be considered non-malleable under CPA.[6]

MANY MESSAGES More interestingly, we also show that this small detail is the *key factor* which allows one to prove composability of the non-malleable encryption scheme, i.e., that a single-message secure scheme will also be non-malleable when the adversary receives many messages. Indeed, Gennaro and Lindell [GL03, p.17] point out that one-message non-malleability *does not* imply many-message non-malleability under the [DDN00] definition. As mentioned above, this feature is very important in practice, since an adversary may receive *several* encryptions and we still want to guarantee non-malleability.

To understand the problem, many-message non-malleability is usually proven through a reduction of the following form: an adversary that can break the many-message non-malleability can be used to break the single-message non-malleability. Such a proof requires a *hybrid argument* in which one feeds the adversary one of two specially-constructed distribution of ciphertexts, on which it is possible to distinguish the adversary's output. However, since the input to the adversary is *hybrid*, the adversary may produce invalid ciphertexts—and such may be the basis upon which the adversary's output is distinguishable. If, in the single-message definition, the experiment unconditionally outputs 0 if the adversary produces *any* invalid ciphertexts, then we do not even have the *chance* to use the distinguisher which breaks the many-message case to break the single-message one.

MANY KEYS There is one final issue which arises in practice. The adversary may receive encryptions under different public keys. Here too, we define a notion of non-malleability. Although this scenario is also usually handled by a standard hybrid argument, there are a few definitional issues to consider.

1.2 Overview of our Construction

The *immunization* paradigm introduced by Dolev, Dwork and Naor is elegant and conceptually simple. To encrypt a message, one (a) generates several encryptions of the same message under independent public keys, (b) gives a non-interactive zero-knowledge proof that *all* resulting ciphertexts are encryptions of the same message, and (c) signs the entire bundle with a one-time signature.

Because one-time signatures can be constructed from one-way functions, only step (b) in the DDN construction requires an extra computational assumption—namely the existence of trapdoor permutations. Indeed, non-interactive zero-knowledge proofs have been well-studied, and it remains a pressing open ques-

[6] A similar attack applies to the classic auction example presented in [DDN00]. Here, if Bob knows Alice's bid is in the range $[x, y]$ and Bob can submit $y - x$ bids, the same attack allows Bob to generate a set of bids such that exactly one bid matches Alice's bid and all the rest decrypt to invalid ciphertexts.

tion whether they can be constructed without using trapdoor permutations or assumptions of comparable magnitude.[7]

In the context of non-malleable encryption, however, our main observation is that standard non-interactive zero-knowledge proofs are not necessary. Instead, we use *designated verifier* proofs in which only a verifier with some special secret information related to the common random string can verify the proof. (In contrast, standard non-interactive proofs can be verified by any third party with access to the proof and the common random string.) For example, the non-malleable encryption constructions of Cramer and Shoup [CS98, CS02] can be interpreted as using some form of designated verifier proofs based on specific number-theoretic assumptions. This was further generalized by Elkind and Sahai [ES02] through the notion of *simulation-sound* designated verifier NIZK proofs. Our main technical result is to show that

† *Plain* designated verifier NIZK proofs are sufficient to obtain non-malleable encryption schemes, and

†† *Plain* designated-verifier NIZK proofs can be constructed from *any* semantically secure encryption scheme.

The first of these results follows by an argument essentially similar to the one employed in [DDN00].

At a high-level, our approach to constructing a *plain* designated verifier NIZK proof is to crush a Σ-protocol for $\mathcal{NP}$ (such as Blum's Hamiltonicity protocol [Blu86]) into a non-interactive proof with only the help of a semantically-secure encryption scheme.

Towards CCA2 Security The main results in this paper relate to non-malleability under a chosen plaintext attack. In the stronger chosen ciphertext attack (of which there are two varieties, CCA1 and CCA2), the adversary also has access to a decryption oracle at various stages of the attack. A natural question is whether our same techniques can be used to immunize against a malleability attack in these stronger CCA attack models.

Unfortunately, when designated-verifier NIZK proofs are sequentially composed, it is possible for the verifier's secret information to *leak*. In particular, a malicious prover who can interactively submit proofs and learn whether they are accepted or not might be able to *learn* about the verifier's secret information. Because a CCA1 or CCA2 attack provides precisely such an opportunity, the soundness of the designated verifier proof system will no longer hold. Thus, our technique does not provide a method to achieve full CCA security.

Organization Our paper is organized into three sections. In §2, we present our new definition of non-malleable encryption. In §3, we define and construct designated verifier NIZK proof systems from any semantically-secure encryption scheme. Finally, in §4, we show that the DDN scheme when implemented with

[7] The work of [PS05] begins to address this question by presenting certain models of NIZK proofs that are either unconditional or only require one-way functions.

designated verifier NIZK proofs is non-malleable with respect to our stronger definition from §2.

2 Definitions

Preliminaries. If A is a probabilistic polynomial time (p.p.t) algorithm that runs on input x, $A(x)$ denotes the random variable corresponding to the output of A on input x and uniformly random coins. Sometimes, we want to make the randomness used by A explicit, in which case, we let $A(x;r)$ denote the output of A on input x and random coins r. We denote computational indistinguishability [GM84] of ensembles A and B by $A \stackrel{c}{\approx} B$.

2.1 Semantically Secure Encryption

Definition 1 (Encryption Scheme). *A triple* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *is an encryption scheme, if* Gen *and* Enc *are p.p.t. algorithms and* Dec *is a deterministic polynomial-time algorithm which satisfies the following property:*
Perfect Correctness. *There exists a polynomial* $p(k)$ *and a negligible function* $\mu(k)$ *such that for* every message m, *and* every random tape r_e,

$$\Pr[r_g \stackrel{R}{\leftarrow} \{0,1\}^{p(k)}\ ;\ (\textsc{pk}, \textsc{sk}) \leftarrow \mathsf{Gen}(1^k; r_g);\ \mathsf{Dec}_{\textsc{sk}}(\mathsf{Enc}_{\textsc{pk}}(m; r_e)) \neq m] \leq \mu(k).$$

Definition 2 (Indistinguishability of Encryptions). *Let* $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *be an encryption scheme and let the random variable* $\mathsf{IND}_b(\Pi, A, k)$, *where* $b \in \{0,1\}$, A *is a p.p.t. algorithm and* $k \in \mathbb{N}$, *denote the result of the following probabilistic experiment:*

$\mathsf{IND}_b(\Pi, A, k)$:
  $(\textsc{pk}, \textsc{sk}) \leftarrow \mathsf{Gen}(1^k)$
  $(m_0, m_1, \textsc{state}_A) \leftarrow A(\textsc{pk})$ *s.t.* $|m_0| = |m_1|$
  $y \leftarrow \mathsf{Enc}_{\textsc{pk}}(m_b)$
  $D \leftarrow A_2(y, \textsc{state}_A)$
  Output D

$(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *is indistinguishable under a chosen-plaintext attack if* $\forall$ *p.p.t. algorithms* A *the following two ensembles are computationally indistinguishable:*

$$\left\{\mathsf{IND}_0(\Pi, A, k)\right\}_{k\in\mathbb{N}} \stackrel{c}{\approx} \left\{\mathsf{IND}_1(\Pi, A, k)\right\}_{k\in\mathbb{N}}$$

2.2 Our Definition of Non-malleable Encryption

Definition 3 (Non-Malleable Encryption). *Let* $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *be an encryption scheme and let the random variable* $\mathsf{NME}_b(\Pi, A, k, \ell)$ *where* $b \in \{0,1\}$, $A = (A_1, A_2)$ *and* $k, \ell \in \mathbb{N}$ *denote the result of the following probabilistic experiment:*

$\mathsf{NME}_b(\mathit{\Pi}, A, k, \ell):$

$(\text{PK}, \text{SK}) \leftarrow \mathsf{Gen}(1^k)$

$(m_0, m_1, \text{STATE}_A) \leftarrow A_1(\text{PK})$ *s.t.* $|m_0| = |m_1|$

$y \leftarrow \mathsf{Enc}_{\text{PK}}(m_b)$

$(c_1, \ldots, c_\ell) \leftarrow A_2(y, \text{STATE}_A)$

Output $(d_1, \ldots, d_\ell)$ *where* $d_i = \begin{cases} \perp & \text{if } c_i = y \\ \mathsf{Dec}_{\text{SK}}(c_i) & \text{otherwise} \end{cases}$

$(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *is non-malleable under a chosen-plaintext attack if* $\forall$ *p.p.t. algorithms* $A = (A_1, A_2)$ *and for any polynomial* $p(k)$*, the following two ensembles are computationally indistinguishable:*

$$\left\{\mathsf{NME}_0(\mathit{\Pi}, A, k, p(k))\right\}_{k \in \mathbb{N}} \overset{c}{\approx} \left\{\mathsf{NME}_1(\mathit{\Pi}, A, k, p(k))\right\}_{k \in \mathbb{N}}$$

Let us remark on the natural similarity of the above definition with the definition of indistinguishable security of an encryption scheme. Indeed, the first few lines of the experiment are exactly the same. In the last step, we add the requirement that the *decryptions* (modulo copying) of the output of A_2 are indistinguishable in the two experiments. This captures the requirement that even the decryption of the adversary's output must be *computationally independent* of the values (even when if they are encrypted) received as inputs.

This definition both highlights the essence of non-malleability for encryption schemes and, similar to the original notion of indistinguishable security, provides the most technically convenient formalization to use in larger proofs.

Syntactic Differences with the IND-PA0 definition As we discussed in the Introduction, there is ambiguity as to how the IND-PA0 [BS99] definition treats invalid ciphertexts. Aside from this semantic difference, one can see that our definition is a simplification of IND-PA0. We simply eliminate the guess stage of the experiment.

2.3 Many Message Non-malleability

Notice that Definition 3 only applies when the adversary A_2 receives one encryption as an input. In practice, an adversary may receive *several* encryptions under many different public keys, and we would still like to guarantee non-malleability.

Definition 4 (Many Message Non-Malleability). *Let* $\mathit{\Pi} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *be an encryption scheme and let the random variable* $\mathsf{mNM}_b(\mathit{\Pi}, A, k, \ell)$ *where* $b \in \{0, 1\}$*,* $A = (A_1, A_2)$ *and* $k, \ell, J \in \mathbb{N}$ *denote the result of the following probabilistic experiment :*

$\mathsf{mNM}_b(\Pi, A, k, \ell, J)$:

For $i = 0, \ldots, J$, $(\text{PK}_i, \text{SK}_i) \leftarrow \mathsf{Gen}(1^k)$

$((m_1^0, m_1^1, t_1), \ldots, (m_\ell^0, m_\ell^1, t_\ell), \text{STATE}) \leftarrow A_1(\text{PK}_1, \ldots, \text{PK}_J)$ *s.t.* $|m_i^0| = |m_i^1|$

For $j = 0, \ldots, \ell \quad y_j \leftarrow \mathsf{Enc}_{\text{PK}_{t_j}}(m_j^b)$

$((c_1, s_1), \ldots, (c_\ell, s_\ell)) \leftarrow A_2(y_1, \ldots, y_\ell, \text{STATE})$

Output $(d_1, \ldots, d_\ell)$ *where* $d_i = \begin{cases} \perp & \text{if } s_i \notin [1, J] \\ \perp & \text{if } c_i = y_j \text{ and } s_i = t_j,\ j \in [1, J] \\ \mathsf{Dec}_{\text{SK}_{s_i}}(c_i) & \text{otherwise} \end{cases}$

We say that $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *is non-malleable under a chosen-plaintext attack if for all p.p.t. algorithms* $A = (A_1, A_2)$ *and for all polynomials* $\ell(\cdot), J(\cdot)$, *the following two ensembles are computationally indistinguishable:*

$$\left\{\mathsf{mNM}_0(\Pi, A, k, \ell(k), J(k))\right\}_{k \in \mathbb{N}} \overset{c}{\approx} \left\{\mathsf{mNM}_1(\Pi, A, k, \ell(k), J(k))\right\}_{k \in \mathbb{N}}$$

Theorem 1. *An encryption scheme is non-malleable iff it is many message non-malleable.*

Sketch: To prove the forward implication, we use an adversary (A_1, A_2) and a distinguisher D which breaks the many-message non-malleability of Π with advantage η to to break the non-malleability of Π with advantage η/ℓ^2.

Define a new experiment $\mathsf{mNM}_{(b_1,\ldots,b_\ell)}(\Pi, A, k, \ell, J)$ indexed by an ℓ-bit string $(b_1, \ldots, b_\ell)$ which is the same as $\mathsf{mNM}_0(\Pi, A, k, \ell, J)$ except in the fifth line (change is underlined): $y_j \leftarrow \mathsf{Enc}_{\text{PK}_{t_j}}(\underline{m_j^{b_j}})$. Define $B(i) = (\overbrace{0, \ldots, 0}^{i}, \overbrace{1, \ldots, 1}^{\ell - i})$ and note that $\mathsf{mNM}_0 = \mathsf{mNM}_{B(0)}$ and $\mathsf{mNM}_1 = \mathsf{mNM}_{B(\ell)}$. Because D distinguishes mNM_0 from mNM_1, there exists some $g^* \in [1, \ell]$ such that D distinguishes $\mathsf{mNM}_{B(g^*)}$ from $\mathsf{mNM}_{B(g^*+1)}$ with advantage η/ℓ. This suggests the following adversary: $A_1'(\text{PK})$ guesses value $g \in [1, \ell]$, generates $J - 1$ public keys $(\text{PK}_i, \text{SK}_i)$, and feeds $(\text{PK}_1, \ldots, \text{PK}_{g-1}, \text{PK}, \text{PK}_{g+1}, \ldots, \text{PK}_J)$ to A_1 to get a vector of triples $((m_1^0, m_1^1, t_1), \ldots, (m_\ell^0, m_\ell^1, t_\ell))$. Finally, A' outputs (m_g^0, m_g^1) as its challenge pair and outputs state information with g, all public keys, etc.

Adversary $A_2'(y, \text{STATE}')$, on input an encryption y, simulates the replaced line 5 of experiment $\mathsf{mNM}_{B(g)}$ with the exception that it uses y for the gth encryption: $y_g \leftarrow y$. It then feeds vector $\boldsymbol{y}$ to A_2 to produce $((c_1, s_1), \ldots, (c_\ell, s_\ell))$. A_2' must produce ciphertexts only for PK, and can do so by simply decrypting and re-encrypting[8] appropriately. The rest of the argument concludes by conditioning on the event that $g = g^*$. $\square$

One can see here the importance of removing the invalid ciphertext restriction for the hybrid argument to work. Because the reduction feeds a hybrid distribution to A_2, A_2 may produce invalid ciphertexts, and these $\perp$ values may form

[8] There is one technicality concerning how A_2' can form invalid ciphertexts; we defer it to the full version.

the basis for distinguishability in the NME experiments. If one simply forces the single-message experiment to return 0 when invalid ciphertexts are produced, then the value of both experiments ($b = 0, 1$) will be 0 and the weaker definition will thus be met *even though* there might still be a distinguisher which could have distinguished the output of A'_2.

3 Designated Verifier NIZK

In this section, we define and construct a designated verifier NIZK proof system. The overall approach to our construction is to *crush* a 3-round Σ protocol into a one-round proof by having the prover *encrypt* all possible third-round responses to the verifier's challenge. Because we use a Σ-protocol in which the verifier's challenge is a single bit, this approach is feasible and results in short proofs. The notable benefit of this approach is that our only complexity assumption is the existence of a semantically-secure encryption scheme.

3.1 Defining Designated Verifier NIZK Proof Systems

In the designated verifier model, a non-interactive proof system has an associated polynomial-time sampleable distribution $\mathcal{D}$ over binary strings of the form (PP, SP). During a setup phase, a trusted party samples from $\mathcal{D}$, publishes PP and privately hands the Verifier SP. The Prover and Verifier then use their respective values during the proof phase.

This definition is very similar to the definition of NIZK proofs in the secret parameter model (as in [PS05]). The difference between the secret parameter model and the designated verifier model is that in the former case, the prover might be given some *secret information*, whereas this is strictly not the case in the latter. Also note that in the standard notion of NIZK proofs, the common random (resp. reference) string model can be derived as special cases of this definition by setting SP to the empty string.

Definition 5 (Designated Verifier NIZK Proof System). *A triple of algorithms, $(\mathcal{D}, P, V)$, is called a* designated verifier non-interactive zero-knowledge proof system *for an $\mathcal{NP}$-language L with witness relation R_L, if the algorithms $\mathcal{D}$ and P are probabilistic polynomial-time, the algorithm V is deterministic polynomial-time and there exists a negligible function μ such that the following three conditions hold:*

- COMPLETENESS: *For every $(x, w) \in R_L$*

$$\Pr\Big[(\text{PP}, \text{SP}) \leftarrow \mathcal{D}(1^{|x|});\ \pi \leftarrow P(\text{PP}, x, w)\ :\ V(\text{PP}, \text{SP}, x, \pi) = 1\Big] \geq 1 - \mu(|x|)$$

- SOUNDNESS: *For every prover algorithm B*

$$\Pr\left[\begin{array}{l} (\text{PP}, \text{SP}) \leftarrow \mathcal{D}(1^{|x|})\ ;\ (x', \pi') \leftarrow B(\text{PP})\ :\ x' \notin L \quad \textit{and} \\ \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad V(\text{PP}, \text{SP}, x', \pi') = 1 \end{array}\right] \leq \mu(|x|)$$

– ADAPTIVE ZERO-KNOWLEDGE: *For every p.p.t. theorem chooser A, there exists a p.p.t. simulator $S = (S_1, S_2)$ such that the outputs of the following experiments are indistinguishable.*

$\mathsf{ZK}_A(k)$	$\mathsf{ZK}_A^S(k)$
$(\text{PP}, \text{SP}) \leftarrow \mathcal{D}(1^k)$	$(\text{PP}', \text{SP}', \text{STATE}) \leftarrow S_1(1^k)$
$(x, w, \text{STATE}_A) \leftarrow A(\text{PP})$	$(x, w, \text{STATE}_A) \leftarrow A(\text{PP}')$
$\pi \leftarrow P(\text{PP}, x, w)$	$\pi' \leftarrow S_2(\text{PP}', \text{SP}', x, \text{STATE})$
If $(x, w) \notin R_L$*, output* $\perp$	*If* $(x, w) \notin R_L$*, output* $\perp$
Else output $(\text{PP}, \text{SP}, x, \pi, \text{STATE}_A)$	*Else output* $(\text{PP}', \text{SP}', x, \pi', \text{STATE}_A)$

Note that the Verifier V is a deterministic machine. This extra restriction is only used to simplify the exposition of our constructions.

3.2 Constructions

Before giving a high-level view of our protocol, let us briefly recall the structure of a 3-round honest-verifier zero-knowledge proof of knowledge for NP, also referred to as a Σ-protocol [CDS94].

Σ-protocols A Σ-protocol consists of four algorithms P_1, V_1, P_2, V_2. On input, a statement x and a witness w, the first algorithm $P_1(x, w)$ produces a pair (a, s) which represent a "commitment" and related state information. The Prover then sends a to the Verifier. The verifier runs $V_1(x, a)$ to produce a random challenge b and sends b to the Prover. The prover runs $P_2(s, b)$ to produce a response c_b and sends it to the Verifier. Finally, the verifier runs $V_2(x, a, b, c_b)$ to determine whether to accept the proof. An example of such a protocol is Blum's Hamiltonicity protocol. The properties of Σ-protocols, namely completeness, special soundness, and special honest-verifier zero-knowledge, are presented in [CDS94].

Construction summary. In our construction, the prover receives k pairs of public encryption keys as the public parameter. The verifier receives the same k pairs, and also receives a secret parameter consisting of the secret key for *exactly* one key in each pair. A proof consists of k triples. To generate the ith triple of a proof, the prover runs the Σ-protocol prover algorithm on (x, w) using both 0 and 1 as the verifier's challenge to produce a triple $(a_i, c_{0,i}, c_{1,i})$. The prover encrypts this triple as $(a_i, \mathsf{Enc}_{\text{PK}_{0,i}}(c_{0,i}), \mathsf{Enc}_{\text{PK}_{1,i}}(c_{1,i}))$. (We note that Camenisch and Damgård use a similar idea in [CD00] to construct an interactive verifiable encryption scheme. The roots of this idea begin to appear much earlier in [KMO89].)

To verify the proof, V considers each triple $(a_i, \alpha_{0,i}, \alpha_{1,i})$ and decrypts either the second or third component using the secret key given in SP. He then runs the Σ-protocol verifier on $(a_i, f_i, \mathsf{Dec}(\alpha_{f_i,i}))$, and accepts if all k triples are accepting proofs.

There is one additional detail to consider. The Σ-protocol for Blum Hamiltonicity requires a commitment scheme. Normally, a one-round commitment scheme requires trapdoor permutations. To get around this problem, we use a two-round commitment scheme (such as Naor [Nao91]) which can be constructed from one-way functions, which are in turn implied by the semantically-secure encryption scheme. The first round message of the commitment scheme is placed in the public parameter.

Theorem 2. *Assume there exists a semantically secure encryption scheme. Then there exists a designated verifier NIZK proof system for any language $L \in \mathcal{NP}$.*

Sampling Algorithm $\mathcal{D}(1^k)$**.** For $i = 1, \ldots, k$ and $b = 0, 1$, run $\mathsf{Gen}(1^k)$ $2k$ times with independent random coins, to get k key-pairs $(\text{PK}_i^b, \text{SK}_i^b)$. For $i = 1, \ldots, k$, flip coin $f_i \overset{R}{\leftarrow} \{0, 1\}$. Generate the receiver message σ for a two-round commitment scheme.

Let $\text{PP}_{dv} \overset{\text{def}}{=} [(\text{PK}_i^0, \text{PK}_i^1, \sigma)]_{i=1}^k$ and $\text{SP}_{dv} \overset{\text{def}}{=} [f_i, \text{SK}_i^{f_i}]_{i=1}^k$. Output $(\text{PP}_{dv}, \text{SP}_{dv})$.

Prover $P(\text{PP}_{dv}, x, w)$**.** For $i = 0, \ldots, k$, generate triples as follows:

$$
\begin{aligned}
&(a_i, s_i) \leftarrow P_1(x, w) \\
&c_{b,i} \leftarrow P_2(s, b) \text{ for both } b = 0, 1 \\
&\alpha_{b,i} \leftarrow \mathsf{Enc}_{\text{PK}_{b,i}}(c_{b,i}) \text{ for } b = 0, 1.
\end{aligned}
$$

and output $\pi \overset{\text{def}}{=} [(a_i, \alpha_{0,i}, \alpha_{1,i})]_{i=1}^k$.

Verifier $V(\text{PP}_{dv}, \text{SP}_{dv}, x, \pi)$**.** Parse π into k triples of the form $(a_i, \alpha_{0,i}, \alpha_{1,i})$. For $i = 1, \ldots, k$, compute $m_i \overset{\text{def}}{=} \mathsf{Dec}_{\text{SK}_i^{f_i}}(\alpha_{f_i,i})$ and run the verifier $V_2(a_i, f_i, m_i)$. If all k proofs are accepted, output ACCEPT, else output REJECT.

Fig. 1. DESIGNATED VERIFIER NIZK PROTOCOL

The proof that the scheme in Fig. 1 is a designated verifier NIZK proof system is standard and omitted for lack of space. □

4 Constructing a Non-malleable Encryption Scheme

In this section, we construct an encryption scheme that is non-malleable under CPA attacks. Our construction is exactly the DDN construction [DDN00] in which the standard NIZK proof is replaced with a designated-verifier NIZK proof. By the results from the previous section, our construction only relies on the assumption of the existence of a semantically secure encryption scheme.

Theorem 3 (Main Theorem, restated). *Assume there exists an encryption scheme that is semantically-secure under a CPA attack. Then, there exists an encryption scheme that is non-malleable for many messages under a CPA attack.*

Proof. (of Theorem 3) Let $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ be any semantically secure encryption scheme. Let $(\mathsf{GenSig}, \mathsf{Sign}, \mathsf{Ver})$ be any existentially unforgeable *strong* one-time signature scheme.[9] Without loss of generality, assume that GenSig produces verification keys of length k.[10] Define the $\mathcal{NP}$-language L as follows:

$$\big[(c_1, \ldots, c_k), (p_1, \ldots, p_k)\big] \in L \text{ if and only if}$$
$$\exists\big[m, (r_1, \ldots, r_n)\big] \text{ such that } c_i = \mathsf{Enc}_{p_i}(m; r_i) \text{ for } i = 1, \ldots, n.$$

In words, the language L contains pairs consisting of a k-tuple of ciphertexts and a k-tuple of public keys such that the ciphertexts are encryptions of the *same message* m under the k public keys.

Let $(\mathcal{D}, P, V)$ be a designated verifier NIZK proof system for L. We show that the encryption scheme $\Pi = (\mathsf{NMGen}, \mathsf{NMEnc}, \mathsf{NMDec})$ defined in Figure 4 is a non-malleable encryption scheme. The proof has two parts.

Just as in [DDN00], we define an encryption scheme $E' = (\mathsf{Gen}', \mathsf{Enc}', \mathsf{Dec}')$ in which one simply encrypts a message k times with k independently chosen public keys, and we show that E' is a semantically secure encryption scheme under the assumption that $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is one. This is done in Lemma 1.

Then in Lemma 2, we show that Π is a non-malleable encryption scheme if E' is a semantically secure encryption scheme. The proof is concluded by noting that both designated verifier NIZK proofs and strong one-time signatures can be constructed given any semantically secure encryption scheme (The former is true by virtue of Theorem 2. The latter follows by combining the observation that encryption implies one-way functions, Rompel's result showing that one-way functions imply universal one-way hash functions [Rom90], and the result that universal one-way hash functions imply strong one-time signature schemes [Gol04, Lam79]). □

The definition of the encryption scheme $E' = (\mathsf{Gen}', \mathsf{Enc}', \mathsf{Dec}')$ below and the proof of Lemma 1 are exactly as in DDN, reproduced below for the sake of completeness.

- $\mathsf{Gen}'(1^k)$: For $i = 1, \ldots, k$, run $(\mathrm{PK}_i, \mathrm{SK}_i) \leftarrow \mathsf{Gen}(1^k)$ with independent random coins. Set $\mathrm{PK} \stackrel{\text{def}}{=} (\mathrm{PK}_1, \ldots, \mathrm{PK}_k)$ and $\mathrm{SK} \stackrel{\text{def}}{=} (\mathrm{SK}_1, \ldots, \mathrm{SK}_k)$.
- $\mathsf{Enc}'_{\mathrm{PK}}(m)$: Output $[\mathsf{Enc}_{\mathrm{PK}_1}(m; r_1), \ldots, \mathsf{Enc}_{\mathrm{PK}_k}(m; r_k)]$.
- $\mathsf{Dec}'_{\mathrm{SK}}([c_1, c_2, \ldots, c_k])$: Compute $m'_i = \mathsf{Dec}_{\mathrm{SK}_i}(c_i)$. If all the m'_i are not equal, output $\perp$, else output m'_1.

Lemma 1. *If* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *is semantically secure, then* $(\mathsf{Gen}', \mathsf{Enc}', \mathsf{Dec}')$ *is semantically secure.*

Proof. Via a standard hybrid argument. Omitted for space. □

[9] A strong signature is one in which, given a signature σ of a message m, it is infeasible to produce a message m' and a valid signature σ' of m', such that $(\sigma, m) \neq (\sigma', m')$. i.e, it is infeasible also to produce a different signature for the *same message*.

[10] This is without loss of generality since we can set k to be an upperbound on the length of verification keys that GenSig produces.

$\mathsf{NMGen}(1^k)$:

1. For $i \in [1,k], b \in \{0,1\}$, run $\mathsf{Gen}(1^k)$ to generate key-pairs $(\mathrm{PK}_i^b, \mathrm{SK}_i^b)$.
2. Run $\mathcal{D}(1^k)$ to generate $(\mathrm{PP}, \mathrm{SP})$.

Set $\mathrm{PK} \stackrel{\text{def}}{=} \left\{ (\langle \mathrm{PK}_i^0, \mathrm{PK}_i^1 \rangle)_{i=1}^k, \mathrm{PP} \right\}$ and $\mathrm{SK} \stackrel{\text{def}}{=} \left\{ (\langle \mathrm{SK}_i^0, \mathrm{SK}_i^1 \rangle)_{i=1}^k, \mathrm{SP} \right\}$.

$\mathsf{NMEnc}_{\mathrm{PK}}(m)$:

1. Run the signature algorithm $\mathsf{GenSig}(1^k)$ to generate $(\mathrm{SKSIG}, \mathrm{VKSIG})$. Let $(v_1, \ldots, v_k)$ be the binary representation of VKSIG.
2. Compute the ciphertexts $c_i \leftarrow \mathsf{Enc}_{\mathrm{PK}_i^{v_i}}(m)$. Let $\boldsymbol{c} \stackrel{\text{def}}{=} (c_1, c_2, \ldots, c_k)$.
3. Run the designated verifier NIZK Prover to generate a proof π that $[(c_1, \ldots, c_k), (\mathrm{PK}_1^{v_1}, \ldots, \mathrm{PK}_k^{v_k})] \in L$.
4. Compute the signature $\sigma \leftarrow \mathsf{Sign}_{\mathrm{SKSIG}}(\langle \boldsymbol{c}, \pi \rangle)$.

Output the tuple $[\boldsymbol{c}, \pi, \mathrm{VKSIG}, \sigma]$.

$\mathsf{NMDec}_{\mathrm{SK}}(c)$:

1. Verify the signature with $\mathsf{Ver}_{\mathrm{VKSIG}}[\langle \boldsymbol{c}, \pi \rangle, \sigma]$; output $\perp$ upon failure.
2. Verify the proof with $V(\mathrm{PP}, \mathrm{SP}, (\boldsymbol{c}, \mathrm{PK}), \pi)$; output $\perp$ upon failure.
3. Let $\mathrm{VKSIG} = (v_1, \ldots, v_k)$. Compute $m_1 = \mathsf{Dec}_{\mathrm{SK}_1^{v_i}}(c_1)$ and output the result.

Fig. 2. The Non-malleable Encryption Scheme Π

Lemma 2. *If $E' = (\mathsf{Gen}', \mathsf{Enc}', \mathsf{Dec}')$ is a semantically secure encryption scheme, then Π is a non-malleable encryption scheme.*

Proof. To prove that Π is a non-malleable encryption scheme, we need to show that for any p.p.t. adversary A and for all polynomials $p(k)$,

$$\left\{ \mathsf{NME}_0(\Pi, A, k, p(k)) \right\}_{k \in \mathbb{N}} \stackrel{c}{\approx} \left\{ \mathsf{NME}_1(\Pi, A, k, p(k)) \right\}_{k \in \mathbb{N}}$$

We show this by a hybrid argument. Consider the following experiments:

Experiment $\mathsf{NME}_b^{(1)}(\Pi, A, k, p(k))$ – *Using a Simulated NIZK Proof:* proceeds exactly like NME_b except that the simulator for the designated verifier NIZK proof system is used to generate the public parameters and to compute the challenge ciphertext (as opposed to generating an honest proof by running the prover algorithm P). Let $S = (S_1, S_2)$ denote the simulator guaranteed by the adaptive zero-knowledge of $(\mathcal{D}, P, V)$. More formally, $\mathsf{NME}_b^{(1)}$ proceeds exactly like NME_b except for the following differences:

1. The encryption key $(\mathrm{PK}, \mathrm{SK})$ is generated by (1) honestly running the key-generation algorithm Gen to generate the $2k$ encryption keys $(\mathrm{PK}_i^b, \mathrm{SK}_i^b)$, *but* (2) running the simulator $S_1(1^k)$ to generate the key-pair $(\mathrm{PP}, \mathrm{SP})$ for the designated verifier NIZK (instead of running $\mathcal{D}(1^k)$ as in NMGen).
2. Generate k encryptions of m_b (just as in Steps 1 and 2 of NMEnc). *But*, instead of using the designated verifier prover, generate a "simulated proof"

by running S_2. (Note that S_2 does not use the witness—namely, m_b and the randomness used for encryption—in order to generate the simulated proof).

Experiment $\mathsf{NME}_b^{(2)}(\Pi, A, k, p(k))$ – *Semantic Security of* E'*:* proceeds exactly like $\mathsf{NME}_b^{(1)}$ except for the following differences:

1. Run Gen' to get two sets of public keys $PK = \{\text{PK}_i\}_{i=1}^k$ and $PK' = \{\text{PK}'_i\}_{i=1}^k$, along with the corresponding secret-keys $SK = \{\text{SK}_i\}_{i=1}^k$ and $SK' = \{\text{SK}'_i\}_{i=1}^k$. Generate a verification key and signing key for the signature scheme $(\text{VKSIG}^*, \text{SKSIG}^*)$. Construct a public-key for Π as follows: Let v_i be the i^{th} bit of VKSIG^*. Set $\text{PK}_i^{v_i} = \text{PK}_i$, $\text{SK}_i^{v_i} = \perp$, $\text{PK}_i^{1-v_i} = \text{PK}'_i$ and $\text{SK}_i^{1-v_i} = \text{SK}'_i$. ($\mathsf{NME}_b^{(2)}$ will use the secret-keys corresponding to each PK'_i, but not PK_i, later in the experiment).
2. After receiving the tuple $(\psi_1, \dots, \psi_\ell)$ of ciphertexts from A_2, decrypt each $\psi_j = \left[\boldsymbol{c}_j, \pi_j, \text{VKSIG}_j, \sigma_j\right]$ as follows: If the signature σ_j in ψ_j does not verify, output $\perp$. If $\text{VKSIG}_j = \text{VKSIG}^*$, output $\perp$. If the NIZK proof π_j fails verification, output $\perp$. Else, decrypt one of the components of ψ_j, for which the secret-key is known (such a component is guaranteed to exist, since $\text{VKSIG}_j \neq \text{VKSIG}^*$) and output the result.

We now show that these experiments are indistinguishable. The following claim follows from the adaptive zero-knowledge property of the NIZK system.

Claim 1 $\left\{\mathsf{NME}_b(\Pi, A, k, p(k))\right\}_{k\in\mathbb{N}} \stackrel{c}{\approx} \left\{\mathsf{NME}_b^{(1)}(\Pi, A, k, p(k))\right\}_{k\in\mathbb{N}}$

Proof. Assume, for contradiction, that there exists a p.p.t. algorithm D which distinguishes $\mathsf{NME}_b(\Pi, A, k, p(k))$ from $\mathsf{NME}_b^{(1)}(\Pi, A, k, p(k))$. Then, we construct a theorem-chooser A_{zk} and a ZK distinguisher D_{zk} that violate the adaptive zero-knowledge of the proof system $(\mathcal{D}, P, V)$ for the language L. That is, D_{zk} distinguishes between the experiments $\mathsf{ZK}_{A_{\mathsf{zk}}}$ and $\mathsf{ZK}_{A_{\mathsf{zk}}}^S$, where S is the zero-knowledge simulator.

On input PP, the theorem-chooser A_{zk} works as follows:

1. Run $\mathsf{Gen}(1^k)$ $2k$ times, to generate $2k$ key-pairs $(\text{PK}_i^b, \text{SK}_i^b)_{i\in[k], b\in\{0,1\}}$. Run the adversary A_1 on input $\left[(\text{PK}_i^b)_{i\in[k], b\in\{0,1\}}, \text{PP}\right]$. A_1 returns a pair of plaintexts m_0 and m_1 and a string STATE.
2. Produce the challenge ciphertext $\boldsymbol{c}$ as follows:
   - Generate a key-pair $(\text{SKSIG}, \text{VKSIG})$ for the signature scheme.
   - Pick a random $b \in \{0, 1\}$, and for $1 \le i \le k$, let $c_i \leftarrow \mathsf{Enc}_{\text{PK}_i^{v_i}}(m_b; r_i)$, where r_i is the randomness used for encryption.
     Let $\boldsymbol{c}$ denote $(c_1, c_2, \dots, c_k)$ and PK denote $(\text{PK}_1^{v_1}, \dots, \text{PK}_k^{v_k})$, and $\boldsymbol{r}$ denote $(r_1, r_2, \dots, r_k)$.
3. Let $\mathbf{x} = (\boldsymbol{c}, \text{PK})$ and $\mathbf{w} = (m_b, \boldsymbol{r})$. Output the theorem-witness pair $(\mathbf{x}, \mathbf{w})$. Also output the contents of the work-tape as STATE_A.

The ZK distinguisher D_{zk}, on input (PP, SP), the theorem $(\boldsymbol{c}, \text{PK})$, the proof π and the state STATE_A, does the following:

1. Run A_2 on input the ciphertext $\left[\boldsymbol{c}, \pi, \text{VKSIG}, \mathsf{Sign}_{\text{SKSIG}}(\langle \boldsymbol{c}, \pi\rangle)\right]$ to produce a sequence of ciphertexts $(\psi_1, \psi_2, \ldots, \psi_{p(k)})$. Run the decryption algorithm $\mathsf{Dec}_{\text{SK}}(\psi_i)$ on each of these ciphertexts to get plaintexts $(\mu_1, \mu_2, \ldots, \mu_{p(k)})$.
2. Run distinguisher D on the sequence of plaintexts $(\mu_1, \mu_2, \ldots, \mu_{p(k)})$ and output whatever D outputs.

The experiment $\mathsf{ZK}_{A_{\mathsf{zk}}}$ (that is, when D_{zk} is given as input the real proof), perfectly simulates the experiment $\mathsf{NME}_b(\Pi, A, k, p(k))$, whereas the experiment $\mathsf{ZK}^S_{A_{\mathsf{zk}}}$ (that is, when D_{zk} is run with a simulated proof) perfectly simulates $\mathsf{NME}^{(1)}_b(\Pi, A, k, p(k))$. If the outputs of D in the experiments are different, then D_{zk} distinguishes between a real proof and a simulated proof, contradicting the adaptive zero-knowledge of the NIZK proof system $(\mathcal{D}, P, V)$. □

Next, we show that experiments $\mathsf{NME}^{(1)}_0(\cdots)$ and $\mathsf{NME}^{(2)}_0(\cdots)$ are statistically indistinguishable. To this end, we define three events, $\mathtt{badNIZK}(\mathsf{Expt})$, $\mathtt{badSig}(\mathsf{Expt})$ and $\mathtt{badKey}(\mathsf{Expt})$, corresponding to the experiment Expt. We show that the experiments $\mathsf{NME}^{(1)}_b$ and $\mathsf{NME}^{(2)}_b$ are *identical*, under the assumption that the events $\mathtt{badNIZK}$, $\mathtt{badSig}$ and $\mathtt{badKey}$ *never* happen in these experiments. Then, we show that the bad events happen with negligible probability in both the experiments. Taken together, these two statements let us conclude that $\mathsf{NME}^{(1)}_b$ and $\mathsf{NME}^{(2)}_b$ are statistically indistinguishable. Details follow.

Claim 2 $\left\{\mathsf{NME}^{(1)}_0(\Pi, A, k, p(k))\right\}_{k\in\mathbb{N}} \overset{s}{\approx} \left\{\mathsf{NME}^{(2)}_0(\Pi, A, k, p(k))\right\}_{k\in\mathbb{N}}$

Proof. Define the event $\mathtt{badNIZK}(\mathsf{Expt})$, to capture the event that the adversary A violates the soundness of the NIZK proof system in experiment Expt (i.e, the adversary produces an accepting proof of a false statement). More precisely, let $(\psi_1, \psi_2, \ldots, \psi_{p(k)})$ denote the tuple of ciphertexts that A_2 returns in the experiment Expt. Let $\mathtt{badNIZK}(\mathsf{Expt})$ denote the following event: In experiment Expt, there exists an index j such that the NIZK proof in ψ_j is accepted by the verifier V, but all the k ciphertexts that are part of ψ_j do not decrypt to the same value (in other words, ψ_j contains an *accepting* proof of a *false* statement).

In the subclaims below, we show that $\mathtt{badNIZK}(\mathsf{NME}^{(j)}_b)$ happens only with negligible probability.

SubClaim. For $b \in \{0,1\}$, $\Pr[\mathtt{badNIZK}(\mathsf{NME}_b)] = \mathsf{negl}(k)$

Proof. Suppose, for contradiction, that this is not true. That is, there is a polynomial $q(k)$ such that $\Pr[\mathtt{badNIZK}(\mathsf{NME}_b)] \geq \frac{1}{q(k)}$. Then, we construct a machine A_s that violates the soundness of the proof system $(\mathcal{D}, P, V)$ with probability at least $\frac{1}{p(k)q(k)}$. On input a public parameter PP, A_s works as follows:

1. Simulate the experiment NME_b using PP, until A_2 outputs $p(k)$ ciphertexts. Note that A_s does not need to know the secret parameter SP to perform these steps. This is because by the definition of NME_b, SP is used only after A_2 outputs the ciphertexts.

2. A_s picks one of the ciphertexts at random, say $[\boldsymbol{c}, \pi, \text{VKSIG}, \sigma]$ and outputs the pair $(\boldsymbol{c}, \pi)$.

The probability that A_s outputs a false statement and an accepting proof pair is, by our assumption, at least $\frac{1}{p(k)q(k)}$, which is a contradiction to the soundness of $(\mathcal{D}, P, V)$. $\square$

SubClaim. For $b \in \{0,1\}$, $\Pr[\texttt{badNIZK}(\mathsf{NME}_b^{(1)})] = \Pr[\texttt{badNIZK}(\mathsf{NME}_b^{(2)})] = \mathsf{negl}(k)$.

Proof. We start by noting that $\Pr[\texttt{badNIZK}(\mathsf{NME}_b^{(1)})] = \Pr[\texttt{badNIZK}(\mathsf{NME}_b^{(2)})]$. This follows because the adversary's view in experiments $\mathsf{NME}_b^{(1)}$ and $\mathsf{NME}_b^{(2)}$ are identical until the point when the adversary A_2 outputs the ciphertexts. We proceed to show that for $b \in \{0,1\}$, $\Pr[\texttt{badNIZK}(\mathsf{NME}_b^{(1)})]$ is negligible in k. This is shown by an argument similar to the one used in the proof of Claim 1. Assume, for contradiction, that $\Pr[\texttt{badNIZK}(\mathsf{NME}_b^{(1)})]$ is non-negligible. Then, we construct a pair of machines $(A_{\mathsf{zk}}, D_{\mathsf{zk}})$ that violate the adaptive zero-knowledge of the proof system $(\mathcal{D}, P, V)$.

On input a public parameter PP for the NIZK proof system, A_{zk} and D_{zk} work exactly as in the proof of Claim 1, except that in Step 3, when A_2 returns a sequence of ciphertexts $(\psi_1, \ldots, \psi_{p(k)})$, D_{zk} looks for a ciphertext ψ_i such that not all the components of ψ_i decrypt to the same message, and the NIZK proof in ψ_i is accepting. If there exists such an i, then D_{zk} returns "Fail" and otherwise returns "OK".

Note that by definition, when D_{zk} receives a real proof, it outputs "Fail" with probability $\Pr[\texttt{badNIZK}(\mathsf{NME}_b)]$. On the other hand, when run on a simulated proof, it outputs "Fail" with probability $\Pr[\texttt{badNIZK}(\mathsf{NME}_b^{(1)})]$. However, in Step 1a, we showed that the former probability is negligible. If the latter probability is non-negligible, then D_{zk} distinguishes between a simulated proof and a real proof, contradicting the adaptive zero-knowledge property of the proof system $(\mathcal{D}, P, V)$. $\square$

Let $\psi_i = [\boldsymbol{c}_i, \pi_i, \text{VKSIG}_i, \sigma_i]$ denote the i^{th} ciphertext returned by A_2. Define $\texttt{badSig}(\mathsf{NME}_\mathsf{b}^{(\mathsf{j})})$ to be the event that, in experiment $\mathsf{NME}_b^{(j)}(\Pi, A, k, p(k))$, there exists an index i such that $\text{VKSIG}_i = \text{VKSIG}$ and $\mathsf{Ver}(\text{VKSIG}_i, \boldsymbol{c}_i, \pi_i) = \text{ACCEPT}$. Since the signature scheme is (strongly) existentially unforgeable, it follows that, for $b \in \{0,1\}$ and $j \in \{1,2\}$, $\Pr[\texttt{badSig}(\mathsf{NME}_b^{(j)})] = \mathsf{negl}(k)$.

Let $\texttt{badKey}(\mathsf{NME}_b^{(j)})$ denote the event that for one of the public keys, say $\hat{\text{PK}}$, generated in the experiment $\mathsf{NME}_b^{(j)}$, there exists a pair of messages m, m' and random coins r, r' such that $m \neq m'$ and $\mathsf{Enc}(\hat{pk}, m, r) = \mathsf{Enc}(\hat{pk}, m', r')$. Since the encryption scheme used is perfectly correct, by the union bound, we have $\Pr[\texttt{badKey}(\mathsf{NME}_b^{(j)})] = \mathsf{negl}(k)$.

Let $\texttt{fail}_b(\cdot)$ denote the event $\texttt{badNIZK}(\cdot) \vee \texttt{badSig}(\cdot) \vee \texttt{badKey}(\cdot)$. It follows, by a union bound, that $\Pr[\texttt{fail}_b(\mathsf{NME}_b^{(j)})] = \mathsf{negl}(k)$, for $j \in \{1,2\}$.

We show that conditioned on the event $\texttt{fail}_b(\mathsf{NME}_b^{(j)})$ (for $j \in \{1,2\}$) not happening, $\mathsf{NME}_b^{(1)}$ and $\mathsf{NME}_b^{(2)}$ are identical. Note that the view of A in both the experiments is (syntactically) the same. Since $\texttt{badSig}(\mathsf{NME}_b^{(j)})$ does not happen, A uses a different verification key in all the ciphertexts ψ_i it returns. This means that $\mathsf{NME}_b^{(j)}$ can decrypt at least *one* of the components of each ψ_i, using a secret-key it knows, to get a message m_i. Since $\texttt{badNIZK}(\mathsf{NME}_b^{(j)})$ does not happen, m_i must be the message that is encrypted in all the other components of ψ_i too. Thus, ψ_i is a *valid* encryption of m_i. Also, since $\texttt{badKey}(\mathsf{NME}_b^{(j)})$ does not happen, m_i is the *unique* such message. Thus the tuple of messages returned in both $\mathsf{NME}_b^{(1)}$ and $\mathsf{NME}_b^{(2)}$ are exactly the same, and thus the outputs of $\mathsf{NME}_b^{(1)}$ and $\mathsf{NME}_b^{(2)}$ are identical.

Combining the above with the fact that the events $\texttt{fail}_b(\cdot)$ occur with a negligible probability, we have $\mathsf{NME}_b^{(1)}(\Pi, A, k, p(k)) \overset{s}{\approx} \mathsf{NME}_b^{(2)}(\Pi, A, k, p(k))$. □

Claim 3 *For every p.p.t. machine A, there exists a p.p.t. machine B such that for $b \in \{0,1\}$,*

$$\left\{\mathsf{NME}_b^{(2)}(\Pi, A, k, p(k))\right\}_{k\in\mathbb{N}} \equiv \left\{\mathsf{IND}_b(E', B, k)\right\}_{k\in\mathbb{N}}$$

Proof. The machine B is constructed as follows. B simply simulates the experiment $\mathsf{NME}_b^{(2)}$, except that instead of generating $\textsc{pk}$ by itself, it uses $\textsc{pk} = \{\textsc{pk}_i\}_{i=1}^k$ received from the outside. Let (m_0, m_1) be the pair of messages the adversary A_1 returns. B then outputs (m_0, m_1) and receives a challenge ciphertext c_b from the outside. B performs the same operations as the experiment $\mathsf{NME}_b^{(2)}$ to generate the challenge ciphertext C_b for A_2. Finally, A_2 returns a sequence of ciphertexts $(\psi_1, \psi_2, \ldots, \psi_{p(k)})$. B decrypts these ciphertexts just as in $\mathsf{NME}_b^{(2)}$ and outputs the plaintexts. (Note that $\mathsf{NME}_b^{(2)}$ uses only $\textsc{sk}'$ and not $\textsc{sk}$ in order to decrypt the messages).

It is easy to see that B simulates the experiment $\mathsf{NME}_b^{(2)}$ perfectly using the public-keys and ciphertexts received from the outside, and thus

$$\left\{\mathsf{NME}_b^{(2)}(\Pi, A, k, p(k))\right\}_{k\in\mathbb{N}} \equiv \left\{\mathsf{IND}_b(E', B, k)\right\}_{k\in\mathbb{N}}$$

□

To conclude the proof, we combine the last three claims to conclude that for every p.p.t. adversary A, there is a p.p.t. adversary B such that

$$\begin{aligned}\mathsf{NME}_b(\Pi, A, k, p(k)) \overset{c}{\approx} \mathsf{NME}_b^{(1)}(\Pi, A, k, p(k)) &\overset{s}{\approx} \mathsf{NME}_b^{(2)}(\Pi, A, k, p(k)) \\ &\equiv \mathsf{IND}_b(E', B, k)\end{aligned}$$

Since by the semantic security of E', $\mathsf{IND}_0(E', B, k) \overset{c}{\approx} \mathsf{IND}_1(E', B, k)$, it holds that $\mathsf{NME}_0(\Pi, A, k, p(k)) \overset{c}{\approx} \mathsf{NME}_1(\Pi, A, k, p(k))$. □

Acknowledgments We thank Cynthia Dwork, Shafi Goldwasser, Yehuda Lindell, and Silvio Micali for delicious comments. We are especially grateful to Yehuda Lindell for his diligent help.

References

[AD97] Miklós Ajtai and Cynthia Dwork. A public-key cryptosystem with worst-case/average-case equivalence. In *STOC*, pages 284–293, 1997.

[BDPR98] M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway. Relations among notions of security for public-key encryption schemes. In *CRYPTO*, 1998.

[Blu86] Manuel Blum. How to prove a theorem so no one can claim it. In *Proc. of The International Congress of Mathematicians*, pages 1444–1451, 1986.

[BS99] Mihir Bellare and Amit Sahai. Non-malleable encryption: Equivalence between two notions, and an indistinguishability-based characterization. In *CRYPTO*, pages 519–536, 1999.

[CD00] Jan Camenisch and Ivan B. Damgård. Verifiable encryption, group encryption, and their applications to group signatures and signature sharing schemes. In *ASIACRYPT*, pages 331–345, 2000.

[CDS94] Ronald Cramer, Ivan Damgard, and Berry Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *CRYPTO*, pages 174–187, 1994.

[CS98] Ronald Cramer and Victor Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In *CRYPTO*, pages 13–25, 1998.

[CS02] Ronald Cramer and Victor Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In *EUROCRYPT*, pages 45–64, 2002.

[DDN00] Danny Dolev, Cynthia Dwork, and Moni Naor. Nonmalleable cryptography. *SIAM J. Comput.*, 30(2):391–437, 2000.

[Dwo99] Cynthia Dwork. The non-malleability lectures. Course notes for Stanford CS 359, 1999. http://theory.stanford.edu/g̃durf/cs359-s99/.

[ES02] Edith Elkind and Amit Sahai. A unified methodology for constructing public-key encryption schemes secure against adaptive chosen-ciphertext attack. ePrint Archive 2002/042, 2002.

[GL03] Rosario Gennaro and Yehuda Lindell. A framework for password-based authenticated key exchange. In *EUROCRYPT*, pages 524–543, 2003.

[GM84] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984.

[Gol04] Oded Goldreich. *Foundations of Cryptography, Volume 2*. Cambridge University Press, 2004.

[KMO89] Joe Kilian, Silvio Micali, and Rafail Ostrovsky. Minimum resource zero-knowledge proofs. In *FOCS*, pages 474–479, 1989.

[Lam79] Leslie Lamport. Constructing digital signatures from a one-way function. Technical Report CSL-98, SRI International, October 1979.

[Nao91] Naor. Bit commitment using pseudorandomness. *J. of Cryptology*, 4, 1991.

[Nao04] Moni Naor. A taxonomy of encryption scheme security. 2004.

[PS05] Rafael Pass and Abhi Shelat. Unconditional characterizations of non-interactive zero-knowledge. In *CRYPTO*, pages 118–134, 2005.

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *STOC*, pages 84–93, 2005.
[Rom90] J. Rompel. One-way functions are necessary and sufficient for secure signatures. In *STOC*, pages 387–394, 1990.

A DDN-Lite Does Not Achieve Definition 3

DDN-Lite is a candidate single-message non-malleable encryption scheme based on any semantically secure scheme which has been informally discussed in [Nao04, Dwo99]. In this section, we show that DDNLite does not meet our stronger definition of non-malleability (Definition 3). We remark that DDNLite can, however, be proven secure under the DDN and equivalent [BDPR98, BS99]) definitions of non-malleability because in weaker notions, the adversary is considered to have lost the game if she produces invalid ciphertexts in the experiment.

Let us briefly summarize the DDN-Lite scheme. The public key consists of k pairs of encryption keys (just as in our scheme) and a universal one-way hash function h. To encrypt a message m, generate a key pair $(\text{SKSIG}, \text{VKSIG})$ for a one-time signature scheme, hash the verification key VKSIG to get $(b_1, \ldots, b_k) \leftarrow h(\text{VKSIG})$, compute the ciphertexts $c_i = E_{\text{PK}_i^{b_i}}(m)$, compute the signature $\sigma = \mathsf{Sign}_{\text{SKSIG}}(c_1, \ldots, c_k)$ and output the tuple $[(c_1, c_2, \ldots, c_k), \text{VKSIG}, \sigma]$. To decrypt a ciphertext c parsed as $[(c_1, c_2, \ldots, c_k), \text{VKSIG}, \sigma]$, first verify the signature σ and output $\perp$ if Ver rejects. Otherwise, decrypt the c_i's with the corresponding secret-keys to get corresponding messages m_i. If all the m_i's are equal, then output one of the m_i's, else output $\perp$.

Claim 4 *The DDN-lite encryption scheme does not satisfy Definition 3.*

Proof. We specify an adversary $A = (A_1, A_2)$ such that the two experiments $\{\mathsf{NME}_0(\Pi, A, k, p(k))\}_{k\in\mathbb{N}}$ and $\{\mathsf{NME}_1(\Pi, A, k, p(k))\}_{k\in\mathbb{N}}$ are distinguishable. A works as follows:

1. A_1 outputs two arbitrary messages (m_0, m_1) and no state information.
2. On input ciphertext $c = \big[(e_1, \ldots, e_k), \text{VKSIG}, \sigma\big]$, let $(b_1, \ldots, b_k) \leftarrow h(\text{VKSIG})$. A_2 produces a new ciphertext c' as follows. Generate a new signing key $(\text{SKSIG}', \text{VKSIG}')$. Compute $(b'_1, \ldots, b'_k) \leftarrow h(\text{VKSIG}')$. Output ciphertexts $c' = ((x_1, \ldots, x_k), \text{VKSIG}', \sigma')$ where

$$x_i = \begin{cases} e_i & \text{if } b'_i = b_i \\ E_{\text{PK}_i^{b'_i}}(m_0) & \text{otherwise} \end{cases}$$

and σ' is the signature of $(x_1, \ldots, x_k)$ under the signing key SKSIG'.

Now, notice that $\mathsf{NME}_0(\Pi, A, k, \ell) = m_0$ and $\mathsf{NME}_1(\Pi, A, k, \ell) = \perp$ which can be easily distinguished. □