Towards Plaintext-Aware Public-Key Encryption without Random Oracles

Mihir Bellare and Adriana Palacio

Dept. of Computer Science & Engineering, University of California, San Diego
9500 Gilman Drive, La Jolla, CA 92093, USA.
Email: {mihir,apalacio}@cs.ucsd.edu
URL: http://www-cse.ucsd.edu/users/{mihir,apalacio}

Abstract. We consider the problem of defining and achieving plaintext-aware encryption without random oracles in the classical public-key model. We provide definitions for a hierarchy of notions of increasing strength: PA0, PA1 and PA2, chosen so that PA1+IND-CPA $\rightarrow$ IND-CCA1 and PA2+IND-CPA $\rightarrow$ IND-CCA2. Towards achieving the new notions of plaintext awareness, we show that a scheme due to Damgård [12], denoted DEG, and the "lite" version of the Cramer-Shoup scheme [11], denoted CS-lite, are both PA0 under the DHK0 assumption of [12], and PA1 under an extension of this assumption called DHK1. As a result, DEG is the most efficient proven IND-CCA1 scheme known.

1 Introduction

The theory of encryption is concerned with defining and implementing notions of security for encryption schemes [22, 23, 17, 25, 27, 15]. One of the themes in its history is the emergence of notions of security of increasing strength that over time find applications and acceptance.

Our work pursues, from the same perspective, a notion that is stronger than any previous ones, namely plaintext awareness. Our goal is to strengthen the foundations of this notion by lifting it out of the random-oracle model where it currently resides. Towards this end, we provide definitions of a hierarchy of notions of plaintext awareness, relate them to existing notions, and implement some of them. We consider this a first step in the area, however, since important questions are left unresolved. We begin below by reviewing existing work and providing some motivation for our work.

1.1 Background

Intuitively, an encryption scheme is plaintext aware (PA) if the "only" way that an adversary can produce a valid ciphertext is to apply the encryption algorithm to the public key and a message. In other words, any adversary against a PA scheme that produces a ciphertext "knows" the corresponding plaintext.

Random-Oracle model work. The notion of PA encryption was first suggested by Bellare and Rogaway [6], with the motivation that PA+IND-CPA should imply IND-CCA2. That is, security against chosen-plaintext attack coupled with plaintext awareness should imply security against adaptive chosen-ciphertext attack. The intuition, namely, that if an adversary knows the plaintext corresponding to a ciphertext it produces, then a decryption oracle must be useless to it, goes back to [8, 9]. Bellare and Rogaway [6] provided a formalization of PA in the random oracle (RO) model. They asked that for every adversary A taking the public key and outputting a ciphertext, there exist an extractor that, given the same public key and a transcript of the interaction of A with its RO, is able to decrypt the ciphertext output by A. We will refer to this notion as PA-BR.

Subsequently, it was found that PA-BR was too weak for PA-BR+IND-CPA to imply IND-CCA2. Bellare, Desai, Pointcheval and Rogaway [4] traced the cause of this to the fact that PA-BR did not capture the ability of the adversary to obtain ciphertexts via eavesdropping on communications made to the receiver. (Such eavesdropping can put into the adversary's hands ciphertexts whose decryptions it does not know, lending it the ability to create other ciphertexts whose decryptions it does not know.) They provided an appropriately enhanced definition (still in the RO model) that we denote by PA-BDPR, and showed that PA-BDPR+IND-CPA $\rightarrow$ IND-CCA2.

Plaintext awareness is exploited, even though typically implicitly rather than explicitly, in the proofs of the IND-CCA2 security of numerous RO-model encryption schemes, e.g., [16, 28, 7].

PA and the RO model. By restricting the above-mentioned RO-model definitions to schemes and adversaries that do not query the RO, one obtains natural counterpart standard (i.e., non-RO) model definitions of PA. These standard-model definitions turn out, however, not to be achievable without sacrificing privacy, because the extractor can simply be used for decryption. This indicates that the use of the RO model in the definitions of [6, 4] is central.

Indeed, PA as per [6, 4] is "designed" for the RO model in the sense that the definition aims to capture certain properties of certain RO-model schemes, namely, the fact that possession of the transcript of the interaction of an adversary with its RO permits decryption of ciphertexts formed by this adversary. It is not clear what counterpart this intuition has in the standard model.

The lack of a standard-model definition of PA results in several gaps. One such arises when we consider that RO-model PA schemes are eventually instantiated to get standard-model schemes. In that case, what property are these instantiated schemes even supposed to possess? There is no definition that we might even discuss as a target.

PA via key registration. PA without ROs was first considered by Herzog, Liskov and Micali [21], who define and implement it in an extension of the usual public-key setting. In their setting, the sender (not just the receiver) has a public key, and, in a key-registration phase that precedes encryption, proves knowledge of the corresponding secret key to a key-registration authority via an interactive proof of knowledge. Encryption is a function of the public keys of both the sender and the receiver, and the PA extractor works by extracting the sender secret key using the knowledge extractor of the interactive proof of knowledge.

Their work also points to an application of plaintext-aware encryption where the use of the latter is crucial in the sense that IND-CCA2-secure encryption does not suffice, namely to securely instantiate the ideal encryption functions of the Dolev-Yao model [14].

1.2 Our goals and motivation

The goal of this work is to provide definitions and constructions for plaintext-aware public-key encryption in the standard and classical setting of public-key encryption, namely the one where the receiver (but not the sender) has a public key, and anyone (not just a registered sender) can encrypt a message for the receiver as a function of the receiver's public key. In this setting there is no key-registration authority or key-registration protocol akin to [21].

Motivations include the following. As in the RO model, we would like a tool enabling the construction of public-key encryption schemes secure against chosen-ciphertext attack. We would also like to have some well-defined notion that can be viewed as a target for instantiated RO-model PA schemes. (One could then evaluate these schemes with regard to meeting the target.)

Additionally, we would like to enable the possibility of instantiating the ideal encryption functions of the Dolev-Yao model [14] without recourse to either random oracles or the key-registration model. Note that the last is an application where, as per [21], PA is required and IND-CCA2 does not suffice, meaning plaintext-awareness is crucial. (However, see also [1].)

As we will see later, consideration of PA in the standard model brings other benefits, such as some insight, or at least an alternative perspective, on the design of existing encryption schemes secure against chosen-ciphertext attack. Let us now discuss our contributions.

1.3 Definitions

The first contribution of this paper is to provide definitions for plaintext-aware encryption in the standard model and standard public-key setting.

Overview. We provide a hierarchy consisting of three notions of increasing strength that we denote by PA0, PA1 and PA2. There are several motivations for this. One is that these will be seen (in conjunction with IND-CPA) to imply security against chosen-ciphertext attacks of different strengths. Another is that, as will become apparent, PA is difficult to achieve, and progress can be made by first achieving it in weaker forms. Finally, it is useful, pedagogically, to bring in new definitional elements incrementally.

A closer look. Our basic definitional framework considers a polynomial-time adversary $\boldsymbol{C}$, called a ciphertext creator, that takes input the public key and can query ciphertexts to an oracle. A polynomial-time algorithm $\boldsymbol{C}^*$ is said to be a successful extractor for $\boldsymbol{C}$ if it can provide replies to the oracle queries of $\boldsymbol{C}$

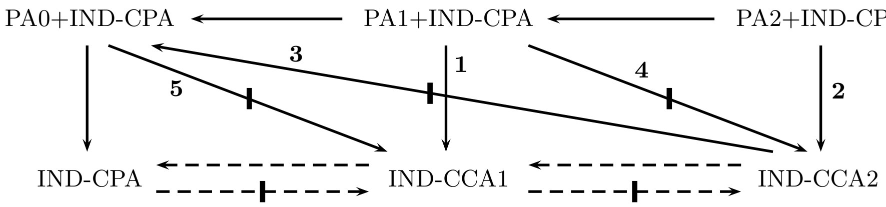

Fig. 1. *An arrow is an implication, and, in the directed graph given by the arrows, there is a path from A to B if and only if A implies B. The hatched arrows represent separations. Solid lines represent results from this paper, while dashed lines represent results from prior work [4, 15]. The number on an arrow or hatched arrow refers to the theorem in this paper that establishes this relationship. Absence of a number on a solid arrow means the result is trivial.*

---

that are computationally indistinguishable from those provided by a decryption oracle.

An important element of the above framework is that the extractor gets as input *the same public key as the ciphertext creator, as well as the coin tosses of the ciphertext creator*. This reflects the intuition that the extractor is the "subconscious" of the adversary, and begins with exactly the same information as the adversary itself.

We say that an encryption scheme is PA0 (respectively, PA1) if there exists a successful extractor for any ciphertext creator that makes only a single oracle query (respectively, a polynomial number of oracle queries).

Eavesdropping capability in PA2 is captured by providing the ciphertext creator $\boldsymbol{C}$ with an additional oracle that returns ciphertexts, but care has to be taken in defining this oracle. It does *not* suffice to let it be an encryption oracle because we want to model the ability of the adversary to obtain ciphertexts whose decryptions it may not know. Our formalization of PA2 allows the additional oracle to compute a plaintext, as a function of the query made to it and coins unknown to $\boldsymbol{C}$, and return the encryption of this plaintext to $\boldsymbol{C}$.

Formal definitions of PA0, PA1 and PA2 are in Section 3.

1.4 Relations

PA by itself is not a notion of privacy, and so we are typically interested in PA coupled with the minimal notion of privacy, namely IND-CPA [22, 23]. We consider six notions, namely, PA0+IND-CPA, PA1+IND-CPA and PA2+IND-CPA, on the one hand, and the standard notions of privacy IND-CPA, IND-CCA1 [25] and IND-CCA2 [27], on the other. We provide implications and separations among these six notions in the style of [4, 15]. The results are depicted in Figure 1. For notions A, B, an implication, represented by A $\rightarrow$ B, means that every encryption scheme satisfying notion A also satisfies notion B, and a separation, represented by A $\not\rightarrow$ B, means that there exists an encryption scheme satisfying notion A but not satisfying notion B. (The latter assumes there exists some encryption scheme satisfying notion A, since otherwise the question is vacuous.)

Figure 1 shows a minimal set of arrows and hatched arrows, but the relation between any two notions is resolved by the given relations. For example, IND-CCA1 $\not\rightarrow$ PA1+IND-CPA, because, otherwise, there would be a path from IND-CCA2 to PA0+IND-CPA, contradicting the hatched arrow labeled 3. Similarly, we get PA0 $\not\rightarrow$ PA1 $\not\rightarrow$ PA2, meaning the three notions of plaintext awareness are of increasing strength.

The main implications are that PA1+IND-CPA implies IND-CCA1 and PA2+IND-CPA implies IND-CCA2. The PA1+IND-CPA $\rightarrow$ IND-CCA1 result shows that even a notion of PA not taking eavesdropping adversaries into account is strong enough to imply security against a significant class of chosen-ciphertext attacks. Since the PA+IND-CPA $\rightarrow$ IND-CCA2 implication has been a motivating target for definitions of PA, the PA2+IND-CPA $\rightarrow$ IND-CCA2 result provides some validation for the definition of PA2.

Among the separations, we note that IND-CCA2 does not imply PA0, meaning even the strongest form of security against chosen-ciphertext attack is not enough to guarantee the weakest form of plaintext awareness.

1.5 Constructions

The next problem we address is to find provably-secure plaintext-aware encryption schemes.

Approaches. A natural approach to consider is to include a non-interactive zero-knowledge proof of knowledge [13] of the message in the ciphertext. However, as we explain in [2], this fails to achieve PA.

As such approaches are considered and discarded, it becomes apparent that achieving even the weaker forms of PA in the standard (as opposed to RO) model may be difficult. We have been able to make progress, however, under some strong assumptions that we now describe.

DHK assumptions. Let G be the order q subgroup of $\mathbb{Z}^*_{2q+1}$, where $q, 2q+1$ are primes, and let g be a generator of G. Damgård [12] introduced and used an assumption that states, roughly, that an adversary given g^a and outputting a pair of the form (g^b, g^{ab}) must "know" b. The latter is captured by requiring an extractor that given the adversary coins and inputs can output b. We call our formalization of this assumption (cf. Assumption 2) DHK0.[1] We also introduce an extension of this assumption called DHK1 (cf. Assumption 1), in which the adversary does not just output one pair (g^b, g^{ab}), but instead interacts with the

[1] Another formalization, called DA-1, is used by Hada and Tanaka [19]. (We refer to the full version of their paper [19], which points out that the formalization of the preliminary version [20] is wrong.) This differs from DHK0 in being for a non-uniform setting. DA-1 is called KEA1 by [5], based on Naor's terminology [24]: KEA stands for "knowledge of exponent." Hada and Tanaka [19] also introduced and used another assumption, that they call DA-2 and is called KEA2 in [5], but the latter show that this assumption is false. The DHK0/DA-1/KEA1 assumptions, to the best of our knowledge, are not known to be false.

extractor, feeding it such pairs adaptively and each time expecting back the discrete logarithm of the first component of the pair.

The DEG scheme. Damgård presented a simple ElGamal variant that we call DEG. It is efficient, requiring only three exponentiations to encrypt and two to decrypt.

We prove that DEG is PA0 under the DHK0 assumption and PA1 under the DHK1 assumption. Since DEG is easily seen to be IND-CPA under the DDH assumption, and we saw above that PA1+IND-CPA $\rightarrow$ IND-CCA1, a consequence is that DEG is IND-CCA1 assuming DHK1 and DDH. DEG is in fact the most efficient IND-CCA1 scheme known to date to be provably secure in the standard model.

Damgård [12] claims that DEG meets a notion of security under ciphertext attack that we call RPR-CCA1, assuming DHK0 and assuming the ElGamal scheme meets a notion called RPR-CPA. (Both notions are recalled in the full version of this paper [2], and are weaker than IND-CCA1 and IND-CPA, respectively). As we explain in [2], his proof has a flaw, but his overall approach and intuition are valid, and the proof can be fixed by simply assuming DHK1 in place of DHK0. In summary, our contribution is (1) to show that DEG meets a stronger and more standard notion of security than RPR-CCA1, namely IND-CCA1, and (2) to show it is PA0 and PA1, indicating that it has even stronger properties, and providing some formal support for the intuition given in [12] about the security underlying the scheme.

CS-lite. CS-lite is a simpler and more efficient version of the Cramer-Shoup encryption scheme [11] that is IND-CCA1 under the DDH assumption. We show that CS-lite is PA0 under the DHK0 assumption and PA1 under the DHK1 assumption. (IND-CPA under DDH being easy to see, this again implies CS-lite is IND-CCA1 under DHK1 and DDH, but in this case the conclusion is not novel.) What we believe is interesting about our results is that they show that some form of plaintext awareness underlies the CS-lite scheme, and this provides perhaps an alternative viewpoint on the source of its security. We remark, however, that DEG is more efficient than CS-lite.

Warning and discussion. DHK0 and DHK1 are strong and non-standard assumptions. As pointed out by Naor [24], they are not efficiently falsifiable. (However, such assumptions can be shown to be false as exemplified in [5]). However standard-model schemes, even under strong assumptions, might provide better guarantees than RO model schemes, for we know that the latter may not provide real-world security guarantees at all [10, 26, 18, 3]. Also, PA without random oracles is challenging to achieve, and we consider it important to "break ground" by showing it is possible, even if under strong assumptions.

Open questions. The central open question is to find an IND-CPA+PA2 scheme provably secure under some plausible assumption. We suggest, in particular, that an interesting question is whether the Cramer-Shoup scheme, already known to be IND-CCA2, is PA2 under some appropriate assumption. (Intuitively, it seems to be PA2.) It would also be nice to achieve PA0 or PA1 under weaker and more standard assumptions than those used here.

2 Notation and standard definitions

We let $\mathbb{N} = \{1, 2, 3, \ldots\}$. We denote by ε the empty string, by $|x|$ the length of a string x, by $\bar{x}$ the bitwise complement of x, by "$\|$" the string-concatenation operator, and by 1^k the string of $k \in \mathbb{N}$ ones. We denote by $[\,]$ the empty list. Given a list L and an element x, $L \,@\, x$ denotes the list consisting of the elements in L followed by x. If S is a randomized algorithm, then $S(x, y, \ldots ; R)$ denotes its output on inputs $x, y, \ldots$ and coins R; $s \stackrel{\$}{\leftarrow} S(x, y, \ldots)$ denotes the result of picking R at random and setting $s = S(x, y, \ldots ; R)$; and $[S(x, y, \ldots)]$ denotes the set of all points having positive probability of being output by S on inputs $x, y, \ldots$. Unless otherwise indicated, an algorithm is randomized.

Encryption schemes. We recall the standard syntax. An asymmetric (also called public-key) encryption scheme is a tuple $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D}, \mathrm{MsgSp})$ whose components are as follows. The polynomial-time key-generation algorithm $\mathcal{K}$ takes input 1^k, where $k \in \mathbb{N}$ is the security parameter, and returns a pair (pk, sk) consisting of a public key and matching secret key. The polynomial-time encryption algorithm $\mathcal{E}$ takes a public key pk and a message M to return a ciphertext C. The deterministic, polynomial-time decryption algorithm $\mathcal{D}$ takes a secret key sk and a ciphertext C to return either a message M or the special symbol $\perp$ indicating that the ciphertext is invalid. The polynomial-time computable message-space function MsgSp associates to each public key pk a set $\mathrm{MsgSp}(pk)$ called the message space of pk. It is required that for every $k \in \mathbb{N}$

$$\Pr\left[\, (pk, sk) \stackrel{\$}{\leftarrow} \mathcal{K}(1^k)\, ;\; M \stackrel{\$}{\leftarrow} \mathrm{MsgSp}(pk)\, ;\; C \stackrel{\$}{\leftarrow} \mathcal{E}(pk, M) \;:\; \mathcal{D}(sk, C) = M \,\right] = 1\,.$$

Standard security notions. We recall the definitions of IND-CPA, IND-CCA1, and IND-CCA2 security that originate in [22], [25], and [27], respectively. We use the formalizations of [4]. Let $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D}, \mathrm{MsgSp})$ be an asymmetric encryption scheme, let $k \in \mathbb{N}$ and $b \in \{0, 1\}$. Let $\boldsymbol{X}$ be an algorithm with access to an oracle. For $\mathrm{aaa} \in \{\mathrm{cpa}, \mathrm{cca1}, \mathrm{cca2}\}$, consider the following experiment

Experiment $\mathbf{Exp}^{\text{ind-aaa-}b}_{\mathcal{AE},\boldsymbol{X}}(k)$

$(pk, sk) \stackrel{\$}{\leftarrow} \mathcal{K}(1^k)\, ;\; (M_0, M_1, \mathrm{St}) \stackrel{\$}{\leftarrow} \mathbf{X}^{\mathcal{O}_1(\cdot)}(\mathsf{find}, pk)\, ;\; C \stackrel{\$}{\leftarrow} \mathcal{E}(pk, M_b)$

$d \leftarrow \mathbf{X}^{\mathcal{O}_2(\cdot)}(\mathsf{guess}, C, \mathrm{St})\, ;\;$ Return d

where

If $\mathrm{aaa} = \mathrm{cpa}$ then $\mathcal{O}_1(\cdot) = \varepsilon$ and $\mathcal{O}_2(\cdot) = \varepsilon$
If $\mathrm{aaa} = \mathrm{cca1}$ then $\mathcal{O}_1(\cdot) = \mathcal{D}_{sk}(\cdot)$ and $\mathcal{O}_2(\cdot) = \varepsilon$
If $\mathrm{aaa} = \mathrm{cca2}$ then $\mathcal{O}_1(\cdot) = \mathcal{D}_{sk}(\cdot)$ and $\mathcal{O}_2(\cdot) = \mathcal{D}_{sk}(\cdot)$

In each case it is required that $M_0, M_1 \in \mathrm{MsgSp}(pk)$ and $|M_0| = |M_1|$. In the case of IND-CCA2, it is also required that $\boldsymbol{X}$ not query its decryption oracle

Experiment $\mathbf{Exp}^{\text{pa1-d}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}}(k)$

$(pk, sk) \stackrel{\$}{\leftarrow} \mathcal{K}(1^k)$; $x \stackrel{\$}{\leftarrow} \boldsymbol{C}^{\mathcal{D}(sk,\cdot)}(pk)$; $d \stackrel{\$}{\leftarrow} \boldsymbol{D}(x)$; Return d

Experiment $\mathbf{Exp}^{\text{pa1-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D},\boldsymbol{C}^*}(k)$

$(pk, sk) \stackrel{\$}{\leftarrow} \mathcal{K}(1^k)$

Choose coins $R[\boldsymbol{C}], R[\boldsymbol{C}^*]$ for $\boldsymbol{C}, \boldsymbol{C}^*$, respectively ; $\mathrm{St}[\boldsymbol{C}^*] \leftarrow (pk, R[\boldsymbol{C}])$

Run $\boldsymbol{C}$ on input pk and coins $R[\boldsymbol{C}]$ until it halts, replying to its oracle queries as follows:

– If $\boldsymbol{C}$ makes query Q then

$(M, \mathrm{St}[\boldsymbol{C}^*]) \leftarrow \boldsymbol{C}^*(Q, \mathrm{St}[\boldsymbol{C}^*]; R[\boldsymbol{C}^*])$; Return M to $\boldsymbol{C}$ as the reply EndIf

Let x denote the output of $\boldsymbol{C}$; $d \stackrel{\$}{\leftarrow} \boldsymbol{D}(x)$; Return d

Fig. 2. Experiments used to define PA1 and PA0.

with ciphertext C. We call $\boldsymbol{X}$ an *ind-aaa-adversary.* The *ind-aaa-advantage* of $\boldsymbol{X}$ is

$$\mathbf{Adv}^{\text{ind-aaa}}_{\mathcal{AE},\boldsymbol{X}}(k) \;=\; \Pr\left[\mathbf{Exp}^{\text{ind-aaa-1}}_{\mathcal{AE},\boldsymbol{X}}(k) = 1\right] - \Pr\left[\mathbf{Exp}^{\text{ind-aaa-0}}_{\mathcal{AE},\boldsymbol{X}}(k) = 1\right].$$

For AAA $\in$ {CPA, CCA1, CCA2}, $\mathcal{AE}$ is said to be IND-AAA secure if $\mathbf{Adv}^{\text{ind-aaa}}_{\mathcal{AE},\boldsymbol{X}}(\cdot)$ is negligible for every polynomial-time ind-aaa-adversary $\boldsymbol{X}$.

3 New notions of plaintext awareness

In this section we provide our formalizations of plaintext-aware encryption. We provide the formal definitions first and explanations later. We begin with PA1, then define PA0 via this, and finally define PA2.

Definition 1. [**PA1**] Let $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D}, \mathrm{MsgSp})$ be an asymmetric encryption scheme. Let $\boldsymbol{C}$ be an algorithm that has access to an oracle, takes as input a public key pk, and returns a string. Let $\boldsymbol{D}$ be an algorithm that takes a string and returns a bit. Let $\boldsymbol{C}^*$ be an algorithm that takes a string and some state information, and returns a message or the symbol $\perp$, and a new state. We call $\boldsymbol{C}$ a *ciphertext-creator* adversary, $\boldsymbol{D}$ a *distinguisher*, and $\boldsymbol{C}^*$ a *pa1-extractor.* For $k \in \mathbb{N}$, we define the experiments shown in Figure 2. The *pa1-advantage* of $\boldsymbol{C}$ relative to $\boldsymbol{D}$ and $\boldsymbol{C}^*$ is

$$\mathbf{Adv}^{\text{pa1}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D},\boldsymbol{C}^*}(k) \;=\; \Pr\left[\mathbf{Exp}^{\text{pa1-d}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D}}(k) = 1\right] - \Pr\left[\mathbf{Exp}^{\text{pa1-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D},\boldsymbol{C}^*}(k) = 1\right].$$

We say that $\boldsymbol{C}^*$ is a *successful pa1-extractor for* $\boldsymbol{C}$ if for every polynomial-time distinguisher $\boldsymbol{D}$ the function $\mathbf{Adv}^{\text{pa1}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{D},\boldsymbol{C}^*}(\cdot)$ is negligible. We say $\mathcal{AE}$ is *PA1 secure* if for any polynomial-time ciphertext creator there exists a successful polynomial-time pa1-extractor. ▌

Definition 2. [**PA0**] Let $\mathcal{AE}$ be an asymmetric encryption scheme. We call a ciphertext-creator adversary that makes *exactly one* oracle query a *pa0 ciphertext* *creator*. We call a pa1-extractor for a pa0 ciphertext creator a *pa0-extractor*. We say that $\mathcal{AE}$ is *PA0 secure* if for any polynomial-time pa0 ciphertext creator there exists a successful polynomial-time pa0-extractor. ▮

We now explain the ideas behind the above formalisms. The core of the formalization of plaintext awareness of asymmetric encryption scheme $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D}, \text{MsgSp})$ considers a polynomial-time ciphertext-creator adversary $\boldsymbol{C}$ that takes input a public key pk, has access to an oracle and returns a string. The adversary tries to distinguish between the cases that its oracle is $\mathcal{D}(sk, \cdot)$, or it is an *extractor* algorithm $\boldsymbol{C}^*$ that takes as input the same public key pk. PA1 security requires that there exist a polynomial-time $\boldsymbol{C}^*$ such that $\boldsymbol{C}$'s outputs in the two cases are indistinguishable. We allow $\boldsymbol{C}^*$ to be stateful, maintaining state $\text{St}[\boldsymbol{C}^*]$ across invocations. Importantly, $\boldsymbol{C}^*$ *is provided with the coin tosses of* $\boldsymbol{C}$; otherwise, $\boldsymbol{C}^*$ would be functionally equivalent to the decryption algorithm and thus could not exist unless $\mathcal{AE}$ were insecure with regard to providing privacy. We remark that this formulation is stronger than one not involving a distinguisher $\boldsymbol{D}$, in which $\boldsymbol{C}$ simply outputs a bit representing its guess, since $\boldsymbol{C}^*$ gets the coins of $\boldsymbol{C}$, but not the coins of $\boldsymbol{D}$.

PA0 security considers only adversaries that make a *single* query in their attempt to determine if the oracle is a decryption oracle or an extractor.

Definition 3. [**PA2**] Let $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D}, \text{MsgSp})$ be an asymmetric encryption scheme. Let $\boldsymbol{C}$ be an algorithm that has access to an oracle, takes as input a public key pk, and returns a string. Let $\boldsymbol{P}$ be an algorithm that takes a string and some state information, and returns a message and a new state. Let $\boldsymbol{D}$ be an algorithm that takes a string and returns a bit. Let $\boldsymbol{C}^*$ be an algorithm that takes a string, a list of strings and some state information, and returns a message or the symbol $\perp$, and a new state. We call $\boldsymbol{C}$ a *ciphertext-creator* adversary, $\boldsymbol{P}$ a *plaintext-creator* adversary, $\boldsymbol{D}$ a *distinguisher*, and $\boldsymbol{C}^*$ a *pa2-extractor*. For $k \in \mathbb{N}$, we define the experiments shown in Figure 3. It is required that, in these experiments, $\boldsymbol{C}$ not make a query (dec, C) for which $C \in \textsc{Clist}$. The *pa2-advantage* of $\boldsymbol{C}$ relative to $\boldsymbol{P}$, $\boldsymbol{D}$ and $\boldsymbol{C}^*$ is

$$\mathbf{Adv}^{\text{pa2}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P},\boldsymbol{D},\boldsymbol{C}^*}(k) \;=\; \Pr\left[\mathbf{Exp}^{\text{pa2-d}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P},\boldsymbol{D}}(k) = 1\right] - \Pr\left[\mathbf{Exp}^{\text{pa2-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P},\boldsymbol{D},\boldsymbol{C}^*}(k) = 1\right] .$$

We say that $\boldsymbol{C}^*$ is a *successful pa2-extractor for* $\boldsymbol{C}$ if for every polynomial-time plaintext creator $\boldsymbol{P}$ and distinguisher $\boldsymbol{D}$, the function $\mathbf{Adv}^{\text{pa2}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P},\boldsymbol{D},\boldsymbol{C}^*}(\cdot)$ is negligible. We say $\mathcal{AE}$ is *PA2 secure* if for any polynomial-time ciphertext creator there exists a successful polynomial-time pa2-extractor. ▮

In the definition of PA2, the core setting of PA1 is enhanced to model the real-life capability of a ciphertext creator to obtain ciphertexts via eavesdropping on communications made by a third party to the receiver (cf. [4]). Providing $\boldsymbol{C}$ with an encryption oracle does not capture this because eavesdropping puts into $\boldsymbol{C}$'s hands ciphertexts of which it does *not* know the corresponding plaintext, and, although we disallow $\boldsymbol{C}$ to query these to its oracle, it might be able to use them to create other ciphertexts whose corresponding plaintext it does not know and on which the extractor fails.

Experiment $\mathbf{Exp}^{\text{pa2-d}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P},\boldsymbol{D}}(k)$

$(pk, sk) \xleftarrow{\$} \mathcal{K}(1^k)$; $\text{CLIST} \leftarrow [\,]$
Choose coins $R[\boldsymbol{C}], R[\boldsymbol{P}]$ for $\boldsymbol{C}, \boldsymbol{P}$, respectively ; $\text{St}[\boldsymbol{P}] \leftarrow \varepsilon$
Run $\boldsymbol{C}$ on input pk and coins $R[\boldsymbol{C}]$ until it halts, replying to its oracle queries as follows:

– If $\boldsymbol{C}$ makes query (dec, Q) then
    $M \leftarrow \mathcal{D}(sk, Q)$; Return M to $\boldsymbol{C}$ as the reply EndIf
– If $\boldsymbol{C}$ makes query (enc, Q) then
    $(M, \text{St}[\boldsymbol{P}]) \leftarrow \boldsymbol{P}(Q, \text{St}[\boldsymbol{P}]; R[\boldsymbol{P}])$; $C \xleftarrow{\$} \mathcal{E}(pk, M)$; $\text{CLIST} \leftarrow \text{CLIST} @ C$
    Return C to $\boldsymbol{C}$ as the reply EndIf

Let x denote the output of $\boldsymbol{C}$; $d \xleftarrow{\$} \boldsymbol{D}(x)$; Return d

---

Experiment $\mathbf{Exp}^{\text{pa2-x}}_{\mathcal{AE},\boldsymbol{C},\boldsymbol{P},\boldsymbol{D},\boldsymbol{C}^*}(k)$

$(pk, sk) \xleftarrow{\$} \mathcal{K}(1^k)$; $\text{CLIST} \leftarrow [\,]$
Choose coins $R[\boldsymbol{C}], R[\boldsymbol{P}], R[\boldsymbol{C}^*]$ for $\boldsymbol{C}, \boldsymbol{P}, \boldsymbol{C}^*$, respectively
$\text{St}[\boldsymbol{P}] \leftarrow \varepsilon$; $\text{St}[\boldsymbol{C}^*] \leftarrow (pk, R[\boldsymbol{C}])$
Run $\boldsymbol{C}$ on input pk and coins $R[\boldsymbol{C}]$ until it halts, replying to its oracle queries as follows:

– If $\boldsymbol{C}$ makes query (dec, Q) then
    $(M, \text{St}[\boldsymbol{C}^*]) \leftarrow \boldsymbol{C}^*(Q, \text{CLIST}, \text{St}[\boldsymbol{C}^*]; R[\boldsymbol{C}^*])$
    Return M to $\boldsymbol{C}$ as the reply EndIf
– If $\boldsymbol{C}$ makes query (enc, Q) then
    $(M, \text{St}[\boldsymbol{P}]) \leftarrow \boldsymbol{P}(Q, \text{St}[\boldsymbol{P}]; R[\boldsymbol{P}])$; $C \xleftarrow{\$} \mathcal{E}(pk, M)$; $\text{CLIST} \leftarrow \text{CLIST} @ C$
    Return C to $\boldsymbol{C}$ as the reply EndIf

Let x denote the output of $\boldsymbol{C}$; $d \xleftarrow{\$} \boldsymbol{D}(x)$; Return d

Fig. 3. Experiments used to define PA2.

Modeling eavesdropping requires balancing two elements: providing $\boldsymbol{C}$ with a capability to obtain ciphertexts of plaintexts it does not know, yet capturing the fact that $\boldsymbol{C}$ might have partial information about the plaintexts, or control of the distribution from which these plaintexts are drawn. We introduce a companion *plaintext-creator* adversary $\boldsymbol{P}$ who, upon receiving a communication from $\boldsymbol{C}$, creates a plaintext and forwards it to an encryption oracle. The ciphertext emanating from the encryption oracle is sent to both $\boldsymbol{C}$ and $\boldsymbol{C}^*$. $\boldsymbol{C}$ has some control over $\boldsymbol{P}$ via its communication to $\boldsymbol{P}$, but we ensure this is not total by *denying* $\boldsymbol{C}$ *and* $\boldsymbol{C}^*$ *the coin tosses of* $\boldsymbol{P}$, and also by asking that $\boldsymbol{C}^*$ depend on $\boldsymbol{C}$ but not on $\boldsymbol{P}$.

The extractor $\boldsymbol{C}^*$ is, as before, provided with the coin tosses of $\boldsymbol{C}$. Two types of oracle queries are allowed to $\boldsymbol{C}$. Via a query (dec, Q), it can ask its oracle to decrypt ciphertext Q. Alternatively, it can make a query (enc, Q) to call $\boldsymbol{P}$ with argument Q, upon which the latter computes a message M and forwards it to the encryption oracle, which returns the resulting ciphertext to $\boldsymbol{C}$, and $\boldsymbol{C}^*$ in the case that $\boldsymbol{C}$'s oracle is $\boldsymbol{C}^*$. We observe that if an asymmetric encryption scheme is PA2 secure then it is PA1 secure, and if it is PA1 secure then it is PA0 secure.

See [2] for extensive comparisons of these definitions with previous ones, and also for stronger, statistical versions of these notions.

4 Relations among notions

We now state the formal results corresponding to Figure 1, beginning with the two motivating applications of our notions of plaintext awareness. Proofs of these results are provided in the full version of this paper [2].

Theorem 1. [**PA1+IND-CPA $\Rightarrow$ IND-CCA1**] *Let $\mathcal{AE}$ be an asymmetric encryption scheme. If $\mathcal{AE}$ is PA1 secure and IND-CPA secure, then it is IND-CCA1 secure.* ▌

Theorem 2. [**PA2+IND-CPA $\Rightarrow$ IND-CCA2**] *Let $\mathcal{AE}$ be an asymmetric encryption scheme. If $\mathcal{AE}$ is PA2 secure and IND-CPA secure, then it is IND-CCA2 secure.* ▌

Theorem 3. [**IND-CCA2 $\not\Rightarrow$ PA0+IND-CPA**] *Assume there exists an IND-CCA2-secure asymmetric encryption scheme. Then there exists an IND-CCA2-secure asymmetric encryption scheme that is not PA0 secure.* ▌

Theorem 4. [**PA1+IND-CPA $\not\Rightarrow$ IND-CCA2**] *Assume there exists a PA1 secure and IND-CPA-secure asymmetric encryption scheme. Then there exists a PA1 secure and IND-CPA-secure asymmetric encryption scheme that is not IND-CCA2 secure.* ▌

Theorem 5. [**PA0+IND-CPA $\not\Rightarrow$ IND-CCA1**] *Assume there exists a PA0 secure and IND-CPA-secure asymmetric encryption scheme. Then there exists a PA0 secure and IND-CPA-secure asymmetric encryption scheme that is not IND-CCA1 secure.* ▌

5 Constructions

PRIME-ORDER GROUPS. If p, q are primes such that $p = 2q + 1$, then we let G_q denote the subgroup of quadratic residues of $\mathbb{Z}_p^*$. Recall this is a cyclic subgroup of order q. If g is a generator of G_q then $\mathrm{dlog}_{q,g}(X)$ denotes the discrete logarithm of $X \in G_q$ to base g. A *prime-order-group generator* is a polynomial-time algorithm G that on input 1^k returns a triple (p, q, g) such that p, q are primes with $p = 2q + 1$, g is a generator of G_q, and $2^{k-1} < p < 2^k$ (p is k bits long).

THE DHK ASSUMPTIONS. Let G be a prime-order-group generator, and suppose $(p, q, g) \in [G(1^k)]$. We say that (A, B, W) is a *DH-triple* if there exist $a, b \in Z_q$ such that $A = g^a \bmod p$, $B = g^b \bmod p$ and $W = g^{ab} \bmod p$. We say that (B, W)

Experiment $\mathbf{Exp}^{\mathrm{dhk1}}_{G,\boldsymbol{H},\boldsymbol{H}^*}(k)$

$(p,q,g) \stackrel{\$}{\leftarrow} G(1^k)$; $a \stackrel{\$}{\leftarrow} \mathbb{Z}_q$; $A \leftarrow g^a \bmod p$

Choose coins $R[\boldsymbol{H}], R[\boldsymbol{H}^*]$ for $\boldsymbol{H}, \boldsymbol{H}^*$, respectively ; $\mathrm{St}[\boldsymbol{H}^*] \leftarrow ((p,q,g,A), R[\boldsymbol{H}])$

Run $\boldsymbol{H}$ on input p,q,g,A and coins $R[\boldsymbol{H}]$ until it halts, replying to its oracle queries as follows:

– If $\boldsymbol{H}$ makes query (B,W) then

  $(b, \mathrm{St}[\boldsymbol{H}^*]) \leftarrow \boldsymbol{H}^*((B,W), \mathrm{St}[\boldsymbol{H}^*]; R[\boldsymbol{H}^*])$

  If $W \equiv B^a \pmod p$ and $B \not\equiv g^b \pmod p$ then return 1

  Else return b to $\boldsymbol{H}$ as the reply EndIf EndIf

Return 0

Fig. 4. Experiment used to define the DHK1 and DHK0 assumptions.

is a *DH-pair relative to* A if (A,B,W) is a DH-triple. One way for an adversary $\boldsymbol{H}$ taking input p,q,g,A to output a DH-pair (B,W) relative to A is to pick —and thus "know"— some $b \in \mathbb{Z}_q$, set $B = g^b \bmod p$ and $W = A^b \bmod p$, and output (B,W). Damgård [12] makes an assumption which, informally, says that this is the "only" way that a polynomial-time adversary $\boldsymbol{H}$ can output a DH-pair relative to A. His framework to capture this requires that there exist a suitable extractor $\boldsymbol{H}^*$ that can compute $\mathrm{dlog}_{q,g}(B)$ whenever $\boldsymbol{H}$ outputs some DH-pair (B,W) relative to A.

We provide a formalization of this assumption that we refer to as the DHK0 (DHK stands for Diffie-Hellman Knowledge) assumption. We also present a natural extension of this assumption that we refer to as DHK1. Here the adversary $\boldsymbol{H}$, given p,q,g,A, interacts with the extractor, querying it adaptively. The extractor is required to be able to return $\mathrm{dlog}_{q,g}(B)$ for each DH-pair (B,W) relative to A that is queried to it. Below we first present the DHK1 assumption, and then define the DHK0 assumption via this.

Assumption 1. **[DHK1]** Let G be a prime-order-group generator. Let $\boldsymbol{H}$ be an algorithm that has access to an oracle, takes two primes and two group elements, and returns nothing. Let $\boldsymbol{H}^*$ be an algorithm that takes a pair of group elements and some state information, and returns an exponent and a new state. We call $\boldsymbol{H}$ a dhk1-*adversary* and $\boldsymbol{H}^*$ a dhk1-*extractor*. For $k \in \mathbb{N}$ we define the experiment shown in Figure 4. The dhk1-*advantage* of $\boldsymbol{H}$ relative to $\boldsymbol{H}^*$ is

$$\mathbf{Adv}^{\mathrm{dhk1}}_{G,\boldsymbol{H},\boldsymbol{H}^*}(k) \;=\; \Pr\left[\,\mathbf{Exp}^{\mathrm{dhk1}}_{G,\boldsymbol{H},\boldsymbol{H}^*}(k) = 1\,\right] .$$

We say that G satisfies the DHK1 assumption if for every polynomial-time dhk1-adversary $\boldsymbol{H}$ there exists a polynomial-time dhk1-extractor $\boldsymbol{H}^*$ such that $\mathbf{Adv}^{\mathrm{dhk1}}_{G,\boldsymbol{H},\boldsymbol{H}^*}(\cdot)$ is negligible. ▮

Assumption 2. **[DHK0]** Let G be a prime-order-group generator. We call a dhk1-adversary that makes *exactly one* oracle query a dhk0-*adversary.* We call a dhk1-extractor for a dhk0-adversary a dhk0-*extractor*. We say that G satisfies the Diffie-Hellman Knowledge (DHK0) assumption if for every polynomial-time

Algorithm $\mathcal{K}(1^k)$ $(p,q,g) \xleftarrow{\$} G(1^k)$ $x_1 \xleftarrow{\$} \mathbb{Z}_q$; $X_1 \leftarrow g^{x_1} \bmod p$ $x_2 \xleftarrow{\$} \mathbb{Z}_q$; $X_2 \leftarrow g^{x_2} \bmod p$ Return $((p,q,g,X_1,X_2),(p,q,g,x_1,x_2))$	Algorithm $\mathcal{E}((p,q,g,X_1,X_2),M)$ $y \xleftarrow{\$} \mathbb{Z}_q$; $Y \leftarrow g^y \bmod p$ $W \leftarrow X_1^y \bmod p$; $V \leftarrow X_2^y \bmod p$ $U \leftarrow V \cdot M \bmod p$ Return (Y,W,U)
Algorithm $\mathcal{D}((p,q,g,x_1,x_2),(Y,W,U))$ If $W \not\equiv Y^{x_1} \pmod p$ then return $\perp$ Else $M \leftarrow U \cdot Y^{-x_2} \bmod p$; Return M EndIf	$\mathrm{MsgSp}((p,q,g,X_1,X_2)) = G_q$

Fig. 5. Algorithms of the encryption scheme DEG $= (\mathcal{K},\mathcal{E},\mathcal{D},\mathrm{MsgSp})$ based on prime-order-group generator G.

dhk0-adversary $\boldsymbol{H}$ there exists a polynomial-time dhk0-extractor $\boldsymbol{H}^*$ such that $\mathbf{Adv}^{\mathrm{dhk1}}_{G,\boldsymbol{H},\boldsymbol{H}^*}(\cdot)$ is negligible. ▮

We observe that DHK1 implies DHK0 in the sense that if a prime-order-group generator satisfies the former assumption then it also satisfies the latter assumption.

Constructions. We would like to build an asymmetric encryption scheme that is PA0 secure (and IND-CPA secure) under the DHK0 assumption. An obvious idea is to use ElGamal encryption. Here the public key is $X = g^x$, where x is the secret key, and an encryption of message $M \in G_q$ has the form (Y,U), where $Y = g^y \bmod p$ and $U = X^y \cdot M \bmod p = g^{xy} \cdot M \bmod p$. However, we do not know whether this scheme is PA0 secure.

We consider a modification of the ElGamal scheme that was proposed by Damgård [12]. We call this scheme *Damgård ElGamal* or DEG. It is parameterized by a prime-order group generator G, and its components are depicted in Figure 5. The proof of the following is in the full version of this paper [2]:

Theorem 6. *Let G be a prime-order-group generator and let* DEG $= (\mathcal{K},\mathcal{E},\mathcal{D},\mathrm{MsgSp})$ *be the associated Damgård ElGamal asymmetric encryption scheme defined in Figure 5. If G satisfies the DHK0 and DDH assumptions then* DEG *is PA0+IND-CPA secure. If G satisfies the DHK1 and DDH assumptions then* DEG *is PA1+IND-CPA secure.* ▮

As a consequence of the above and Theorem 1, DEG is IND-CCA1 secure under the DHK1 and DDH assumptions. DEG is in fact the most efficient known IND-CCA1 scheme with some proof of security in the standard model.

Next we consider the "lite" version of the Cramer-Shoup asymmetric encryption scheme [11]. The scheme, denoted CS-lite, is parameterized by a prime-order group generator G, and its components are depicted in Figure 6. This scheme is known to be IND-CCA1 secure under the DDH assumption [11]. We are able to show the following. The proof can be found in [2].

Algorithm $\mathcal{K}(1^k)$	Algorithm $\mathcal{E}((p,q,g_1,g_2,X,Z),M)$
$(p,q,g_1) \xleftarrow{\$} G(1^k)$; $g_2 \xleftarrow{\$} G_q \setminus \{1\}$ $x_1 \xleftarrow{\$} \mathbb{Z}_q$; $x_2 \xleftarrow{\$} \mathbb{Z}_q$; $z \xleftarrow{\$} \mathbb{Z}_q$ $X \leftarrow g_1^{x_1} \cdot g_2^{x_2} \bmod p$; $Z \leftarrow g_1^z \bmod p$ Return $((p,q,g_1,g_2,X,Z),(p,q,g_1,g_2,x_1,x_2,z))$	$r \xleftarrow{\$} \mathbb{Z}_q$ $R_1 \leftarrow g_1^r \bmod p$ $R_2 \leftarrow g_2^r \bmod p$ $E \leftarrow Z^r \cdot M \bmod p$ $V \leftarrow X^r \bmod p$ Return (R_1,R_2,E,V)
Algorithm $\mathcal{D}((p,q,g_1,g_2,x_1,x_2,z),(R_1,R_2,E,V))$ If $V \not\equiv R_1^{x_1} \cdot R_2^{x_2} \pmod p$ then return $\perp$ Else $M \leftarrow E \cdot R_1^{-z} \bmod p$; Return M EndIf	$\mathrm{MsgSp}((p,q,g_1,g_2,X,Z)) = G_q$

Fig. 6. Algorithms of the encryption scheme CS-lite $= (\mathcal{K},\mathcal{E},\mathcal{D},\mathrm{MsgSp})$ based on prime-order-group generator G.

Theorem 7. *Let G be a prime-order-group generator, and let* CS-lite $= (\mathcal{K},\mathcal{E},\mathcal{D},\mathrm{MsgSp})$ *be the associated Cramer-Shoup lite asymmetric encryption scheme defined in Figure 6. If G satisfies the DHK0 and DDH assumptions then* CS-lite *is PA0+IND-CPA secure. If G satisfies the DHK1 and DDH assumptions then* CS-lite *is PA1+IND-CPA secure.* ▮

Again, the above and Theorem 1 imply that CS-lite is IND-CCA1 secure under the DHK1 and DDH assumptions. This however is not news, since we already know that DDH alone suffices to prove it IND-CCA1 [11]. However, it does perhaps provide a new perspective on why the scheme is IND-CCA1, namely that this is due to its possessing some form of plaintext awareness.

In summary, we have been able to show that plaintext awareness without ROs is efficiently achievable, even though under very strong and non-standard assumptions.

References

1. M. Backes, B. Pfitzmann and M. Waidner. A composable cryptographic library with nested operations. CCS 03.
2. M. Bellare and A. Palacio. Towards plaintext-aware public-key encryption without random oracles. Full version of this extended abstract. Available at `http://www-cse.ucsd.edu/users/mihir`.
3. M. Bellare, A. Boldyreva and A. Palacio. An un-instantiable random oracle model scheme for a hybrid encryption problem. EUROCRYPT '04.
4. M. Bellare, A. Desai, D. Pointcheval and P. Rogaway. Relations among notions of security for public-key encryption schemes. CRYPTO '98.
5. M. Bellare and A. Palacio. The knowledge-of-exponent assumptions and 3-round zero-knowledge protocols. CRYPTO '04.
6. M. Bellare and P. Rogaway. Optimal asymmetric encryption. EUROCRYPT '94.

7. D. Boneh. Simplified OAEP for the RSA and Rabin functions. CRYPTO '01.
8. M. Blum, P. Feldman and S. Micali. Non-interactive zero-knowledge and its applications. STOC 88.
9. M. Blum, P. Feldman and S. Micali. Proving security against chosen ciphertext attacks. CRYPTO '88.
10. R. Canetti, O. Goldreich and S. Halevi. The random oracle methodology, revisited. STOC 98.
11. R. Cramer and V. Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM Journal on Computing*, Vol. 33, No. 1, 2003, pp. 167–226.
12. I. Damgård. Towards practical public key systems secure against chosen ciphertext attacks. CRYPTO '91.
13. A. De Santis and G. Persiano. Zero-knowledge proofs of knowledge without interaction. FOCS 92.
14. D. Dolev and A. Yao. On the security of public-key protocols. *IEEE Transactions on Information Theory*, Vol. 29, 1983, pp. 198–208.
15. D. Dolev, C. Dwork, and M. Naor. Non-Malleable cryptography. *SIAM Journal on Computing*, Vol. 30, No. 2, 2000, pp. 391–437.
16. E. Fujisaki, T. Okamoto, D. Pointcheval and J. Stern. RSA-OAEP is secure under the RSA assumption. CRYPTO '01.
17. O. Goldreich. A uniform-complexity treatment of encryption and zero-knowledge. *Journal of Cryptology*, Vol. 6, No. 1, 1993, pp. 21–53.
18. S. Goldwasser and Y. Taumann. On the (in)security of the Fiat-Shamir paradigm. FOCS 03.
19. S. Hada and T. Tanaka. On the existence of 3-round zero-knowledge protocols. IACR Cryptology ePrint Archive, Report 1999/009, March 1999. Available at `http://eprint.iacr.org/1999/009/`. [Revised version of [20].]
20. S. Hada and T. Tanaka. On the existence of 3-round zero-knowledge protocols. CRYPTO '98. [Preliminary version of [19].]
21. J. Herzog, M. Liskov and S. Micali. Plaintext awareness via key registration. CRYPTO '03.
22. S. Goldwasser and S. Micali. Probabilistic Encryption. *Journal of Computer and System Science*, Vol. 28, 1984, pp. 270–299.
23. S. Micali, C. Rackoff, and B. Sloan. The notion of security for probabilistic cryptosystems. *SIAM Journal on Computing*, Vol. 17, No. 2, 1988, pp. 412–426.
24. M. Naor. Cryptographic assumptions and challenges. CRYPTO '03.
25. M. Naor and M. Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. STOC 90.
26. J. B. Nielsen. Separating Random Oracle Proofs from Complexity Theoretic Proofs: The Non-committing Encryption Case. CRYPTO '02
27. C. Rackoff and D. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. CRYPTO '91.
28. V. Shoup. OAEP reconsidered. *Journal of Cryptology* Vol. 15, No. 4, 2002, pp. 223–249.