Full version of the paper to appear at *ProvSec 2009*, 3rd International Conference on Provable Security, Guangzhou, China, November 11-13, 2009.

Chosen-Ciphertext Secure RSA-type Cryptosystems

Benoît Chevallier-Mames[1] and Marc Joye[2]

[1] DCSSI, Laboratoire Crypto
51 Boulevard de la Tour Maubourg, 75700 Paris, France
`benoit.chevallier-mames@sgdn.gouv.fr`
[2] Thomson R&D, Security Competence Center
1 avenue de Belle Fontaine, 35576 Cesson-Sévigné Cedex, France
`marc.joye@thomson.net` – `http://joye.site88.net/`

Abstract. This paper explains how to design fully secure RSA-type cryptosystems from schemes only secure against passive attacks, in the standard model. We rely on instance-independence assumptions, which, roughly speaking, conjecture that for certain problems, an interactive access to a solver for another problem does not help the challenger. Previously, instance-independence assumptions were used in a "negative" way, to prove that certain schemes proven in the random oracle model were not provable in the standard model.
Our paradigm applies virtually to all (weakly secure) RSA-type encryption schemes for which public-key RSA exponent can be arbitrarily chosen. As an illustration, we present a chosen-ciphertext secure variant of the Naccache-Stern encryption scheme.

Keywords: Chosen-ciphertext security, public-key encryption, standard model, RSA-based encryption schemes, instance-independence assumptions, one-time mappable chameleon hashing.

1 Introduction

1.1 Chosen-ciphertext security

Building on the seminal work of Goldwasser and Micali [24], the commonly accepted security notion for public-key encryption is that of *indistinguishability under adaptive chosen-ciphertext attacks* (IND-CCA2, or IND-CCA for short) [40] (see also [33] for a weaker notion considering non-adaptive adversaries). To reach this security notion, there has been an extensive body of work on public-key encryption, a recent survey of which can be can be found in [17].

Two-key paradigm. The first attempt to get security against chosen-ciphertext attacks was given by Naor and Yung [33]. This paradigm consists in providing two independent encryptions of the same plaintext together with a non-interactive zero-knowledge (NIZK) proof that the two plaintexts are the same. Unfortunately, this technique suffers from a certain inefficiency.

Random oracles. To overcome the inefficiency of first techniques to achieve IND-CCA-security, Bellare and Rogaway formalized the notion of the random oracle methodology [4]. The random oracle model assumes that a hash function behaves as a random function. This assumption allows one to design very efficient schemes (e.g., [5]). Indeed, in the random oracle model, security proofs are much simpler to design, since the random oracle gives an extra power to the reduction algorithm. Generic transformations to turn a weakly secure encryption scheme into a fully secure encryption scheme are also known (e.g., [20]).

Smooth hash proof systems. It is hard to write a paper on chosen-ciphertext security without citing [15], where Cramer and Shoup proposed the first provably (efficient) secure scheme in the standard model. They later generalized the underlying technique with the notion of *smooth hash-proof systems* [16], which are very efficient non-interactive zero-knowledge proof for certain languages. The current state-of-the-art scheme is due to Kurosawa and Desmedt [28].

Identity-based encryption. More recently, in [10], Canetti, Halevi and Katz proposed a different way to get chosen-ciphertext security (see also [7, 6, 2]) . They observe how to turn an IND-ID-CPA identity-based encryption (IBE) scheme into an IND-CCA public-key encryption scheme, by using a strongly unforgeable one-time signature scheme. The construction in [7] considers the same paradigm but uses a MAC rather than a one-time signature, leading to some efficiency improvements. Further efficiency improvements are achieved in [2] by considering *partitioned* IBE schemes.

1.2 Instance-independence assumptions

In [35, 34], Paillier and Villar considered several assumptions in order to show that several RSA-based schemes were unprovable (under a black-box reduction) in the standard model while provably secure in the random oracle model. They refer to these assumptions as *non-malleability assumptions*, but to avoid any confusion with the classical notion of non-malleability of an encryption scheme (as proposed by Dolev, Dwork, and Naor [18]), we use the term of *instance-independence assumptions* throughout this paper.

Let $\mathcal{P}$ be a cryptographic problem. An access (possibly with certain restrictions on entries) to an oracle solving a problem $\mathcal{Q}$ may be a way to find more easily a solution to problem $\mathcal{P}$. More precisely, the oracle solving $\mathcal{Q}$ is an *interactive* resource that is given to the adversary. For a pair $(\mathcal{P}, \mathcal{Q})$, the problem consisting in solving $\mathcal{P}$ with an access to a $\mathcal{Q}$-solver oracle is called the $(\mathcal{P}, \mathcal{Q})$ *assisted problem.* Even if the problem $\mathcal{Q}$ is expected to be computationally hard (and thus, even if the $\mathcal{Q}$-solver oracle gives a lot of extra power to the attacker), the $(\mathcal{P}, \mathcal{Q})$ assisted problem may remain hard. These are the cases that are considered in [35, 34] (see also [23, 29] for non-adaptive versions) and in this paper: The assumption that solving the $(\mathcal{P}, \mathcal{Q})$ assisted problem is only negligibly easier than solving the $\mathcal{P}$ problem is called the $(\mathcal{P}, \mathcal{Q})$ *instance-independence assumption.* See also [1].

The first example given in [35, 34] is the *assisted factorization*: with an oracle solving factorization of $n \neq n_0$ (with the restriction that $n \neq 0 \bmod n_0$ for obvious reasons), it should remain hard to factor modulus n_0. Another example is the *assisted e-th root computation*: with an oracle returning e-th roots modulo n (for $(e, n) \neq (e_0, n_0)$) — where (e, n) is a possible output of some RSA key generator[3]), it should remain hard to compute an e_0-th root modulo n_0.

Remark 1. We stress that instance-independence assumptions should not be confused with *one-more assumptions.* In a one-more-$\mathcal{P}$ problem, one is given access to two oracles, $\mathcal{O}_{\mathsf{Gen}}$ and $\mathcal{O}_{\mathsf{Solve}}$, and needs to solve $(n+1)$ independent instances of the $\mathcal{P}$ challenge (these challenges being given by $\mathcal{O}_{\mathsf{Gen}}$), using at most n accesses to $\mathcal{O}_{\mathsf{Solve}}$, which is an oracle that solves any chosen instance of problem $\mathcal{P}$. Examples of one-more problems include the one-more RSA, the one-more DL and the one-more CDH [3].

The assisted $(\mathcal{P}, \mathcal{Q})$-problems, which are supposedly as hard as their primitive problem $\mathcal{P}$ under instance-independence assumptions, are of different nature. Indeed, they consist in solving only *one* instance of the $\mathcal{P}$-problem, using a possibly very large number of queries to an oracle solving the $\mathcal{Q}$-problem. What makes these assisted problems non-trivial is the fact that, in contrast with one-more problems, the entries of the $\mathcal{Q}$-problem solver are supposed to be "not too related" to the $\mathcal{P}$-challenge (hence the name of *instance-independence*). As a consequence, assisted problems and one-more problems can hardly been linked or compared together.

1.3 Our contributions

Instance-independence assumptions were used in [35, 34] to prove that security proofs of several classical cryptosystems proven in the random oracle were impossible to achieve in the standard model, under a black-box reduction. On the contrary, this paper considers positive applications of instance-independence assumptions. Namely, we consider some simple and known IND-CPA primitives with certain properties (we refer to these primitives as *public-key encryptions with ephemeral key*), and show how to transform them into fully secure encryption schemes. We provide a proof that, under clearly defined instance-independence assumptions, the so-obtained schemes are as secure as their underlying primitives, in the standard model. Interestingly, RSA primitives we consider in this paper corresponds — under some instance-independence assumptions — to cryptographic objects named as *adaptive one-way functions* in [36]. Our primitives are more efficient than the general constructions given in [36] but, being specific, they rely on stronger assumptions.

For the sake of illustration, we will present an application of our paradigm from the knapsack encryption scheme of Naccache and Stern [32]. Further applications of our paradigm can be found in Appendix A: one is based on the

[3] For example, it should be supposed that n is not a multiple of n_0, or that e does not share common factors with e_0. Therefore, to hope to achieve the assumption, one may assume that e is prime, and that n has a certain fixed-length ℓ_n.

dependent-RSA problem [38] and another one is based on the decisional small e-th residues problem [11].

1.4 Relation to IBE technique

Our technique shares many common features with the identity-based construction in [6]. Very roughly, the idea behind the identity-based encryption (IBE) paradigm can be sum up as follows. For each new encryption, the user generates a fresh key pair $(\mathsf{pk}, \mathsf{sk})$ for a certain signature scheme, and encrypts a message by using an IND-ID-CPA secure IBE scheme (the result being noted c), using the public key pk as the identity in the IBE scheme. The resulting ciphertext is then made of pk, c, appended with a signature of c under private key sk. The IND-CCA security of the so-obtained encryption scheme is achieved since access to the decryption oracle is useless for an attacker: indeed, there is no way for this latter to gain information about the challenge from the oracle since it does not know the secret key sk_*, and thus, cannot create ciphertexts corresponding to identity pk_* of the challenge. The IND-ID-CPA security of the underlying IBE scheme suffices to conclude the proof.[4]

Our scheme uses in a way the same kind of strategy. Imagine one starts from an IND-CPA RSA-type encryption primitive, where additionally the user can decide the value of the public exponent e for each ciphertext (the public exponent being thus *ephemeral*). If the user proves that it has generated e by himself (by exhibiting a proof of knowledge on something closely related to e), such an exponent would not be anymore reusable by an attacker, i.e., an attacker would not be able to submit to a decryption oracle a valid ciphertext with the ephemeral public exponent e_* appearing in the challenge. In this way, we forbid the attacker to learn information from the decryption oracle in a way similar to the IBE setting: the signature corresponding to public key pk makes that no valid ciphertext with same identity pk_* as in the challenge can be queried.

Unfortunately, there is an additional important issue to address in our setting: we have to provide the attacker with the decryption queries.. In the IBE setting, decryption queries are answered using the IND-ID-CPA security of the underlying identity-based encryption. In our context, the underlying IND-CPA RSA-type encryption primitive is insufficient. This is where instance-independence assumptions come into play in order to assume that our reduction has an access to an oracle answering decryption queries.

One may finally remark that the [6] transformation has been modified by Kiltz in [26], who proposed an efficient conversion from a tag-based encryption to form a chosen-ciphertext secure encryption scheme. This transformation is more general than the IBE based one since any identity-based encryption scheme can be turned into a tag-encryption scheme.

[4] We refer the reader to [10] for a more precise security proof.

1.5 Outline of the paper

The rest of this paper is organized as follows. Section 2 reviews standard definitions and security notions for public-key encryption, as well as the paradigm of chameleon hashing. Section 3 introduces new cryptographic problems, and defines instance-independence assumptions. Section 4 describes our main construction, including a complete chosen-ciphertext secure scheme. Finally, Section 5 concludes the paper.

2 Preliminaries

In this section, we introduce some background on public-key encryption. We also review some ingredients that will be used in our constructions.

2.1 Public-key encryption

A *public-key encryption scheme*, $\mathcal{E} = (\textsc{Gen}, \textsc{Enc}, \textsc{Dec})$, can be described as a tuple of probabilistic polynomial-time algorithms. By default, the message space is $\{0,1\}^*$.

Key Generation. Given a security parameter κ, $\textsc{Gen}(1^\kappa)$ produces a pair $(\mathsf{pk}, \mathsf{sk})$ of matching public and private keys.

Encryption. Given a message m in message space $\mathcal{M}$ and public key pk, $\textsc{Enc}_{\mathsf{pk}}(m)$ produces a ciphertext $c \leftarrow \textsc{Enc}_{\mathsf{pk}}(m)$. If x denotes the random coins used by $\textsc{Enc}$, we equivalently write $c = \textsc{Enc}_{\mathsf{pk}}(m, x)$.

Decryption. Given a ciphertext c and private key sk, $\textsc{Dec}_{\mathsf{sk}}(c)$ returns a plaintext m or a special symbol $\perp$ denoting that the ciphertext is invalid.

We require that if $c \leftarrow \textsc{Enc}_{\mathsf{pk}}(m)$, then $\textsc{Dec}_{\mathsf{sk}}(c)$ returns m for all $(\mathsf{pk}, \mathsf{sk}) \leftarrow \textsc{Gen}(1^\kappa)$ and messages drawn in the message space.

2.2 Security notions for encryption schemes

Semantic security. The notion of *semantic security* (IND) [24], also known as *indistinguishability of encryptions*, captures a strong notion of privacy: The attacker should not learn any information whatsoever about a plaintext given its encryption. The adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ is said to $(\kappa, \varepsilon, \tau)$-break IND when

$$\mathsf{Adv}_{\mathcal{E}}^{\mathsf{IND}}(\mathcal{A}) = 2 \times \Pr_{b,x}\left[\begin{matrix}(\mathsf{pk},\mathsf{sk}) \leftarrow \textsc{Gen}(1^\kappa), (m_0, m_1, s) \leftarrow \mathcal{A}_1(\mathsf{pk}), \\ c \leftarrow \textsc{Enc}_{\mathsf{pk}}(m_b) \ : \ \mathcal{A}_2(m_0, m_1, s, c) = b\end{matrix}\right] - 1 \geq \varepsilon \ , \quad (1)$$

where the probability is taken over the random coins of the experiment according to the distribution induced by $\textsc{Gen}(1^\kappa)$ as well as the ones the adversary, where $b \in \{0,1\}$ and $m_0, m_1 \in \mathcal{M}$. $\mathcal{A}$ must run in at most τ steps and it is imposed that $|m_0| = |m_1|$. An encryption scheme is said to be *semantically secure* (or

IND secure) if no probabilistic algorithm can $(\kappa, \varepsilon, \tau)$-break IND for $\tau \leq \mathsf{poly}(\kappa)$ and $\varepsilon \geq 1/\mathsf{poly}(\kappa)$.

As we are in the public-key setting, it is worth noting that adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ is given the public-key pk and so can encrypt any message of its choice. In other words, the adversary can mount chosen-plaintext attacks (CPA). Hence, we write $\mathsf{IND\text{-}CPA}$ the security level offered by an IND-secure encryption scheme, emphasizing the fact that $\mathcal{A}$ has access to an encryption oracle (for free).

Chosen-ciphertext attacks. $\mathsf{IND\text{-}CPA}$ security offers an adequate security level in the presence of a *passive* adversary. In a number of situations however (e.g., see [41]), this may reveal insufficient in the presence of a more powerful adversary that can do more than merely eavesdropping the exchanged messages.

The "right" security level against *active* attacks is that of $\mathsf{IND\text{-}CCA}$ security, or *security against chosen-ciphertext attacks* [40, 18] (see also [33]). The definition of the adversary's advantage as given by Eq. (1) readily extends to the $\mathsf{IND\text{-}CCA}$ model but the adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ now is given an adaptive access to a decryption oracle to which it can submit any ciphertext of its choice with the exception that $\mathcal{A}_2$ may not query the decryption oracle on challenge ciphertext c.

2.3 Chameleon hashing

Chameleon hash functions [27] (see also [8]) are hash functions associated with a pair $(\mathsf{hk}, \mathsf{tk})$ of hashing/trapdoor keys. The name chameleon refers to the ability of the owner of the trapdoor key to modify the input without changing the output.

A chameleon hash function is defined by a triple of probabilistic polynomial-time algorithms $(\textsc{ChamGen}, \textsc{ChamHash}, \textsc{ChamColl})$. $\textsc{ChamGen}$, on input 1^κ, produces a pair of hashing/trapdoor keys: $(\mathsf{hk}, \mathsf{tk}) \leftarrow \textsc{ChamGen}(1^\kappa)$. Given a message m and a random string r, chameleon hash algorithm with hashing key hk

$$\textsc{ChamHash}_{\mathsf{hk}} : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*, (m, r) \mapsto h = \textsc{ChamHash}_{\mathsf{hk}}(m, r)$$

generates a hash value $\textsc{ChamHash}_{\mathsf{hk}}(m, r)$. Moreover, a chameleon hash function should satisfy the properties that

(i) there exists no probabilistic polynomial-time algorithm that, on input random hashing key hk, can find two pairs of message/random string, (m_1, r_1) and (m_2, r_2) with $m_1 \neq m_2$, such that the relation $\textsc{ChamHash}_{\mathsf{hk}}(m_1, r_1) = \textsc{ChamHash}_{\mathsf{hk}}(m_2, r_2)$ is satisfied, except with negligible probability;

(ii) on input trapdoor key tk, given (m_1, r_1, m_2), algorithm $\textsc{ChamColl}$ can find a string $r_2 = \textsc{ChamColl}_{\mathsf{tk}}(m_1, r_1, m_2)$ such that $\textsc{ChamHash}_{\mathsf{hk}}(m_1, r_1) = \textsc{ChamHash}_{\mathsf{hk}}(m_2, r_2)$.

(iii) all messages m induce (computationally) indistinguishable probability distributions on chameleon hash function $\textsc{ChamHash}_{\mathsf{hk}}(m, r)$ for r chosen uniformly at random.

Example 1. Consider the group $\mathbb{G} = \langle g \rangle$ generated by an element $g \in \mathbb{Z}_n^*$, where n is an RSA modulus. Suppose that $\mathcal{H} : \mathbb{G} \to \{0,1\}^{\ell_h}$ is a second-preimage resistant hash function. It is easy to see that function $\mathcal{F}_v$ given by

$$\mathcal{F}_v : \mathbb{Z}_{|\mathbb{G}|} \times \mathbb{Z}_{|\mathbb{G}|} \to \{0,1\}^{\ell_h}, \ \ (m,r) \mapsto \mathcal{F}_v(m,r) := \mathcal{H}(g^r\, v^m \bmod n)$$

is a chameleon hash function where hashing key is $\mathsf{hk} = \{n, g, v\}$ while trapdoor key is $\mathsf{tk} = \{y\}$, with $g^y \equiv v \pmod{n}$. Given a triple (m_1, r_1, m_2), a colliding value can be found using trapdoor key y as $r_2 = r_1 + y(m_1 - m_2) \bmod |\mathbb{G}|$.

Example 2. Suppose now that the order of $\mathbb{G}$ is unknown but that its order of magnitude is known: $|\mathbb{G}| \approx 2^{\ell_g}$. We then restrict the range of messages in $\mathcal{F}_v$ to $\{0,1\}^{\ell_c}$ and allow for larger random strings $r \in \{0,1\}^{\ell_g+\ell_c+2\kappa}$ (where κ is some security parameter); trapdoor key y is chosen in $\{0,1\}^{\ell_g+\kappa}$.[5] Since $|\mathbb{G}|$ is unknown, trapdoor key y cannot be recovered as $y = (r_2 - r_1)/(m_1 - m_2)$, as in Example 1. Two different colliding pairs (m_1, r_1) and (m_2, r_2) will only yield a multiple of $|\mathbb{G}|$:

$$L := (r_1 - r_2) + y(m_1 - m_2) \propto |\mathbb{G}| \ .$$

For random strings $r_1 \leftarrow \{0,1\}^{\ell_g+\ell_c+2\kappa}$, the statistical zero-knowledge property guarantees that L is non-zero with overwhelming probability (see e.g. [39]). Suppose further that n is a *strong* RSA modulus (i.e., $n = (2p'+1)(2q'+1)$ for some primes p' and q') and that $\mathbb{G}$ is the set of quadratic residues modulo n. In this case, the value of $4L$ is a multiple of $\phi(n)$, which, using Miller's algorithm [31], leads to the factorization of n.

3 Complexity Assumptions

3.1 Decision RSA short encrypted-prime problem

We start with the definition.

Definition 1 (Decision RSA Short Encrypted-Prime Problem (D-RSA-SEP)). *Let (ℓ_e, ℓ_n, ℓ_k) be security parameters. The challenger is given a triple (e_0, n_0, z_0), where n_0 is an ℓ_n-bit RSA modulus, e_0 is an ℓ_e-bit random prime, and $z_0 \leftarrow \mathbb{Z}_{n_0}^*$. Its goal is to decide whether or not z_0 is of the form $k^{e_0} \bmod n_0$, for some $k \in \textsc{Primes}(\{2^{\ell_k-1}, 2^{\ell_k}\})$.*

We have no proof that D-RSA-SEP problem is hard. Yet, we try to list some key points which are needed to assess its difficulty. We also relate it to other classical and close problems that appeared in the literature.

First, parameter ℓ_k is important, as it codes for the number of possible primes: there are around $2^{\ell_k}/\ell_k$ primes of ℓ_k bits. Thus, we want this latter quantity to be larger than the expected security limit of 2^κ, where κ is the security parameter. For example, for a 112-bit security level, one needs $\ell_k \geq 119$.

[5] Such a setting is sufficient to ensure that the discrete logarithm of v in base g (i.e., $y \bmod |\mathbb{G}|$) is statistically indistinguishable from a random integer modulo $|\mathbb{G}|$.

Second, exponent e cannot be too small. Indeed, if e is too small, i.e., if $e \cdot \ell_k < \ell_n$, the value $k^e \bmod n$ is actually equal to k^e over the integers. Therefore, it is easy for the challenger to solve the D-RSA-SEP problem by computing e-th roots over the integers. The same holds true when $e \cdot \ell_k \approx \ell_n$ as the challenger can then first guess the quotient $\frac{k^e}{n}$. Hence, typically, $\ell_k = 119$ and $e = 3$ is a bad setting. All in all, it is necessary for the problem to be hard that $e \cdot \ell_k \gg \ell_n$. Note that this is also imposed by Coppersmith's algorithm [13].

It is very simple to see that D-RSA-SEP $\Leftarrow$ RSA $\Leftarrow$ FACT, where $\Leftarrow$ denotes polynomial reductions. Apart from these easy reductions, we do not see any way to prove the relative security of the new problem we present. This may therefore be the subject of future research.

RSA-type problems. We review below in this section some problems that were proposed in the literature, that are more or less related to D-RSA-SEP.

*Relationship with dependent-*RSA *problem.* The dependent-RSA problem (DRSA) was introduced by Pointcheval in [38]: it was one of the first decisional RSA-type problems proposed in the literature. The idea consists in trying to mount an RSA equivalent of the ElGamal pair (g^r, y^r). Notably, one of the proposed mode is the pair $(r^e \bmod n, (r+1)^e \bmod n)$. The Dependent-RSA problem, in its computational version, is defined as computing $(r+1)^e \bmod n$ from $(e, n, r^e \bmod n)$, while its decisional version consists in distinguishing $(r+1)^e \bmod n$ from a random element in $\mathbb{Z}_n^*$, being given $(e, n, r^e \bmod n)$.

Essentially, Dependent-RSA and D-RSA-SEP problems feature in common that their hardness requires e to be large enough.

Relationship with decisional small e-th residues problem. In [11], Catalano, Gennaro, Howgrave-Graham and Nguyen proposed the (computational/decisional) small e-th roots problem (DSeR being the notation of the decisional version). It arises from a variant of Paillier scheme they proposed.

The decisional small e-roots problem is by definition to decide whether an input is of the form $r^e \bmod n^2$ (with $r \in \mathbb{Z}_n^*$) or not, being given (e, n). This problem is close to D-RSA-SEP in the sense that it basically requires to distinguish small values raised to a power e (a small prime in D-RSA-SEP, an half-size integer in DSeR) from random values.

Relationship with Micali-Schnorr pseudo-random bit generator. The pseudo-random bit generator designed by Micali and Schnorr [30] is based on the hypothesis that the distribution of $k^e \bmod n$ for random ℓ_k-bit integers is indistinguishable from the distribution of elements of $\mathbb{Z}_n^*$. This is clearly an assumption that is very close to ours: in our case, we only added that the k that are considered are primes.

3.2 Instance-independence assumption

In [35, 34], Paillier and Villar conjectured that the $\mathsf{RSA}(e_0, n_0)$ problem (that is, the computation of e_0-th roots modulo n_0) is not significantly simpler when one was given an interactive oracle returning e-th roots modulo n, for any pair $(e, n) \neq (e_0, n_0)$ of adversary's choice. They refer to this kind of hypothesis as *non-malleability assumptions*, in the sense that an RSA challenge should not be transformed into another challenge, so that the oracle would help. To avoid confusion with the classical security notion of non-malleability [18], we rename in this paper the [35, 34]-style assumptions as *instance-independence assumptions*.

Intuitively, in this paper, we consider an instance-independence variant about the D-RSA-SEP problem, conjecturing that $\mathsf{D\text{-}RSA\text{-}SEP}(e_0, n_0)$ is not significantly simpler when given an $\mathsf{RSA}(e, n)$ oracle, where we restrict even more the control of the adversary than in [35, 34]: this latter can only have access to an oracle returning e-th roots modulo n_0, with e prime and where modulus is fixed to n_0.

More formally, this gives the following definition.

Definition 2 (Assisted Decision RSA Short Encrypted-Prime Problem (A-D-RSA-SEP)). *Let (ℓ_e, ℓ_n, ℓ_k) be some security parameters. The challenger is given (e_0, n_0, z_0), where n_0 is an RSA ℓ_n-bit modulus, e_0 is a ℓ_e-bit random prime, and $z_0 \in \mathbb{Z}_{n_0}^*$. The challenger has q_D adaptive accesses to an oracle $\mathcal{O}_{n_0}$, which returns an e-th root of a chosen element β modulo n_0, for a chosen ℓ_e-bit prime $e \neq e_0$:*

$$\alpha \leftarrow \mathcal{O}_{n_0}(e, \beta) \ \textit{such that}\ \alpha^e = \beta \bmod n_0 \ .$$

The challenger's goal is to decide whether or not z_0 is of the form $k^{e_0} \bmod n_0$, for some $k \in \textsc{Primes}([2^{\ell_k - 1}, 2^{\ell_k}[)$.

We conjecture that the assisted decision RSA short encrypted-prime problem is at most negligibly easier than the (standard) decision RSA short encrypted-prime problem. Indeed, we suppose that the $\mathsf{RSA}(e, n_0)$ problems are sufficiently *independent* for different prime exponents e, so that the access to the (limited) RSA oracle does not "dramatically" simplify the task of the challenger. In other words, because of this supposed independence, we believe that the A-D-RSA-SEP problem, which is interactive by nature, is almost as hard as its plain version (see Definition 1). This assumption is what we call the (D-RSA-SEP, RSA) *instance-independence assumption.* We remark that this latter assumption is stronger than the ones used in [35, 34], because the problems to solve in our constructions are decisional.

4 New RSA-type Schemes

4.1 An IND-CPA encryption scheme

We describe a semantically secure RSA-type encryption scheme derived from the multiplicative knapsack cryptosystem Naccache-Stern [32].

Key Generation. Some parameters are defined, namely a bit-length ℓ_n, a bit-length ℓ_k and a maximum index I so that

$$\Pi := \prod_{i=1}^{I} p_i \le 2^{\ell_n - \ell_k - 1} \quad \text{and} \quad p_I < 2^{\ell_k - 1} ,$$

where p_i's denote the first primes sorted in increasing order. The message space is $\{0,1\}^I$. Let $n = pq$ be an ℓ_n-bit RSA modulus, made of two equal-length primes p and q. Define $\lambda = \operatorname{lcm}(p-1, q-1)$ and for an integer $e > 1$ with $\gcd(e, \lambda) = 1$, compute $d = e^{-1} \bmod \lambda$.

The public key is $\mathsf{pk} = \{n, e\}$ while the private key is $\mathsf{sk} = \{d\}$.

Encryption. To encrypt a message $m = \{m_i\}_i \in \{0,1\}^I$, one picks a random ℓ_k-bit prime k, computes $w = \prod_{i=1}^{I} {p_i}^{m_i}$ and $c = (w \cdot k)^e \bmod n$. The ciphertext is c.

Decryption. To decrypt a ciphertext c, the legitimate receiver calculates $t = c^d \bmod n$ and decomposes it into the base of primes allowing a maximal valuation of 1 for each prime p_i, i.e.,

$$t = \prod_{i=1}^{I} {p_i}^{m_i} \cdot k' \quad \text{with } m_i \in \{0,1\} \ .$$

Message m is then recovered bit-by-bit as $m_i = \begin{cases} 1 & \text{if } t \bmod p_i = 0, \\ 0 & \text{otherwise}. \end{cases}$

It is easy to show that the above scheme achieves $\mathsf{IND\text{-}CPA}$ security under the assumption that the $\mathsf{D\text{-}RSA\text{-}SEP}$ problem is hard (see Definition 1).

4.2 Making it chosen-ciphertext secure

As for most RSA-type schemes, public exponent e can be "freely" chosen in the key generation of the previous scheme, subject to the condition that $\gcd(e, \lambda) = 1$. In other words, this scheme can be modified into a public-key encryption scheme with *ephemeral key*. Namely, the public key pk can be split into two parts: a fixed part n and a variable (ephemeral) part e. If the private key is defined as λ, the private exponent corresponding to e can be computed from λ as $d = e^{-1} \bmod \lambda$.

Suppose for a moment that the ciphertexts are formed by pairs (c, e) obtained from the previous $\mathsf{IND\text{-}CPA}$ encryption scheme from a random ephemeral key e. Now, in the security proof, when adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ requests the decryption of (c, e), we simply call the RSA-oracle and get back corresponding plaintext m (or $\perp$). This however supposes that $e \ne e_*$, where (c_*, e_*) denotes the challenge ciphertext, because of the restriction on the oracle in the $\mathsf{A\text{-}D\text{-}RSA\text{-}SEP}$ problem. First, we note that the probability that $\mathcal{A}_1$ queries on a ciphertext (c, e) with

$e = e_*$ is negligible as $\mathcal{A}_1$ has no *a priori* knowledge on the value of e_*. For this, we assume that the set from which the ephemeral key e is drawn from is sufficiently large.

For $\mathcal{A}_2$, things are more complicated: $\mathcal{A}_2$ knows e_* from the challenge ciphertext (c_*, e_*). The limitation on the oracle in the A-D-RSA-SEP problem prevents to obtain the corresponding plaintext from the RSA oracle (remember that only primes $e \neq e_*$ can be submitted). Therefore we adopt another strategy: in some sense we "forbid" $\mathcal{A}_2$ to submit ciphertexts of the form (c, e_*) by making "infeasible" for anyone to reuse a prior ephemeral key picked by someone else. To this end, we append a "proof", say π, to the ciphertexts and modify the generation of e accordingly. The ciphertexts of are now represented by tuples (c, e, π). There are several possible realizations of this idea ([6]).

One-time mappable chameleon hash function. Our proof π relies on a special kind of chameleon hash functions. As will become apparent, chameleon hash functions will be used so that the value of e can be recovered from (c, π). Their salient feature is that it is infeasible to find a second pair $(c', \pi') \neq (c, \pi)$ that will yield the same e, *unless some trapdoor information on the generation of* e *is given.* Since $\mathcal{A}_2$ has no such information about e_*, the probability that $\mathcal{A}_2$ requests the decryption of a ciphertext (c, e_*, π) with $(c, \pi) \neq (c_*, \pi_*)$ is negligible.

The attentive reader will observe there is a last technical complication: e_*, the ephemeral key used in the ciphertext challenge, is an input to the reduction algorithm and so it is unclear how to get a pair (c_*, π_*) that will lead to the value of e_*.[6] This problem is alleviated through the use of *one-time mappable* chameleon hashing. Basically, a one-time mappable hash function is a keyed hash function that can be controlled for *one* given pair of input/output.

Definition 3 (One-time mappable hash function). *Given a family of hash functions* $\{\mathcal{H}^{(i)}\}_j$*, a* one-time mappable hash function $\mathcal{H}^{(a)}$ *is a hash function such that on input* (x_*, h_*)*, it is easy to find an index* a *such that* $\mathcal{H}^{(a)}(x_*) = h_*$.

It turns out that it is straightforward to construct a one-time mappable hash function from a regular hash function. Let $\mathcal{H}$ be a function mapping strings of arbitrary length to strings of ℓ_h bits. From $\mathcal{H}$, we define the family of hash functions $\{\mathcal{H}^{(i)}\}_i$, $0 \leq i < 2^{\ell_h}$, given by

$$\mathcal{H}^{(i)} : \{0,1\}^* \to \{0,1\}^{\ell_h}, x \mapsto \mathcal{H}^{(i)}(x) := (\mathcal{H}(x) + i) \bmod 2^{\ell_h} \quad . \tag{2}$$

Now given a pair (x_*, h_*), if we set $a := (h_* - \mathcal{H}(x_*)) \bmod 2^{\ell_h}$, we obviously have $\mathcal{H}^{(a)}(x_*) = h_*$.

The above construction can be composed with chameleon hash functions so as to obtain *one-time mappable chameleon hash functions.* Moreover, it is easy to see that doing so, Properties (i)–(iii) for chameleon hash functions (cf. § 2.3) are

[6] This technical complication appears clearly in the security analysis; see § 4.3.

still satisfied, provided that function $\mathcal{H}$ in Eq. (2) is second pre-image collision resistant.

In order to satisfy the instance-independence assumption, ephemeral keys e are required to be ℓ_e-bit *primes*. Hence, we need to modify the output of the hash function to accommodate this additional requirement.[7] We give below an example of such a function that maps integers (of length at most ℓ_h bits) to ℓ_e-bit primes:

$$\begin{aligned}\mathcal{J}_{\ell_e} &: \{0,1\}^{\ell_h} \to \textsc{Primes}([2^{\ell_e-1}, 2^{\ell_e}[),\\ &x \mapsto \mathcal{J}_{\ell_e}(x) := \mathrm{NextPrime}_{\ell_e}[T(x)] \quad \text{with } T(x) := x \cdot 2^{\ell_e-\ell_h-1} + 2^{\ell_e-1} \ .\end{aligned}$$

For an integer t, function $\mathrm{NextPrime}_{\ell_e}(t)$ returns the first prime larger than t, provided that this prime is smaller than 2^{ℓ_e}; if no such prime exists, it returns the first prime larger than 2^{ℓ_e-1}. For any $x_1, x_2 \in \{0,1\}^{\ell_h}$, $T(x_1) = T(x_2)$ implies $x_1 = x_2$ and thus function T is injective. In our construction, ℓ_e and ℓ_h satisfy $\ell_e \gg \ell_h$ so that there always[8] exists an ℓ_e-bit prime in a range of $2^{\ell_e-\ell_h-1}$ consecutive ℓ_e-bit integers, which in turn implies that function $\mathcal{J}_{\ell_e}$ is injective too.

Putting all together, the one-time mappable chameleon hash to be used, obtained from chameleon hash function of Example 2, is given by:[9]

$$\begin{aligned}\textsc{ChamHash}_v^{(a)} &: \{0,1\}^{\ell_c} \times \{0,1\}^{\ell_n+\ell_c+2\kappa} \to \textsc{Primes}([2^{\ell_e-1}, 2^{\ell_e}[),\\ &(m,r) \mapsto \textsc{ChamHash}_v^{(a)}(m,r) := \mathcal{J}_{\ell_e}\Big(\mathcal{H}^{(a)}(g^r\, v^m \bmod n, v)\Big) \ . \qquad (3)\end{aligned}$$

Our scheme. We are now ready to present our IND-CCA-secure RSA-type scheme. The security analysis is presented in the next section.

Key Generation. On input security parameter κ, bit-lengths ℓ_n, ℓ_k, and index I are defined as in the basic scheme (see § 4.1). The message space is $\{0,1\}^I$. Let $n = (2p'+1)(2q'+1)$ be a strong RSA modulus and let $\mathrm{QR}(n)$ denote the set of quadratic residues modulo n. Let also $\mathcal{H} : \mathrm{QR}(n) \to \{0,1\}^{\ell_h}$ be a second pre-image resistant hash function, for some bit-length $\ell_h \geq \kappa$. Select a parameter $a \leftarrow \{0,1\}^{\ell_h}$ and define $\mathcal{H}^{(a)}(x) := (\mathcal{H}(x) + a) \bmod 2^{\ell_h}$. For some bit-length $\ell_c \geq \kappa$, let $\mathcal{G} : \mathbb{Z}_n \to \{0,1\}^{\ell_c}$ be a second pre-image resistant hash function. Finally, let the one-time mappable chameleon hash function given by Eq. (3) for some $g \leftarrow \mathrm{QR}(n)$ and bit-length ℓ_e satisfying $(\kappa + \log\kappa) \leq \ell_e < \ell_n/2$ and $\ell_h \ll \ell_e$.[10]

[7] Actually, so-called *division-intractable hash functions* [21] would suffice for our purposes; but as pointed out in [14], the easiest way to meet the requirement of division-intractability is to construct hash functions mapping strings to prime numbers.

[8] Unless a breaking, surprising fact on the distribution of prime numbers.

[9] The pair $(g^r\, v^m \bmod n, v)$ is viewed as a bit-string in the definition of $\textsc{ChamHash}_v^{(a)}$.

[10] Typical values for the different parameters are $\ell_n = 2048$, $\ell_k = 160$, $I = 216$, $\ell_h = \ell_c = 128$, and $\ell_e = 256$.

The public key is $\mathsf{pk} = \{n, g, a\}$ while the private key is $\mathsf{sk} = \{\lambda\}$ with $\lambda = 2p'q'$.

Encryption. To encrypt a message $m = \{m_i\} \in \{0,1\}^I$, do the following:

1. choose a random integer $y \leftarrow \{0,1\}^{\ell_n+\kappa}$ and compute $v = g^y \bmod n$;
2. choose a random pair $(c, r) \leftarrow \{0,1\}^{\ell_c} \times \{0,1\}^{\ell_n+\ell_c+2\kappa}$ and obtain prime $e = \textsc{ChamHash}_v^{(a)}(c, r)$;
3. pick a random ℓ_k-bit prime k and compute $c' = (w \cdot k)^e \bmod n$ with $w = \prod_{i=1}^{I} {p_i}^{m_i}$;
4. compute $\rho = r + y(c - \mathcal{G}(c'))$.

The ciphertext is $C = (c', v, \rho)$.

Decryption. To decrypt a ciphertext $C = (c', v, \rho)$, the legitimate user does the following:

1. recover $e = \textsc{ChamHash}_v^{(a)}(\mathcal{G}(c'), \rho)$;
2. compute $d = e^{-1} \bmod \lambda$;
3. compute $t = c'^d \bmod n$ and derive bit-by-bit corresponding message m as in the basic scheme.

Observe that there is no validity test in the decryption: any ciphertext is considered as valid. This is not the first time that an encryption scheme which always decrypts is adaptively secure (e.g., [37]).

4.3 Security analysis

We prove that the scheme is IND-CCA under the (D-RSA-SEP, RSA) assumption (see § 3.2).

We have the following theorem:

Theorem 1. *Let $\mathcal{A}$ be an adversary which can break the* IND-CCA *security with success probability ε under a chosen-ciphertext attack within time τ. Assume that $\mathcal{H}$ and $\mathcal{G}$ are second pre-image resistant hash functions. Then the* (D-RSA-SEP, RSA) *problem can be solved with success probability $\frac{\varepsilon}{2}$ and within time $\tau + q_D \cdot T_{\mathcal{O}_n} + \mathsf{poly}(\kappa)$, where $T_{\mathcal{O}_n}$ is the time of an* RSA*-oracle execution and q_D is the number of decryption queries.*

In other words, assuming the $\mathsf{RSA}(e, n_0)$ instances are independent for different prime exponents e, our scheme is as secure as the regular (i.e., non-interactive) version of the decision RSA short encrypted-prime problem.

Proof (of Theorem 1). We give a proof that our scheme achieves IND-CCA security, namely indistinguishability under adaptive chosen-ciphertext attacks, in the standard model. As stated in the theorem, we consider $\mathcal{H}$ and $\mathcal{G}$ to be second pre-image resistant hash functions, being said that if the adversary breaks this property, the scheme becomes insecure.

We use the following notation. Letters with stars (e_*) correspond to the adversary challenge, letters with zeros (e_0) correspond to the reduction challenges, and letters with j index (e_j) correspond to j-th decryption queries.

We suppose that there exists an adversary $\mathcal{A}$ able to break the scheme in the sense of IND-CCA. We will use $\mathcal{A}$ to devise a reduction algorithm $\mathcal{R}$ solving the D-RSA-SEP problem. On input $(e_0, n_0, z_0, \ell_{k_0})$, $\mathcal{R}$ is given access to an RSA-oracle $\mathcal{O}_{n_0}$ and has to say whether or not z_0 is of the form ${k_0}^{e_0} \pmod{n_0}$ for some ℓ_{k_0}-bit prime k_0.

Setup. The adversary's environment is simulated. We set strong RSA modulus $n = n_0$ and prime $e_* = e_0$, and pick a random element $g \leftarrow \mathrm{QR}(n)$. We choose a random integer $y_* \leftarrow \{0,1\}^{\ell_n+\kappa}$ and compute $v_* = g^{y_*} \bmod n$. We also choose a random pair $(c_*, r_*) \leftarrow \{0,1\}^{\ell_c} \times \{0,1\}^{\ell_n+\ell_c+2\kappa}$. Finally, parameter $a \in \{0,1\}^{\ell_h}$ is selected so that $\textsc{ChamHash}^{(a)}_{v_*}(c_*, r_*) = e_*$ —this is where the one-time mappability feature is used. To start the simulation, the public key $\{n, g, a\}$ is given to $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$.

Decryption queries. When $\mathcal{A}_1$ requests the decryption of $C_j = (c'_j, v_j, \rho_j)$, we compute $e_j = \textsc{ChamHash}^{(a)}_{v_j}(\mathcal{G}(c'_j), \rho_j)$. If $e_j \neq e_*$, we forward (c'_j, e_j) to $\mathcal{O}_{n_0}$ and get back $t_j = {c'_j}^{1/e_j} \bmod n$. From t_j, we recover the corresponding plaintext and return it to $\mathcal{A}_1$. If $e_j = e_*$, we abort the simulation.

Challenge. At some point, $\mathcal{A}$ will issue two plaintexts m_0 and m_1. For a random bit $b \leftarrow \{0,1\}$, we produce the challenge ciphertext corresponding to message m_b (written as $\{(m_b)_i\}$ in binary form), $C_* = (c'_*, v_*, \rho_*)$, where

$$\begin{cases} c'_* = \left(\prod_{1\le i\le I} {p_i}^{(m_b)_i}\right)^{e_*} \cdot z_0 \bmod n \\ \rho_* = r + y_*(c_\star - \mathcal{G}(c'_*)) \end{cases} .$$

The challenge corresponds to ephemeral public exponent e_* and is valid as soon as z_0 is of the form ${k_0}^{e_0} \bmod n_0$ for some ℓ_{k_0}-bit prime k_0. On the contrary, if z_0 is a random of $\mathbb{Z}^*_{n_0}$, the challenge is independent of m_b.

Additional decryption queries. Upon receiving the challenge ciphertext, $\mathcal{A}_2$ is allowed to make further decryption queries $(c'_j, v_j, \rho_j) \neq (c'_*, v_*, \rho_*)$. We proceed as before (i.e., as for $\mathcal{A}_1$).

Outcome. Eventually, provided that the simulation is perfect, $\mathcal{A}$ will return b with success probability ε and within time τ. Let b' denote the output of $\mathcal{A}$. If $b = b'$, we return 1 as the output (meaning that we suppose that $z_0 = {k_0}^{e_0} \bmod n_0$ for some ℓ_{k_0}-bit prime k_0). On the contrary, if $b \neq b'$ (or if $\mathcal{A}$ does not return an output), we return a random answer to the challenge.

There is one case for which the simulation is not perfect, the challenge phase excepted (this is addressed just after): when $\mathcal{A}_1$ or $\mathcal{A}_2$ queries on a ciphertext (c'_j, v_j, ρ_j) leading to $e_j = e_*$. This yields

$$\textsc{ChamHash}^{(a)}_{v_j}(\mathcal{G}(c'_j), \rho_j) = \textsc{ChamHash}^{(a)}_{v_*}(\mathcal{G}(c'_*), \rho_*)$$

$\Rightarrow\ \mathcal{H}^{(a)}(g^{\rho_j}\, {v_j}^{\mathcal{G}(c'_j)} \bmod n, v_j) = \mathcal{H}^{(a)}(g^{\rho_*}\, v_*^{\mathcal{G}(c'_*)} \bmod n, v_*)$ since $\mathcal{J}$ is injective

$\Rightarrow\ v_* = v_j$ and $g^{\rho_j}\, v_j^{\mathcal{G}(c'_j)} \equiv g^{\rho_*}\, v_*^{\mathcal{G}(c'_*)} \pmod{n}$ since $\mathcal{H}$ is supposed second-preimage resistant

$\Rightarrow\ (\rho_j - \rho_*) + y_*(\mathcal{G}(c'_j) - \mathcal{G}(c_*)) \propto p'q'$

which leads to the factorization of n as soon as the multiple of $p'q'$ is non zero, and so to the solution to the (D-RSA-SEP, RSA) assisted problem. To conclude, it is sufficient to see that it happens only negligibly that the multiplier is zero, as, since $\mathcal{G}$ is second pre-image resistant, finding such pair (c'_j, ρ_j) for the adversary is harder than writing y_* as

$$y_* = \frac{\rho_j - \rho_*}{\mathcal{G}(c'_j) - \mathcal{G}(c_*)} \ .$$

As the view of the adversary on y_* is limited to $y_* \bmod p'q'$ (by v_*, since r statistically hides y_* in ρ^*), such recovery for the attacker is only possible with probability $2^{-\kappa}$. Thus, simulation aborts are not harmful, as they allow to solve the reduction problem.

Concerning the challenge phase, there are two cases:

1. First, if z_0 is not of the form ${k_0}^{e_0} \bmod n_0$, i.e., if z_0 is a random of $\mathbb{Z}^*_{n_0}$. Then, the challenge ciphertext is malformed, and the simulation is imperfect because of this. However, one can conclude, as the advantage of the adversary is necessarily 0, since the ciphertext is independent of m_b. Therefore, our answer is right with a probability $\frac{1}{2}$.
2. Second, if $z_0 = {k_0}^{e_0} \bmod n_0$ for some ℓ_{k_0}-bit prime k_0. In this case, the adversary view is normal, and it should have its classical advantage ε. Then, we have an advantage of ε against the decisional problem.

In conclusion, we have an advantage $\frac{\varepsilon}{2}$ over the A-D-RSA-SEP problem; or equivalently, $\varepsilon_{\mathcal{R}} = \frac{\varepsilon}{2}$. □

4.4 Further schemes

Typically, for a 2048-bit modulus, the previous scheme allows one to encrypt 216-bit messages. The message bandwidth can however be significantly increased by deriving an RSA-type scheme from the recent scheme presented in [12] (rather than the basic Naccache-Stern scheme). The above conversion applies in the same way.

There exist other IND-CPA-secure RSA-type schemes allowing to freely exponent e, under various assumptions. These include by Pointcheval scheme [38] (based on dependent-RSA problem) or CGHN scheme [11] by Catalano, Gennaro, Howgrave-Graham and Nguyen (based on decisional small e-th residues problem). Again, the above conversion applies in the same way to get corresponding IND-CCA-secure RSA-type encryption schemes. Moreover, there are several possible constructions of one-time mappable chameleon hash functions. We refer the reader to Appendix A for a description of the so-obtained schemes.

5 Conclusion

In the context of RSA-based cryptography, instance-independence assumptions were used in the literature as a way to provide evidence that some schemes provably attaining a certain level of security in the random oracle would never be proved to achieve the same level in the standard model (in the black-box setting). In contrast, in this paper, we have shown positive applications of instance-independence assumptions in order to construct (simple) new encryption schemes with security reductions to some decisional problems. Among the possible instances of our technique, we pointed out a chosen-ciphertext secure encryption scheme of the Naccache-Stern knapsack, as well as schemes based on the dependent-RSA problem and decisional small e-th residues problem.

References

1. M. Abdalla, M. Bellare, and P. Rogaway. The oracle Diffie-Hellman assumptions and an analysis of DHIES. In: CT-RSA 2001, LNCS, vol. 2020: pp. 143–158. Springer, Heidelberg (2001)
2. M. Abe, Y. Cui, H. Imai, and E. Kiltz. Efficient hybrid encryption from ID-based encryption. *Designs, Codes and Cryptography* (To appear)
3. M. Bellare, C. Namprempre, D. Pointcheval, and M. Semanko. The one-more-RSA-inversion problems and the security of Chaum's blind signature scheme. *Journal of Cryptology*, 16(3):185–215 (2003)
4. M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In: CCS 1993, pp. 62–73. ACM Press, New York (1993)
5. M. Bellare and P. Rogaway. Optimal asymmetric encryption. In: EUROCRYPT '94, LNCS, vol. 839, pp. 92–111. Springer, Heidelberg (1994)
6. D. Boneh, R. Canetti, S. Halevi, and J. Katz. Chosen-ciphertext security from identity-based encryption. *SIAM Journal on Computing*, 36(5):915–942 (2006)
7. D. Boneh and J. Katz. Improved efficiency for CCA-secure cryptosystems built using identity- based encryption. In: CT-RSA 2005, LNCS, vol. 3376, pp. 87–103. Springer, Heidelberg (2005)
8. G. Brassard, D. Chaum, and C. Crépeau. Minimum disclosure proofs of knowledge. *Journal of Computer and System Sciences*, 37(2):156–189 (1988)
9. J. Camenisch and M. Stadler. Efficient group signature schemes for large groups (extended abstract). In: CRYPTO '97, LNCS, vol. 1294, pages 410–424. Springer, Heidelberg (1997)
10. R. Canetti, S. Halevi, and J. Katz. Chosen-ciphertext security from identity-based encryption. In: EUROCRYPT 2004, LNCS, vol. 3027, pages 207–222. Springer, Heidelberg (2004)
11. D. Catalano, R. Gennaro, N. Howgrave-Graham, and P.Q. Nguyen. Paillier's cryptosystem revisited. In: CCS 2001, pp. 206–214. ACM Press, New York (2001)
12. B. Chevallier-Mames, D. Naccache, and J. Stern. Linear bandwidth Naccache-Stern encryption. In: SCN 2008, LNCS, vol. 5229, pp. 327–339. Springer, Heidelberg (2008)
13. D. Coppersmith. Small solutions to polynomial equations, and low exponent RSA vulnerabilities. *Journal of Cryptology*, 10(4):233–260 (1997)

14. J.-S. Coron and D. Naccache. Security analysis of the Gennaro-Halevi-Rabin signature scheme. In: EUROCRYPT 2000, LNCS, vol. 1807, pp. 91–101. Springer, Heidelberg (2000)
15. R. Cramer and V. Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In: CRYPTO '98, LNCS, vol. 1462, pp. 13–25. Springer, Heidelberg (1998)
16. R. Cramer and V. Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In: EUROCRYPT 2002, LNCS, vol. 2332, pp. 45–64. Springer, Heidelberg (2002)
17. A.W. Dent. A brief history of provably secure public-key encryption. In: AFRICACRYPT 2008, LNCS, vol. 5023, pp. 357–370. Springer, Heidelberg (2008)
18. D. Dolev, C. Dwork, and M. Naor. Non-malleable cryptography. *SIAM Journal on Computing*, 30(2):391–437 (2000)
19. A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In: CRYPTO '86, LNCS, vol. 263, pp. 186–184. Springer, Heidelberg (1987)
20. E. Fujisaki and T. Okamoto. How to enhance the security of public-key encryption at minimum cost. *IEICE Transaction of Fundamentals of Electronic Communications and Computer Science*, E83-A(1):24–32 (2000)
21. R. Gennaro, S. Halevi, and T. Rabin. Secure hash-and-sign signatures without the random oracle. EUROCRYPT '99, LNCS, vol. 1592, pp. 123–139. Springer, Heidelberg (1999)
22. M. Girault. An identity-based identification scheme based on discrete logarithms modulo a composite number. In: EUROCRYPT '90, LNCS, vol. 473, pp. 481–486. Springer, Heidelberg (1990)
23. K. Gjøsteen. A new security proof for Damgård's ElGamal. In: CT-RSA 2006, LNCS, vol. 3860, pp. 150–158. Springer, Heidelberg (2006)
24. S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299 (1984)
25. D. Hofheinz and E. Kiltz. Practical chosen ciphertext secure encryption from factoring. In: EUROCRYPT 2009, LNCS, vol. 5479, pp. 313–332. Springer, Heidelberg (2009)
26. E. Kiltz. Chosen-ciphertext security from tag-based encryption. In: TCC 2006, LNCS, vol. 3876, pp. 581–600. Springer, Heidelberg (2006)
27. H. Krawczyk and T. Rabin. Chameleon signatures. In: *Network and Distributed System Security Symposium (NDSS 2000)*, pp. 143–154. Internet Society (2000)
28. K. Kurosawa and Y. Desmedt. A new paradigm of hybrid encryption scheme. In: CRYPTO 2004, LNCS, vol. 3152, pp. 426–442. Springer, Heidelberg (2004)
29. H. Lipmaa. On CCA1-security of ElGamal and Damgård's ElGamal. Cryptology ePrint Archive, Report 2008/234 (2008)
30. S. Micali and C.-P. Schnorr. Efficient, perfect polynomial random number generators. *Journal of Cryptology*, 3(3):157–172 (1991)
31. G.L. Miller. Riemann's hypothesis and tests for primality. *Journal of Computer and System Sciences*, 13(3):300–317 (1976)
32. D. Naccache and J. Stern. A new public-key cryptosystem. In: EUROCRYPT '97, LNCS, vol. 1233, pp. 27–36. Springer, Heidelberg (1997)
33. M. Naor and M. Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In: 22nd ACM STOC, pp. 427–437. ACM Press, New York (1990)
34. P. Paillier. Impossibility proofs for RSA signatures in the standard model. In: CT-RSA 2007, LNCS, vol. 4377, pp. 31–48. Springer, Heidelberg (2007)

35. P. Paillier and J.L. Villar. Trading one-wayness against chosen-ciphertext security in factoring-based encryption. In: ASIACRYPT 2006, LNCS, vol. 4284, pp. 252–266. Springer, Heidelberg (2006)
36. O. Pandey, R. Pass, and V. Vaikuntanathan. Adaptive one-way functions and applications. In: CRYPTO 2008, LNCS, vol. 5157, pp. 57–74. Springer, Heidelberg (2008)
37. D.H. Phan and D. Pointcheval. Chosen-ciphertext security without redundancy. In: ASIACRYPT 2003, LNCS, vol. 2894, pp. 1–18. Springer, Heidelberg (2003)
38. D. Pointcheval. New public key cryptosystems based on the dependent-RSA problems. In: EUROCRYPT '99, LNCS, vol. 1592, pp. 239–254. Springer, Heidelberg (1999)
39. G. Poupard and J. Stern. Security analysis of a practical "on the fly" authentication and signature generation. In: EUROCRYPT '98, LNCS, vol. 1403, pp. 422–436. Springer, Heidelberg (1998)
40. C. Rackoff and D.R. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In: CRYPTO '91, LNCS, vol. 576, pp. 433–444. Springer, Heidelberg (1992)
41. V. Shoup. Why chosen ciphertext security matters. Technical Report RZ 3076, IBM Research (Nov. 1998)

A Further IND-CCA-Secure RSA-type Schemes

We present here further applications. The underlying decisional problems we consider are the dependent-RSA problem and small e-th residues problem. We refer to the reader to the original papers [38, 11] for a complete description. This shows the generic nature of our paradigm. This also shows that the security can be based on more classical problems than the newly introduced D-RSA-SEP problem.

A.1 Dependent-RSA variant

Here, we describe an application with the dependent-RSA problem chameleon hash $s^E \cdot v^m \bmod n$.

Key Generation. An RSA modulus $n = pq$ as well as a prime public exponent E are generated (so that $\lambda = \mathrm{lcm}(p-1, q-1)$ and $\gcd(E, \lambda) = 1$). A one-time mappable hash function family $\mathcal{H}^{(a)}$ is selected, as well as a parameter a. The public key is $\{n, E, a\}$ while the private key is λ.

Encryption. To encrypt a message $m \in \mathbb{Z}_n^*$, one picks $k, y \leftarrow \mathbb{Z}_n^*$, computes $v = k^E \bmod n$, $u = v^y \bmod n$, and $e = \mathcal{H}^{(a)}(u, v)$. Then, one picks a random $r \leftarrow \mathbb{Z}_n^*$, computes $c_1 = r^e \bmod n$, $c_2 = (r+1)^e \cdot m \bmod n$, and $s = k^{y - \mathcal{G}(c_1, c_2)} \bmod n$.

The ciphertext is (c_1, c_2, s, v).

Decryption. To decrypt a ciphertext (c_1, c_2, s, v), the legitimate user first computes $u = s^E \cdot v^{\mathcal{G}(c_1, c_2)} \bmod n$. Next, he computes $e = \mathcal{H}^{(a)}(u, v)$ and recovers the private key $d = e^{-1} \bmod \lambda$. The plaintext is obtained as $m = ((c_1{}^d \bmod n) + 1)^{-e} \cdot c_2 \bmod n$.

Intuitively, the proof of § 4.3 can be adapted to this scheme, as an adversary colliding on $s^E \cdot v^m \bmod n$ can be used to find an E-th root modulo n, which is a stronger problem that the dependent-RSA problem.

A.2 Decisional small e-th residues variant

In this section, we use the decisional small e-th residue problem and replace the strict chameleon by a non-interactive variant of Σ-protocol using the Fiat-Shamir heuristic [19], which, following the terminology of [9], is called a signature of knowledge.

Encryption. To encrypt a message $m \in \mathbb{Z}_n^*$, one computes $u = g^x \bmod n$ and $v = g^y \bmod n$ for two large random integers x and y. After, one hashes $e = \mathcal{H}^a(u, v)$. Finally, one picks a random $r \leftarrow \mathbb{Z}_n^*$ and computes $c = r^e \cdot (1 + mn) \bmod n^2$ and $s = x + \mathcal{G}(c) \cdot y$.

The ciphertext is (c, s, v).

Decryption. To decrypt a ciphertext (c, s, v), the legitimate user first computes $u = g^s \cdot v^{-\mathcal{G}(c)} \bmod n$. Next, he calculates $e = \mathcal{H}^{(a)}(u, v)$ and recovers the private key $d = e^{-1} \bmod \lambda$. The corresponding plaintext is then given by $m = \frac{((c^d \bmod n))^{-e} \cdot c \bmod n^2 - 1}{n}$.

Again, the proof of § 4.3 can be easily adapted to this scheme, as the used signature of knowledge (which is the GPS signature scheme [22, 39]) is very close to the chameleon used in § 4.2.