Relaxing Chosen-Ciphertext Security

RAN CANETTI*, HUGO KRAWCZYK**, and JESPER B. NIELSEN***

Abstract. Security against adaptive chosen ciphertext attacks (or, CCA security) has been accepted as the standard requirement from encryption schemes that need to withstand active attacks. In particular, it is regarded as the appropriate security notion for encryption schemes used as components within general protocols and applications. Indeed, CCA security was shown to suffice in a large variety of contexts. However, CCA security often appears to be somewhat *too strong:* there exist encryption schemes (some of which come up naturally in practice) that are not CCA secure, but seem sufficiently secure "for most practical purposes."

We propose a relaxed variant of CCA security, called Replayable CCA (RCCA) security. RCCA security accepts as secure the non-CCA (yet arguably secure) schemes mentioned above; furthermore, it suffices for most existing applications of CCA security. We provide three formulations of RCCA security. The first one follows the spirit of semantic security and is formulated via an ideal functionality in the universally composable security framework. The other two are formulated following the indistinguishability and non-malleability approaches, respectively. We show that the three formulations are equivalent in most interesting cases.

1 Introduction

One of the main goals of cryptography is to develop mathematical notions of security that adequately capture our intuition for the security requirements from cryptographic tasks. Such notions are then used to assess the security of protocols and schemes. They also provide abstractions that, when formulated and used correctly, greatly facilitate the design and analysis of cryptographic applications.

With respect to encryption schemes, a first step was taken with the introduction of semantic security of (public key) encryption schemes in [16]. This first step is indeed a giant one, as it introduces the basic definitional approach and techniques that underlie practically all subsequent notions of security, for encryption as well as many other cryptographic primitives.

* IBM T.J. Watson Research Center, PO Box 704, Yorktown Heights, New York 10598. Email: canetti@watson.ibm.com.

** Department of Electrical Engineering, Technion, Haifa 32000, Israel, and IBM T.J. Watson Research Center, New York, USA. Email: hugo@ee.technion.ac.il.

*** BRICS, Centre of the Danish National Research Foundation. Department of Computer Science, University of Aarhus, DK-8000 Arhus C, Denmark. Email: buus@brics.dk. Work done while at IBM T.J. Watson Research Center.

However, semantic security under chosen-plaintext attacks as defined in [16] captures only the very basic requirement from an encryption scheme, namely secrecy against "passive" eavesdroppers. In contrast, when an encryption scheme is used as a component within a larger protocol or system, a much wider array of attacks against the scheme are possible. Specifically, adversaries may have control over the messages that are being encrypted, but may also have control over the ciphertexts being delivered and decrypted. This opens new ways for the attacker to infer the outcome of the decryption of some ciphertexts by observing the system (see e.g. [6, 19]).

Several notions of "security against active adversaries" were proposed over the years in order to capture such often subtle security concerns. These notions include semantic security against Lunchtime Attacks (or, IND-CCA1 security), semantic security against Adaptive Chosen Ciphertext Attacks (or, IND-CCA2 security), Non-Malleability against the above attacks, and more [21, 23, 12, 3]. In particular, CCA2 security (where the semantic-security and the non-malleability formulations are equivalent) became the "golden standard" for security of encryption schemes in a general protocol setting. Indeed, CCA2 security (or simply CCA security) was demonstrated to suffice for a number of central applications, such as authentication and key exchange [12, 2, 10], encrypted password authentication [17], and non-interactive message transmission [7]. In addition, in [7] CCA is shown to suffice for realizing an "ideal public-key encryption functionality" within the universally composable security (UC) framework, thus demonstrating its general composability properties.

CCA security is indeed a very strong and useful notion. But is it *necessary* for an encryption scheme to be CCA-secure in order to be adequate for use within general protocol settings? Some evidence that this may not be the case has been known all along: Take any CCA-secure encryption scheme S, and change it into a scheme S' that is identical to S except that the encryption algorithm appends a 0 to the ciphertext, and the decryption algorithm discards the last bit of the ciphertext before decrypting. It is easy to see that S' is no longer CCA-secure, since by flipping the last bit of a ciphertext one obtains a different ciphertext that decrypts to the same value as the original one, and this "slackness" is prohibited by CCA security. But it seems that this added slackness of S' is of no "real consequence" in most situations. In other words, S' appears to be just as secure as S for most practical purposes. This example may seem contrived, but it in fact turns up, in thin disguises, in a number of very natural settings. (Consider for instance an implementation of some CCA-secure scheme, where for wider interoperability the decryption algorithm accepts ciphertexts represented both in big-endian and in little-endian encodings. We mention other natural examples within.) In fact, some relaxations of CCA-security were already proposed in the literature in order to address this example and similar ones, e.g. [19, 24, 1, 20]. However, while being good first steps, these notions were not fully justified as either sufficient or necessary for applications. (See more details within.)

We propose a new relaxed version of CCA-security, called Replayable CCA (RCCA) security. In essence, RCCA is aimed at capturing encryption schemes that are CCA secure "except that they allow anyone to generate new ciphertexts that decrypt to the same value as a given ciphertext." RCCA is strictly weaker than CCA security. In fact, it is strictly weaker than the relaxations in [19, 24, 1]. The rationale behind RCCA is that as far as an attacker in a protocol setting is concerned, generating *different* ciphertexts that decrypt to the *same* plaintext as a given ciphertext has the same effect as copying (or, "replaying") the same ciphertext multiple times. Since replaying a ciphertext multiple times is unavoidable even for CCA secure encryptions, RCCA security would have "essentially the same effect" as CCA security.

To substantiate this intuition, we prove that RCCA security suffices for all of the above major applications of CCA secure encryption (authentication, key exchange, etc.). We also demonstrate that the *hybrid encryption* paradigm can be based on RCCA security rather than CCA security. (Hybrid encryption calls for encrypting a key k using an asymmetric encryption and then encrypting a long message using symmetric encryption with key k.)

It should be stressed that the above rationale holds only as long as the protocol that uses the scheme makes its decisions based on the outputs of the decryption algorithm, and does not directly compare ciphertexts. Arguably, most applications of CCA secure encryption have this property. However, in some applications it is natural and helpful to directly compare ciphertexts. For instance, consider a voting scheme in which votes are encrypted, and illegal duplicate votes are detected via direct ciphertext comparison. In such cases, the full power of CCA security is indeed used.

We provide three formulations of RCCA security. The first two are formulated via "guessing games" along the lines of CCA security. The first of these, called IND-RCCA, has the flavor of "security by indistinguishability" (or, IND-CCA2 in the terminology of [3]) with a CCA-style game that allows for plaintext replay. The second notion, called NM-RCCA, has the flavor of non-malleability in a CCA-style game that allows for plaintext replay. The third notion, called UC-RCCA, is formulated via an ideal functionality in the UC framework [7]. This ideal functionality,called $\mathcal{F}_{rpke}$, is obtained by modifying the ideal functionality $\mathcal{F}_{pke}$ in [7] to explicitly allow the environment to generate ciphertexts that decrypt to the same value as a given ciphertext. Having been formulated in the UC framework, this notion provides strong and general composability guarantees. Furthermore, in the spirit of semantic security, it provides a clear and explicit formalization of the provided security guarantee. It also explicitly demonstrates the exact sense in which RCCA weakens CCA.[1]

We show that, when applied to encryption schemes where the message domain is "large" (i.e., super-polynomial in the security parameter), the three notions are equivalent.[2] When the message domain is polynomial in size, we have

[1] Krohn in [20] studies various relaxations of CCA security and their respective strengths. One of these notions is essentially the same as IND-RCCA security. However, no concrete justification for this notion is provided.

[2] We say that an encryption scheme has message domain D if, for any message $m \in D$, the process of encrypting m and then decrypting the resulting ciphertext returns m.

that UC-RCCA implies NM-RCCA, and NM-RCCA implies IND-RCCA. We also show, via a separating example, that in this case IND-RCCA does *not* imply NM-RCCA. Whether NM-RCCA implies UC-RCCA for polynomial message domains remains open.

For schemes that handle large message domains, having the three equivalent formalizations allows us to enjoy the best of each one: We have the intuitive appeal and strong composability of the UC-RCCA, together with the relative simplicity of NM-RCCA and IND-RCCA. Indeed, the case of large message domains is arguably the most interesting one, since most existing encryption schemes are either directly constructed for large message domains, or can be extended to deal with large domains in a natural way. Also, most applications of public-key encryption, e.g. encrypting an identity or a key for symmetric encryption, require dealing with large message domains.

The three notions in a nutshell. Let us briefly sketch the three notions. See Section 3 for more detailed description and rationale. First recall the standard (indistinguishability based) formulation of CCA security for public-key cryptosystems. Let $S = (gen, enc, dec)$ be a public-key encryption scheme where gen is the key generation algorithm, enc is the encryption algorithm, and dec is the decryption algorithm. Informally, S is said to be CCA secure if any feasible attacker $\mathcal{A}$ succeeds in the following game with probability that is only negligibly more than one half . Algorithm gen is run to generate an encryption key e and a decryption key d. $\mathcal{A}$ is given e and access to a decryption oracle $dec(d, \cdot)$. When $\mathcal{A}$ generates a pair m_0, m_1 of messages, a bit $b \xleftarrow{R} \{0,1\}$ is chosen and $\mathcal{A}$ is given $c = enc(e, m_b)$. From this point on, $\mathcal{A}$ may continue querying its decryption oracle, with the exception that if $\mathcal{A}$ asks to decrypt the "test ciphertext" c, then $\mathcal{A}$ receives a special symbol `test` instead of the decryption of c. $\mathcal{A}$ succeeds if it outputs b.

IND-RCCA is identical to CCA, with the exception that the decryption oracle answers `test` whenever it is asked to decrypt *any ciphertext that decrypts to either* m_0 *or* m_1*,* even if this ciphertext is different than the test ciphertext c. Indeed, in the IND-RCCA game the ability to generate new ciphertexts that decrypt to the test ciphertext does not help the adversary. (Yet, it is not immediately clear from this formulation that we did not weaken the security requirement by too much. The justification for this notion comes mainly from its equivalence with UC-RCCA, described below.)

NM-RCCA is identical to IND-RCCA, with the exception that $\mathcal{A}$ succeeds if $m_0 \neq m_1$ and it outputs *a ciphertext* c' *that decrypts to* m_{1-b}. Note that if we required $\mathcal{A}$ to output m_{1-b} explicitly, we would get a requirement that is only a reformulation of IND-RCCA. So the difference is in the fact that here $\mathcal{A}$ is only required to output an encryption of m_{1-b}, without necessarily being

Thus the larger the domain, the stronger the requirement. (Encryption schemes with large message domain should not be confused with encryption schemes that guarantee security only if the message is taken uniformly from a large domain. The latter is a weak notion of security, whereas the former is only a correctness requirement, and can be used in conjunction with any security requirement.)

able to output m_{1-b} explicitly. This requirement has a flavor of non-malleability, thus the name. (Indeed, it can be regarded as a non-malleability requirement in which the attacker is considered successful as long as the "malleability relation" it uses is not the "equality relation.")

UC-RCCA is defined via an ideal functionality, $\mathcal{F}_{rpke}$. To best understand $\mathcal{F}_{rpke}$, let us first recall the "ideal public-key encryption" functionality, $\mathcal{F}_{pke}$, from [7], that captures CCA security.[3] In fact, instead of getting into the actual mechanism of $\mathcal{F}_{pke}$ (see Section 3), let us only sketch the security guarantee it provides. Functionality $\mathcal{F}_{pke}$ captures the behavior of an "ideal encryption service." That is, $\mathcal{F}_{pke}$ provides an encryption interface that is available to all parties, and a decryption interface that is available only to one privileged party, the decryptor. When querying the encryption interface with some message m, a ciphertext c is returned. The value of c is chosen by the adversary, without any knowledge of m. This guarantees "perfect secrecy" for encrypted messages. When the decryption interface is queried with a "legitimate encryption of m" (i.e., with a string c that was the outcome of a request to encrypt m), then the returned value is m. Since there is no requirement on how "illegitimate ciphertexts," i.e. strings that were not generated using the encryption interface, are being decrypted, $\mathcal{F}_{pke}$ allows the adversary to choose the decryption values of these ciphertexts.

Functionality $\mathcal{F}_{rpke}$ is identical to $\mathcal{F}_{pke}$, except that it allows the adversary to request to decrypt "illegitimate ciphertexts" to the same value as some previously generated legitimate ciphertext. This directly captures the relaxation where the adversary is allowed to generate new ciphertexts which decrypt to the same (unknown) value as existing ciphertexts. It also demonstrates that RCCA does not weaken CCA-security beyond allowing for "plaintext replay" by the attacker.

Between RCCA security and CCA security. As sketched above, RCCA security allows anyone to modify a given ciphertext c into a different ciphertext c', as long as c and c' decrypts to the same message. One potential strengthening of RCCA security is to require that it will be possible to detect, given two ciphertexts c and c', whether one is a "modified version" of the other. (Indeed, the "endian changing" example given above has this additional property.) Here it is natural to distinguish between schemes where the detection algorithm uses the secret decryption key, and schemes where the detection can be done given only the public encryption key. We call such schemes secretly detectable RCCA (sd-RCCA) and publicly detectable RCCA (pd-RCCA), respectively.

We first observe that pd-RCCA security is essentially equivalent to the notions proposed by Krawczyk [19], Shoup [24], and Ann, Dodis and Rabin [1]. (The notions are called, respectively, loose ciphertext-unforgeability, benign malleability, and generalized CCA security, and are essentially the same.) Next we study the relations between these notions. It is easy to see that:

[3] In [7] it is mistakenly claimed that CCA security is a *strictly* stronger requirement than realizing $\mathcal{F}_{pke}$ for non-adaptive adversaries. However, as shown in this work, the two requirements are actually *equivalent*. The mistake in [7] and the equivalence proof were independently discovered in [18].

CCA security ⇒ pd-RCCA security ⇒ sd-RCCA security ⇒ RCCA security. We show that the two leftmost implications are strict. (The first is implied by the above "endian changing" example; the second holds for schemes with message space 3 or larger, or alternatively under the DDH assumption.) Whether sd-RCCA security is equivalent to RCCA security remains open.

Finally, we provide a generic construction that turns any RCCA secure scheme (with message domain $\{0,1\}^k$ where k is the security parameter) into a CCA scheme. The construction is quite efficient, and uses only shared-key primitives. This in essence demonstrates that the existence of an RCCA secure encryption scheme implies the existence of a CCA secure scheme without any additional computational assumptions. Also, this construction may provide an alternative way of obtaining CCA security.

Symmetric encryption. In this work we develop the RCCA notions mainly for public-key encryption. However, the notion can be adapted to the symmetric-key setting in a straightforward way. We outline this generalization in [8].

Organization. Section 2 recalls the formulation of CCA security, and establishes its equivalence with the universally composable notion of security for public-key encryption schemes (against non-adaptive adversaries) as defined in [7]. Section 3 presents the three variants of RCCA security and establishes the relationships among them. Section 4 studies the detectable variants of RCCA security. It also shows how to turn any RCCA secure scheme into a CCA secure one. Section 5 demonstrates several central applications where RCCA security can be used instead of CCA security. Most proofs are omitted from this extended abstract. They can be found in [8].

2 Prolog: On CCA security

Before introducing our RCCA definitions, we recall the formulation of CCA security. We also demonstrate that the notion of secure public-key encryption in the universally composable framework [7] is *equivalent* to CCA security. (In particular, this corrects the erroneous claim from [7] that the UC characterization is *strictly weaker* than CCA security. The mistake in [7] and the equivalence proof were discovered independently in [18].) This equivalence sets the stage for the presentation of RCCA. In particular, by comparing the UC formalizations of CCA security and of RCCA security it is easier to see that the technical relaxation from CCA to RCCA coincides with the intuition behind the later notion as described above, namely, that "replayable CCA" is identical to CCA except for the added ability of the attacker to generate new ciphertexts that decrypt to the same plaintexts as previously seen ciphertexts. We start by establishing the basic formal setting for public-key encryption schemes.

2.1 CCA secure encryption schemes

Public-key encryption schemes. Throughout the paper we consider (public key) encryption schemes as triples of probabilistic polynomial-time algorithms $S =$

(gen, enc, dec) together with an ensemble of finite domains (sets) $\mathcal{D} = \{D_k\}_{k\in\mathbf{N}}, D_k \subset \{0,1\}^*$. Algorithm gen, on input 1^k (k is a security parameter), generates a pair of keys (e, d). The encryption and decryption algorithms, enc and dec, satisfy that if $(e, d) = gen(1^k)$, then for any message $m \in D_k$ we have $dec_d(enc_e(m)) = m$ except with negligible probability. The range of the decryption function may include a special symbol `invalid` $\notin \mathcal{D}_k, \forall k$.

The CCA Game

The game proceeds as follows, given an encryption scheme $S = (gen, enc, dec)$, an adversary F, and value k for the security parameter.

Key generation: Run $(e, d) \leftarrow gen(1^k)$, and give e to F.

First decryption stage: When F queries `(ciphertext,`c`)`, compute $m = dec_d(c)$ and give m to F.

Encryption stage: When F queries `(test messages,`m_0, m_1`)`, $m_0, m_1 \in D_k$, and $m_0 \neq m_1$, compute $c^* = enc_e(m_b)$ where $b \stackrel{\text{R}}{\leftarrow} \{0,1\}$, and give c^* to F. (This step is performed only once.)

Second decryption stage: When F queries `(ciphertext,`c`)` after c^* is defined, proceed as follows. If $c = c^*$ then give `test` to F.[a] Otherwise, compute $m = dec_d(c)$ and give m to F.

Guessing stage: When F outputs `(guess,`b'`)`, the outcome of the game is determined as follows. If $b' = b$ then F wins the game. Otherwise, F loses the game.

Fig. 1. The CCA Game.

[a] The symbol `test` is a reserved symbol, which is different from all possible outputs of dec.

CCA security. We recall the definition of CCA security (or IND-CCA2) for public key encryption schemes. See [23, 12, 3]. Let $S = (gen, enc, dec)$ be an encryption scheme over domain $\mathcal{D} = \{D_k\}_{k\in\mathbf{N}}$. This definition, presented next, is based on the CCA game described in Figure 1.[4]

Definition 1. *An encryption scheme S is said to be* CCA-secure *if any polynomial-time adversary F wins the IND-CCA game of Figure 1 with probability that is at most negligibly more than one half.*

2.2 Equivalence of CCA and UC security of encryption schemes

A UC characterization of CCA security. Here we assume some familiarity of the reader with the UC framework. See [8] for a quick review. Within the

[4] The explicit requirement in this game that $m_0 \neq m_1$ is immaterial for the definition of CCA and the later definition of IND-RCCA (in which choosing $m_0 = m_1$ is of no benefit to the attacker), but will be substantial in our definition of NM-RCCA. Thus, for the sake of uniformity we present all our definitions using the explicit requirement $m_0 \neq m_1$.

UC framework, public-key encryption is defined via the public-key encryption functionality from [7], denoted $\mathcal{F}_{pke}$ and presented in Figure 2. Functionality $\mathcal{F}_{pke}$ is intended at capturing the functionality of a public-key encryption scheme as a *tool* to be used within other protocols. In particular, $\mathcal{F}_{pke}$ is written in a way that allows realizations that consist of three non-interactive algorithms without any communication. (The three algorithms correspond to the key generation, encryption, and decryption algorithms in the traditional definitions.) All the communication is left to the higher-level protocols that use $\mathcal{F}_{pke}$.

Referring to Figure 2, we note that id serves as a unique identifier for an instance of functionality $\mathcal{F}_{pke}$ (this is needed in a general protocol setting when this functionality can be composed with other components, or even with other instantiations of $\mathcal{F}_{pke}$). The "public key value" e has no particular meaning in the ideal scenario beyond serving as an identifier for the public key related to this instance of the functionality, and can be chosen arbitrarily by the attacker. Also, in the ideal setting ciphertexts serve as identifiers or tags with no particular relation to the encrypted messages (and as such are also chosen by the adversary). Yet, rule 1 of the decryption operation guarantees that "legitimate ciphertexts", i.e. those produced and recorded by the functionality under an `Encrypt` request, are decrypted correctly and the resultant plaintexts remain unknown to the adversary. In contrast, ciphertexts that were not legitimately generated can be decrypted in any way chosen by the ideal-process adversary (yet, since the attacker obtains no information on legitimately encrypted messages, illegitimate ciphertexts will be decrypted to values that are independent from the legitimately encrypted messages.) Note that illegitimate ciphertexts can be decrypted to different values in different activations. This provision allows the decryption algorithm to be non-deterministic with respect to ciphertexts that were not legitimately generated.

$\mathcal{F}_{pke}$ is parameterized by $\mathcal{D} = \{D_k\}_{k\in\mathbf{N}}$, the ensemble of domains of the messages to be encrypted. Given security parameter k, $\mathcal{F}_{pke}$ encrypts messages in domain D_k.

Remarks. In [7] $\mathcal{F}_{pke}$ is slightly different. Specifically, when invoked to encrypt a message m, the functionality there is instructed to hand $|m|$, the length of m, to the adversary. Here we do not hand $|m|$ to the adversary. This means that a protocol that realizes $\mathcal{F}_{pke}$ may not reveal any information on $|m|$, beyond the fact that $m \in \mathcal{D}_k$. (Indeed, knowing that $m \in D_k$ may by itself reveal some information on the length of m.)

$\mathcal{F}_{pke}$ captures CCA security. We show the equivalence between the notion of security induced by functionality $\mathcal{F}_{pke}$ and the notion of CCA security. First, recall the following natural transformation from an encryption scheme S to a protocol π_S that is geared towards realizing $\mathcal{F}_{pke}$.

1. When activated, within some P_i and with input $(\texttt{KeyGen}, id)$, run algorithm gen, output the encryption key e and record the decryption key d.
2. When activated, within some party P_j and with input $(\texttt{Encrypt}, id, e', m)$, return $enc_{e'}(m, r)$ for a randomly chosen r. (Note that it does not necessarily hold that $e' = e$.)

Functionality $\mathcal{F}_{pke}$

$\mathcal{F}_{pke}$ proceeds as follows, when parameterized by message domain ensemble $\mathcal{D} = \{D_k\}_{k \in \mathbf{N}}$ and security parameter k, and interacting with an adversary $\mathcal{S}$, and parties $P_1, ..., P_n$. (Recall that $\mathcal{S}$ can be either an ideal-process adversary or an adversary in the hybrid model.)

Key Generation: Upon receiving a value $(\texttt{KeyGen}, id)$ from some party P_i, do:

1. Hand $(\texttt{KeyGen}, id)$ to the adversary.
2. Receive a value e from the adversary, and hand e to P_i.
3. If this is the first activation then record the value e.

Encryption: Upon receiving from some party P_j a value $(\texttt{Encrypt}, id, e', m)$ proceed as follows:

1. If $m \notin D_k$ then return an error message to P_j.
2. If $m \in D_k$ then hand $(\texttt{Encrypt}, id, e', P_j)$ to the adversary. (If $e' \neq e$ or e is not yet defined then hand also the entire value m to the adversary.)
3. Receive a tag c from the adversary and hand c to P_j. If $e' = e$ then record the pair (c, m). (If the tag c already appears in a previously recorded pair then return an error message to P_j.)

Decryption: Upon receiving a value $(\texttt{Decrypt}, id, c)$ from P_i (and P_i only), proceed as follows:

1. If there is a recorded pair (c, m) then hand m to P_i.
2. Otherwise, hand the value $(\texttt{Decrypt}, id, c)$ to the adversary. When receiving a value m from the adversary, hand m to P_i.

Fig. 2. The public-key encryption functionality, $\mathcal{F}_{pke}$

3. When activated, within P_i and with input $(\texttt{Decrypt}, id, c)$, return $dec_d(c)$.

We show:

Theorem 1. *Let $S = (gen, enc, dec)$ be an encryption scheme over domain $\mathcal{D}$. Then S is CCA-secure if and only if π_S securely realizes $\mathcal{F}_{pke}$ with respect to domain $\mathcal{D}$ and non-adaptive adversaries.*

3 Replayable CCA (RCCA) Security

This section presents the three notions of security for public-key encryption sketched in the Introduction, all aimed at capturing the intuition that "the adversary should gain nothing from seeing a legitimately generated ciphertext, except for the ability to generate new ciphertexts that decrypt to the same value as the given ciphertext". Following the presentation of the three notions, we demonstrate their equivalence for encryption schemes with super-polynomial message domains, and present separating examples for polynomial message domains.

3.1 UC-RCCA: Functionality $\mathcal{F}_{rpke}$

The UC-based formulation of RCCA security is obtained by modifying the $\mathcal{F}_{pke}$ functionality from Figure 2, as to explicitly allow the adversary, together with the environment, to generate ciphertexts that decrypt to the same values as legitimately generated ciphertexts. Specifically, the new functionality, $\mathcal{F}_{rpke}$, modifies step 2 of the **Decryption** stage in Figure 2 in which the decrypting party asks to decrypt a ciphertext that was not legally generated by the functionality. In this case, $\mathcal{F}_{rpke}$ allows the adversary to fix the decrypted value to be the same as a previously encrypted value (without letting the adversary know what this value is). Thus, functionality $\mathcal{F}_{rpke}$ is defined identically to $\mathcal{F}_{pke}$ from Figure 2 except that step 2 of the **Decryption** stage is re-defined as follows:

Decryption: Upon receiving a value $(\texttt{Decrypt}, id, c)$ from P_i (and P_i only), proceed as follows:

1. If there is a recorded pair (c, m) then hand m to P_i.
2. Otherwise, hand the value $(\texttt{Decrypt}, id, c)$ to the adversary, and receive a value (α, v) from the adversary. If $\alpha =$'plaintext' then hand v to P_i. If α='ciphertext' then find a stored pair (c', m) such that $c' = v$, and hand m to P_i. (If no such c' is found then halt.)

In the following definition we use the transformation from an encryption scheme S into a protocol π_S as described in the previous section.

Definition 2. *Let S be an encryption scheme. We say that S is* UC-RCCA secure *if protocol π_S securely realizes $\mathcal{F}_{rpke}$ with respect to non-adaptive adversaries.*

Remark: We stress that protocol π_S is only required to realize $\mathcal{F}_{rpke}$ with respect to *non-adaptive* adversaries. Realizing $\mathcal{F}_{rpke}$ with respect to *adaptive* adversaries is a considerably stronger requirement. Indeed, using the techniques of [22], it can be shown that no protocol π_S can realize $\mathcal{F}_{rpke}$ with respect to adaptive adversaries, for any scheme S. Realizing $\mathcal{F}_{rpke}$ with respect to adaptive adversaries requires additional mechanisms, such as forward secure encryption or highly interactive solutions based on non-committing encryption.

3.2 NM-RCCA and IND-RCCA: The RCCA games

This section presents two notions of security for encryption schemes, that are formulated via relaxed versions of the CCA game (See Figure 1 in Section 2), and demonstrates the equivalence of these notions to the UC-RCCA formulation for encryption schemes with super-polynomial domains. The two notions are called IND-RCCA and NM-RCCA, and are defined via the IND-RCCA game and the NM-RCCA game, respectively.

The IND-RCCA game (see Figure 3) differs from the CCA game in one point: When the adversary generates a `(decrypt,` c`)` request, the answer is `test` whenever c decrypts to either m_0 or m_1. Roughly speaking, this captures the intuition that "the ability to generate different ciphertexts that decrypt to the same values as a given ciphertext should not help the adversary to win the game." (As we demonstrate below, in the case of large message spaces, this intuition is supported by the equivalence between IND-RCCA and UC-RCCA.)

The NM-RCCA game is identical to the IND-RCCA game (Figure 3), with the exception that the guessing stage is defined as follows:

The IND-RCCA game

The game proceeds as follows, given an encryption scheme $S = (gen, enc, dec)$ over a domain ensemble $\mathcal{D}$, an adversary F, and value k for the security parameter.

Key generation: Run $(e, d) \leftarrow gen(1^k)$, and give e to F.
First decryption stage: When F queries `(ciphertext,`c`)`, compute $m = dec_d(c)$ and give m to F.
Encryption stage: When F queries `(test messages,`m_0, m_1`)`, with $m_0, m_1 \in D_k$, and $m_0 \neq m_1$, compute $c^* = enc_e(m_b)$ where $b \stackrel{\text{R}}{\leftarrow} \{0, 1\}$, and give c^* to F. (This step is performed only once.)
Second decryption stage: When F queries `(ciphertext,`c`)` after c^* is defined, compute $m = dec_d(c)$. If $m \in \{m_0, m_1\}$ then give `test` to F. Otherwise, give m to F.
Guessing stage: When F outputs `(guess,`b'`)`, the outcome of the game is determined as follows. If $b' = b$ then F wins the game. Otherwise, F loses the game.

Fig. 3. The IND-RCCA game

Guessing stage for NM-RCCA: When F outputs `(guess,`c`)`, the outcome of the game is determined as follows. Compute $m = dec_d(c)$; if $m = m_{1-b}$ then F wins the game. Otherwise, F loses the game.

In order to consider F successful we require it to output an encryption of m_{1-b}. Changing this requirement to explicitly output m_{1-b} would result in a reformulation of IND-RCCA. Thus the difference from IND-RCCA is in the fact that F is only required to output an encryption of m_{1-b}, without necessarily being able to explicitly output m_{1-b} (or, equivalently, b). As we demonstrate below, this added strength relative to IND-CCA is significant for small message domains. The formulation of the attacker's goal in the NM-RCCA definition follows the non-malleability approach (and hence the name); see the discussion below.

Definition 3. *An encryption scheme S is said to be* IND-RCCA secure *(resp.,* NM-RCCA secure*) if any polynomial-time adversary F wins the IND-RCCA game of Figure 3 (resp., the NM-RCCA game) with probability that is at most negligibly more than one half.*

Discussion. The formulation of NM-RCCA is syntactically different than the usual formulation of definitions of non-malleability. We thus provide an intuitive explanation as to why NM-RCCA indeed captures the non-malleability requirement, in spite of its different formalization. Roughly speaking, an encryption scheme is called *non-malleable* [12, 3] if it is infeasible for an adversary to output ciphertexts which decrypt to plaintexts that satisfy some (non-trivial) relation with the plaintext encrypted under a given "challenge ciphertext" c^*. (A bit more precisely, the attacker is not given the plaintext encrypted under c^* but she may choose the probability distribution under which this plaintext is taken. Thus, "trivial relations" are those that hold for randomly chosen elements from

this distribution.) In the case of non-malleability under chosen-ciphertext attacks (NM-CCA) the only restriction on the attacker is that it is not allowed to include the ciphertext c^* as one of its output ciphertexts (otherwise, the attacker could always output c^* and satisfy the "equality" relation.)
In our formulation of NM-RCCA we use the above non-malleability approach to capture our intuition behind the "replayable CCA" notion. The idea is to relax NM-CCA so that there is only one form of malleability *allowed* to the attacker: outputting a ciphertext that decrypts to *the same* plaintext as c^*. In other words, the attacker is considered successful as long as it uses any relation *other than the "equality" relation.* Now, if we carry this idea to the case where the probability distribution $\mathcal{P}$, from which the plaintext to be encrypted as c^* is selected, is of the special form $\mathcal{P} = [\{m_0, m_1\}, Prob(m_0) = Prob(m_1) = 1/2]$, with m_0, m_1 chosen by the attacker, then we obtain our "non-malleability game" NM-RCCA. Beyond this intuition and relationship to general non-malleability, the main source of confidence for this definition comes from its equivalence (at least over super-polynomial domains) with the UC-RCCA notion which captures in a more explicit way the "intuitive semantics" of RCCA.

3.3 Equivalence for large message domains

Theorem 2. *Let S be an encryption scheme whose domain ensemble $\mathcal{D}$ is super-polynomial in size. Then the following three conditions are equivalent: (I) S is UC-RCCA secure; (II) S is NM-RCCA secure; (III) S is IND-RCCA secure.*

3.4 Polynomial message domains

In Theorem 2 it was proved that for super-polynomial domain ensembles, the notions UC-RCCA, NM-RCCA and IND-RCCA are equivalent. Here we show that this premise is necessary. As mentioned in the proof of Theorem 2 it holds for all domain ensembles that UC-RCCA implies NM-RCCA and that NM-RCCA implies IND-RCCA. A minimal assumption for separating is therefore that there exist IND-RCCA secure encryption schemes in the first place. We do not know whether UC-RCCA and NM-RCCA are equivalent for polynomial domain ensembles.

Theorem 3. *For all polynomial-size domain ensembles $\mathcal{D}$, if there exists an IND-RCCA secure encryption scheme with domain ensemble $\mathcal{D}$, then there exists an IND-RCCA secure encryption scheme with domain ensemble $\mathcal{D}$ which is not NM-RCCA secure.*

For the proof, let $S = (gen, enc, dec)$ be an IND-RCCA secure encryption scheme with polynomial-size domain ensemble $\mathcal{D} = \{D_k\}$. Consider the encryption scheme $S' = (gen', enc', dec')$, where $gen' = gen$, $enc'_e(m) = enc(m)_e enc_e(n)$ for $n \in_R \mathcal{D}_k \setminus \{m\}$, and $dec'_d(c_1, c_2) = dec_d(c_1)$. We claim that S' is IND-RCCA secure but not NM-RCCA secure. See details in [8].

4 Between RCCA and CCA security: Detectable RCCA

In this section we investigate the relations between RCCA security and CCA security, and introduce the notion(s) of "detectable RCCA". In particular, we establish the relationship between RCCA security and the relaxation of CCA security presented in [19, 24, 1]. These results include a (strict) separation between these notions, and consequently between RCCA and CCA security. In particular, this demonstrates that there exist encryption schemes that are not secure in the sense of the definitions from [19, 24, 1] and yet are sufficiently secure for most practical applications of CCA secure encryption. We complement these findings by showing (Section 4.3) how to construct a CCA-secure scheme (with any domain size) from any RCCA-secure scheme whose domain size is exponential in the security parameter. This transformation uses symmetric encryption and message authentication only, thus demonstrating that once RCCA security is obtained for large enough message spaces, CCA security can be obtained with moderate overhead (and without further assumptions).

Remark. Due to the equivalence between the notions of IND-RCCA, NM-RCCA and UC-RCCA security for super-polynomial domain ensembles (Theorem 2), we will usually refer to these notions under the generic term of RCCA security, and assume, for simplicity, super-polynomial domains (except if otherwise stated).

4.1 Detectable RCCA

The first and obvious fact to observe regarding the relation between RCCA and CCA security is that the former is strictly weaker than the latter. Indeed, a simple inspection of the definitions of CCA and RCCA security shows that any scheme that is CCA-secure is also RCCA-secure (under any of the definitions of RCCA security from Section 3). On the other hand, there are simple examples of encryption schemes that are RCCA but not CCA secure. One such example was mentioned in the introduction in which a CCA-secure scheme is modified by instructing the (modified) encryption to append a '0' bit to each ciphertext, and defining the (modified) decryption algorithm to ignore this bit. It is easy to see that the obtained scheme is not CCA but it does satisfy our definition(s) of RCCA security. Other examples exist. Specifically, consider the usual practice of allowing encryption schemes to add arbitrary padding to ciphertexts and later discard this padding before performing decryption. (This padding is usually required in order to align the length of ciphertexts to a prescribed length-boundary – e.g., to a multiple of 4 bytes.) Other examples include encryption schemes that naturally allow for more than one representation of ciphertexts, such as the endianess example in the introduction, or the example in [24] related to dual point representations in elliptic-curve cryptosystems.

All these examples have the property that given a certain ciphertext c, anyone can easily produce a different ciphertext c' that decrypts to the same plaintext (e.g., by changing the endianess representation or modifying the padding). Also common to these examples is the fact that if someone (say, the attacker) indeed modifies a ciphertext c into a ciphertext c' in one of the above ways then c and c' satisfy a relation that is easy to test with the sole knowledge of the public key. This fact (and the realization that these "syntactic deficiencies" do not seem to effect the actual security of these encryption schemes when used in many applications) has motivated the introduction of the relaxations of CCA security presented in [19, 24, 1] and mentioned in the introduction. Essentially, all these notions allow for "replay" of plaintexts by modifying the ciphertext, but restrict the allowed modifications to be efficiently detectable given the public key. RCCA further relaxes this requirement by allowing *any form* of ciphertext modification that do not change the plaintext, without insisting in the ability to detect such a replay. (Indeed, as argued in the introduction and demonstrated in Section 5, RCCA security is sufficient for many applications that use CCA secure encryption.)

A natural question that arises from this discussion is whether RCCA security is truly more relaxed than the notions considered in [19, 24, 1], namely, is there an RCCA-secure scheme for which the modification of ciphertexts is not "publicly detectable"? Here we provide a positive answer to this question. We start by formalizing some of the notions discussed above.

Definition 4. *Let $S = (gen, enc, dec)$ be be an encryption scheme.*

1. *We say that a family of binary relations $\equiv_e$ (indexed by the public keys of S) on ciphertext pairs is a* compatible *relation for S if for all key-pairs (e, d) of S we have:*
   - *(a) For any two ciphertexts c, c', if $c \equiv_e c'$ then $dec_d(c) = dec_d(c')$, except with negligible probability over the random choices of algorithm dec.*
   - *(b) For any plaintext m in the domain of S, if c and c' are two ciphertexts obtained as independent encryptions of m (i.e., two applications of algorithm enc on m using independent random bits), then $c \equiv_e c'$ only with negligible probability.*
2. *We say that a relation family as above is* publicly computable *(resp.* secretly computable*) if for all key pairs (e, d) and ciphertext pairs (c, c') it can be determined whether $c \equiv_e c'$ using a probabilistic polynomial time algorithm taking inputs (e, c, c') (resp. (e, d, c, c')).*
3. *We say that S is* publicly-detectable replayable-CCA (pd-RCCA) *if there exists a compatible and publicly computable relation family $\equiv_e$ such that S is secure according to the standard definition of CCA with the following modification to the CCA game from Figure 1: if, after receiving the challenge ciphertext c^*, the adversary queries the decryption oracle on a ciphertext c such that $c \equiv_e c^*$ then the decryption oracle returns* `test`*.*
   Similarly, we say that S is secretly-detectable replayable-CCA (sd-RCCA) *if the above holds for a secretly computable relation family $\equiv_e$.*

The reader can verify that the notion of pd-RCCA is essentially equivalent to the notions of loose ciphertext-unforgeability, benign malleability, and generalized CCA security, presented, respectively, in [19, 24, 1]. The term "compatible relation" is adapted from [24] where it is defined without item 1(b). Indeed, in the case of "public detectability" this requirement is redundant as it follows from the semantic security of the encryption scheme. In contrast, it is significant in the case of "secret detectability"; without it this notion is trivially equivalent to RCCA. More significantly, requirement 1(b) captures the intuition that "detectability" is useful as long as it can tell apart legitimately generated ciphertexts from those that are created by "mauling" a given (legitimate) ciphertext. We expand on this aspect in [8].

4.2 Relations among notions of detectable RCCA

Here we investigate the relations among the different flavors of detectable RCCA security, and between these and CCA security. We show:

Theorem 4. *CCA ⇒ pd-RCCA ⇒ sd-RCCA ⇒ RCCA. If RCCA-secure encryption schemes with super-polynomial message space exist then the first two implications are strict.*

It is easy to see that any encryption scheme that is CCA secure is also pd-RCCA (simply consider the equality relation as a compatible relation). Also, immediate from the definition we get that pd-RCCA implies sd-RCCA. On the other hand, as discussed above, any CCA-secure scheme can be transformed into a pd-RCCA scheme that is not CCA by appending to the ciphertext a "dummy bit" that is ignored by the decryption operation. Therefore, we get a strict separation between CCA and pd-RCCA security. In [8] we provide examples that separate between sd-RCCA and pd-RCCA.

Remark: The above results leave the open question of whether one can separate between sd-RCCA and RCCA. An interesting related question is whether there exist RCCA-secure encryption schemes, where anyone can "randomize" a given ciphertext c to another ciphertext c' so that c' looks like an "honestly generated ciphertext", even when given the decryption key. In particular, c and c' should not be "linkable" in any way. We call such encryption schemes *randomizable.*
On the other hand, given an RCCA encryption scheme that can encrypt long messages (say, $O(k)$-bit messages where k is the security parameter), one can easily obtain an sd-RCCA scheme by appending a fresh random (and sufficiently long) tag to each message before encryption. Replay of ciphertexts can then be privately detected by decrypting and comparing the received tag to previously received tags.

4.3 From RCCA security to CCA security

This section demonstrates that the existence of an RCCA secure public-key encryption scheme with large message domain implies the existence of a CCA secure public-key encryption scheme. To be precise, by large we will mean that the encryption scheme can encrypt messages of length k, where k is the security parameter.

The construction consists of two steps. First we recall that the existence of any secure encryption scheme implies the existence of a CCA secure *symmetric* encryption scheme. We then show how to combine an RCCA secure public-key encryption scheme with large domain and a CCA secure symmetric encryption scheme to obtain a CCA secure public-key encryption scheme. The first step is very inefficient, whereas the second step results in an efficient encryption scheme if the RCCA secure public-key encryption scheme and the CCA secure symmetric encryption scheme are both efficient.

For the first step, if an RCCA secure public-key encryption scheme (gen, enc, dec) exists, then a one-way function exists, e.g. the function $(e, d) = gen(r)$, where r is the random bits used in generating (e, d). Now, the existence of a one-way function through a series of well-known reductions implies the existence of a CCA secure symmetric encryption scheme (E, D) encrypting unbounded length messages.[5] To prepare for the second step, let $l(k)$ denote the key-length of (E, D) as a function of the security parameter k and consider the public-key encryption scheme $(\overline{gen}, enc, dec)$ given by

$$\overline{gen}(1^k) = gen(1^{\max(k,l(k))}) .$$

Clearly $(\overline{gen}, enc, dec)$ is RCCA secure if (gen, enc, dec) is RCCA secure. Furthermore, $(\overline{gen}, enc, dec)$ can encrypt messages of length $l(k)$. In the following we will therefore assume that we have access to a CCA secure symmetric encryption scheme (E, D) with key-length l and an RCCA secure public-key encryption scheme (gen, enc, dec) capable of encrypting messages of length l.

Consider then the public-key encryption scheme $(gen, \overline{enc}, \overline{dec})$ given by

$$\overline{enc}_e(m) = [K \stackrel{\text{R}}{\leftarrow} \{0,1\}^{l(k)}; c_1 = enc_e(K); c_2 = E_K(c_1\|m) : (c_1, c_2)] ,$$

$$\overline{dec}_d(c_1, c_2) = [K \leftarrow dec_d(c_1); c_1'\|m = D_K(c_2); \text{if } c_1' \neq c_1 \text{ then } m \leftarrow \texttt{invalid} : m] .$$

This construction of $(gen, \overline{enc}, \overline{dec})$ resembles the usual extension of a CCA secure public-key encryption scheme with a CCA secure symmetric encryption scheme for doing hybrid encryption. The only difference is the encryption of c_1 under the symmetric key. This encryption of c_1 functions as a MAC which protects against 'mauling' of c_1. Indeed, if one is not interested in hybrid encryption but only in obtaining CCA security, the encryption of m could be done as $c_1 = enc_e(K\|m); c_2 = mac_K(c_1)$, where mac is a strong message authentication code. The proof would follow the same lines as the proof of the following theorem.

Theorem 5. *If (E, D) is a CCA secure symmetric encryption scheme with key-length l and (gen, enc, dec) is an RCCA secure public-key encryption scheme capable of encrypting messages of length l, then the public-key encryption scheme $(gen, \overline{enc}, \overline{dec})$ is CCA secure.*

[5] I.e., the encryption scheme itself does not contain any bound on the message-length. However, under attack by a given adversary F the length of the messages encrypted is of course bounded by a polynomial as F is required to be PPT.

5 Using RCCA security

This section demonstrates the power of RCCA security by proving its sufficiency for several core applications of public key encryption. Prior proofs for the security of these applications relied on the CCA security (or, in some cases, on the pd-RCCA security) of the underlying encryption schemes.

We first demonstrate that RCCA security suffices for achieving secure communication channels, in exactly the same way that CCA does. That is, we show that the natural protocol of [7] for realizing secure channels given access to $\mathcal{F}_{pke}$ remains secure even if $\mathcal{F}_{pke}$ is replaced with $\mathcal{F}_{rpke}$. Next, we consider the simple key-exchange protocol of [10] based on any CCA-secure encryption scheme. We show that this protocol remains secure even when the underlying encryption is RCCA. A similar result is demonstrated with respect to the password-based key exchange protocol of Halevi and Krawczyk [17]. Finally, we demonstrate that the *hybrid encryption* paradigm can be based on RCCA security rather than CCA security. (Hybrid encryption calls for encrypting a key k using an asymmetric encryption and then encrypting a long message using symmetric encryption with key k.) For lack of space, these results are deferred to [8].

As shown by these results, RCCA security is adequate for most typical encryption applications. Yet, it should be stressed that RCCA cannot be considered as a "drop-in" replacement for *all* applications of CCA. As pointed out in the introduction, if any such application makes decisions based on the ciphertext strings themselves (e..g compares them), rather than just using the ciphertexts as inputs to the decryption algorithm, then replacing CCA with RCCA may not be secure. It is indeed unusual that such examination of the ciphertext strings is performed by applications, yet this cannot be discounted. A simple example is a voting scheme where votes are encrypted and illegitimate duplicate votes are detected via ciphertext comparison.

References

1. JH An, Y. Dodis, and T. Rabin, "On the Security of Joint Signature and Encryption", in *Eurocrypt '02*, pages 83–107, 2002. LNCS No. 2332.
2. M. Bellare, R. Canetti and H. Krawczyk, "A modular approach to the design and analysis of authentication and key-exchange protocols", *30th STOC,* 1998.
3. M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway, "Relations Among Notions of Security for Public-Key Encryption Schemes", *Advances in Cryptology - CRYPTO'98 Proceedings*, Lecture Notes in Computer Science Vol. 1462, H. Krawczyk, ed., Springer-Verlag, 1998.
4. M. Bellare and C. Namprempre, "Authenticated encryption: Relations among notions and analysis of the generic composition paradigm", *Advances in Cryptology - ASIACRYPT'00 Proceedings*, Lecture Notes in Computer Science Vol. 1976, T. Okamoto, ed., Springer-Verlag, 2000.
5. Bellovin, S. M. and Merritt, M., "Encrypted key exchange: Password- based protocols secure against dictionary attacks", In *Proceedings of the IEEE. Computer Society Symposium on Research in Security and Privacy* 1992, pp. 72–84.
6. Bleichenbacher, D., "Chosen Ciphertext Attacks against Protocols Based on RSA Encryption Standard PKCS #1", *Advances in Cryptology - CRYPTO'98*

Proceedings, Lecture Notes in Computer Science Vol. 1462, H. Krawczyk, ed., Springer-Verlag, 1998, pp. 1–12.
7. R. Canetti, "Universally Composable Security: A new paradigm for cryptographic protocols", http://eprint.iacr.org/2000/067. Extended Abstract appears in *42nd FOCS*, 2001.
8. R. Canetti, H. Krawczyk and J. Nielsen, "Relaxing Chosen Ciphertext Security," available online at http://eprint.iacr.org, 2003.
9. R. Canetti and S. Goldwasser, "A practical threshold cryptosystem resilient against adaptive chosen ciphertext attacks", *Eurocrypt'99*, 1999.
10. Canetti, R., and Krawczyk, H., "Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels", *Eurocrypt 01,* 2001.
Full version in: *Cryptology ePrint Archive* (http://eprint.iacr.org/), Report 2001/040.
11. R. Cramer and V. Shoup, "A practical public-key cryptosystem provably secure against adaptive chosen ciphertext attack", *Advances in Cryptology - CRYPTO'98 Proceedings*, Lecture Notes in Computer Science Vol. 1462, H. Krawczyk, ed., Springer-Verlag, 1998.
12. D. Dolev, C. Dwork and M. Naor, Non-malleable cryptography, *SIAM. J. Computing,* Vol. 30, No. 2, 2000, pp. 391-437. Preliminary version in *23rd Symposium on Theory of Computing (STOC),* ACM, 1991.
13. T. ElGamal, A Public-Key cryptosystem and a Signature Scheme based on Discrete Logarithms, *IEEE Transactions*, Vol. IT-31, No. 4, 1985, pp. 469–472.
14. O. Goldreich, "Foundations of Cryptography: Basic Tools", Cambridge Press, 2001.
15. O. Goldreich, S. Goldwasser and S. Micali, "How to construct random functions," *Journal of the ACM,* Vol. 33, No. 4, 210–217, (1986).
16. S. Goldwasser and S. Micali, Probabilistic encryption, *JCSS,* Vol. 28, No 2, 1984.
17. S. Halevi, and H. Krawczyk, "Public-Key Cryptography and Password Protocols", *ACM Transactions on Information and System Security*, Vol. 2, No. 3, August 1999, pp. 230–268.
18. Dennis Hofheinz and Joern Mueller-Quade and Rainer Steinwandt, "On Modeling IND-CCA Security in Cryptographic Protocols", http://eprint.iacr.org/2003/024. 2003.
19. H. Krawczyk, "The order of encryption and authentication for protecting communications (Or: how secure is SSL?)", Crypto 2001. http://eprint.iacr.org/2001/045
20. M. Krohn, "On the definitions of cryptographic security: Chosen-Ciphertext attack revisited," Senior Thesis, Harvard U., 1999.
21. M. Naor and M. Yung, "Public key cryptosystems provably secure against chosen ciphertext attacks". *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing*, 1990.
22. Jesper B. Nielsen, " Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case", in M. Yung, editor, *Advances in Cryptology - Crypto 2002*, pages 111–126,Lecture Notes in Computer Science Volume 2442.
23. C. Rackoff and D. Simon, "Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack", *CRYPTO '91,* 1991.
24. V. Shoup, "A Proposal for an ISO Standard for Public Key Encryption", Crypto Eprint archive entry 2001:112, http://eprint.iacr.org, 2001.
25. A. Sahai, "Non malleable, non-interactive zero knowledge and adaptive chosen ciphertext security", FOCS 99.