Why Provable Security Matters?

Jacques Stern

Dépt d'Informatique, ENS – CNRS, 45 rue d'Ulm, 75230 Paris Cedex 05, France.
E-mail: Jacques.Stern@ens.fr
URL: http://www.di.ens.fr/users/stern

Abstract. Recently, methods from *provable security*, that had been developped for the last twenty years within the research community, have been extensively used to support emerging standards. This in turn has led researchers as well as practitioners to raise some concerns about this methodology. Should provable security be restricted to the standard computational model or can it rely on the so-called random oracle model? In the latter case, what is the practical meaning of security estimates obtained using this model? Also, the fact that proofs themselves need time to be validated through public discussion was somehow overlooked. Building on two case studies, we discuss these concerns. One example covers the public key encryption formatting scheme OAEP originally proposed in [3]. The other comes from the area of signature schemes and is related to the security proof of ESIGN [43]. Both examples show that provable security is more subtle than it at first appears.

1 Provable Security

1.1 A Brief Perspective

Public key cryptography was proposed in the 1976 seminal article of Diffie and Hellman [19]. Shortly afterwards, Rivest, Shamir and Adleman introduced the RSA cryptosystem as a first example. From an epistemological perspective, one can say that Diffie and Hellman have drawn the most extreme consequence of a principle already stated by Auguste Kerckhoffs in the XIXth century: "Le mécanisme de chiffrement doit pouvoir tomber sans inconvénient aux mains de l'ennemi[1]". Indeed, Diffie and Hellman understood that only the deciphering operation has to be controlled by a secret key: the enciphering method may perfectly be executed by means of a publicly available key, provided it is virtually impossible to infer the secret deciphering key from the public data. Diffie and Hellman also understood that the usual challenge/response authentication method was granting non repudiation in the public key setting, thus creating the concept of digital signature, an analog of handwritten signatures.

Very quickly, it was understood that the naive textbook RSA algorithm could not be used as it stands: in particular, it has algebraic multiplicative properties which are highly undesirable from a security perspective. Accordingly, it was

[1] The enciphering mechanism may fall into the enemy's hands without drawback.

found necessary to define formatting schemes adding some redundancy, both for encryption and signature. Besides bringing an improved security level, this approach had the additional practical benefit of establishing interoperability between different implementations.

For several years, standard makers worked by trials and errors. However, in the late nineties, it was acknowledged that this was not appropriate. In 1998, D. Bleichenbacher [5] devised a subtle attack against the PKCS #1 v1.5 encryption scheme [52]. In this attack, the adversary discloses the secret key of an SSL server, based on the information coming form the error messages received when an incorrectly formatted ciphertext is submitted to the server. One year later, the work of J. S. Coron, D. Naccache and J. P. Stern [16] on one hand, D. Coppersmaith, S. Halevy and C. Jutla [13] on the other hand, resulted in breaking the ISO/IEC 9796-1 signature scheme, by showing how to manufacture a fresh message/signature pair, having previously requested the signature of a few related messages.

Thus, a more formal approach appeared necessary, borrowing methods from the theory of complexity. This approach allows a correct specification of the security requirements, which in turn can be established by means of a *security proof*.

1.2 Public key Encryption Schemes

In modern terms, a public-key encryption scheme on a message space $\mathcal{M}$ consists of three algorithms $(\mathcal{K}, \mathcal{E}, \mathcal{D})$:

- The key generation algorithm $\mathcal{K}(1^k)$ outputs a random pair of private/public keys $(\mathsf{sk}, \mathsf{pk})$, relatively to a security parameter k.
- The encryption algorithm $\mathcal{E}_{\mathsf{pk}}(m; r)$ outputs a ciphertext c corresponding to the plaintext $m \in \mathcal{M}$, using random coins r.
- The decryption algorithm $\mathcal{D}_{\mathsf{sk}}(c)$ outputs the plaintext m associated to the ciphertext c.

We will occasionnally omit the random coins and write $\mathcal{E}_{\mathsf{pk}}(m)$ in place of $\mathcal{E}_{\mathsf{pk}}(m; r)$. Note that the decryption algorithm is deterministic.

The starting point of the new approach is semantic security, also called *polynomial security/indistinguishability of encryptions*, introduced by Goldwasser and Micali [25] : an encryption scheme is *semantically secure* if no polynomial-time attacker can learn any bit of information about the plaintext from the ciphertext, except its length. More formally, an encryption scheme is semantically secure if, for any two-stage adversary $\mathcal{A} = (A_1, A_2)$ running in polynomial time, the advantage $\mathsf{Adv}^{\mathsf{ind}}(\mathcal{A})$ is negligible, where $\mathsf{Adv}^{\mathsf{ind}}(\mathcal{A})$ is formally defined as

$$2 \times \Pr\left[\begin{array}{l}(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathcal{K}(1^k), (m_0, m_1, s) \leftarrow A_1(\mathsf{pk}), \\ b \overset{R}{\leftarrow} \{0,1\}, c = \mathcal{E}_{\mathsf{pk}}(m_b) : A_2(m_0, m_1, s, c) = b\end{array}\right] - 1,$$

where the probability space includes the internal random coins of the adversary, and m_0, m_1 are two equal length plaintexts chosen by A_1 in the message-space $\mathcal{M}$.

Another security notion has been defined in the literature [20], called *non-malleability* (NM). Informally, it states that it is impossible to derive from a given ciphertext a new ciphertext such that the plaintexts are meaningfully related. We won't dicuss this notion extensively since it has been proven equivalent to semantic security in an extended attack model (see below).

The above definition of semantic security covers passive adversaries. It is a *chosen–plaintext* or CPA attack since the attacker can only encrypt plaintext. In extended models, the adversary is given restricted or non restricted access to various oracles. A *plaintext-checking* oracle receives as its input a pair (m, c) and answers whether c encrypts message m. This gives rise to *plaintext-checking attacks* [44]. A *validity-checking* oracle answers whether its input c is a valid ciphertext or not. The corresponding scenario has been termed *reaction attack* [30]. This is exactly the situation met by Bleichenbacher when breaking the PKCS #1 v1.5 encryption scheme [5]. Finally, a decryption oracle returns the decryption of any ciphertext c, with the only restriction that it should be different from the challenge ciphertext. When access to the decryption oracle is only granted to A_1, i.e. during the first stage of the attack, before the challenge ciphertext is received, the corresponding scenario is named *indifferent chosen-ciphertext attack* (CCA1) [35]. When the attacker also receives access to the decryption oracle in the second stage, the attack is termed the *adaptive chosen-ciphertext attack* (CCA2) [50]. The security notions defined above and their logical relationships have been discussed at length in [1]. The main results are summarized in the well-known diagram shown on figure 1.

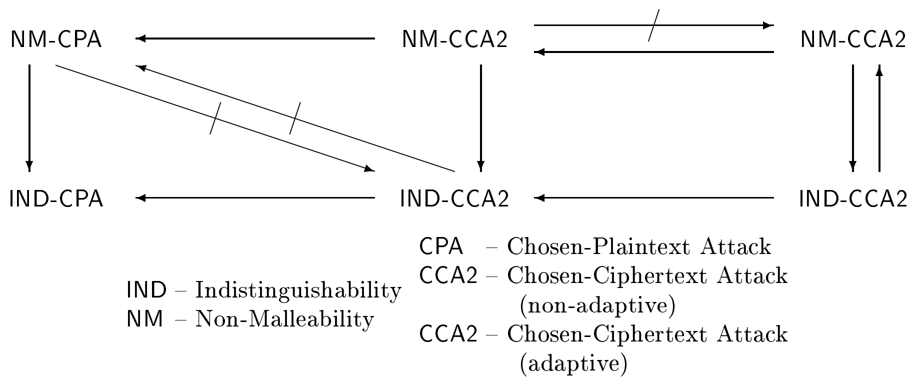

Fig. 1. Relations between security notions

Thus, under CCA2, semantic security and non-malleability are equivalent. This is the strongest security notion currently considered. We restate its definition in a more formal manner: any adversary $\mathcal{A}$ with unrestricted access to the decryption oracle $\mathcal{D}_{\mathsf{sk}}$, has negligible advantage, where the advantage is:

$$\mathsf{Adv}^{\mathsf{ind}}(\mathcal{A}^{\mathcal{D}_{\mathsf{sk}}}) = 2 \times \Pr\left[\begin{array}{l}(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathcal{K}(1^k), (m_0, m_1, s) \leftarrow A_1^{\mathcal{D}_{\mathsf{sk}}}(\mathsf{pk}), \\ b \stackrel{R}{\leftarrow} \{0,1\}, c = \mathcal{E}_{\mathsf{pk}}(m_b) : A_2^{\mathcal{D}_{\mathsf{sk}}}(m_0, m_1, s, c) = b\end{array}\right] - 1,$$

1.3 Digital Signatures

In modern terms (see [28]), a digital signature scheme consists of three algorithms $(\mathcal{K}, \Sigma, V)$:

- A *key generation algorithm* $\mathcal{K}$, which, on input 1^k, where k is the security parameter, outputs a pair $(\mathsf{pk}, \mathsf{sk})$ of matching public and private keys. Algorithm $\mathcal{K}$ is probabilistic.
- A *signing algorithm* Σ, which receives a message m and the private key sk, and outputs a signature $\sigma = \Sigma_{\mathsf{sk}}(m)$. The signing algorithm might be probabilistic.
- A *verification algorithm* V, which receives a candidate signature σ, a message m and a public key pk, and returns an answer $V_{\mathsf{pk}}(m, \sigma)$ testing whether σ is a valid signature of m with respect to pk. In general, the verification algorithm need not be probabilistic.

Attacks against signature schemes can be classified according to the goals of the adversary and to the resources that it can use. The goals are diverse:

- Disclosing the private key of the signer. It is the most drastic attack. It is termed *total break*.
- Constructing an efficient algorithm which is able to sign any message with significant probability of success. This is called *universal forgery*.
- Providing a single message/signature pair. This is called *existential forgery*.

In many cases the latter does not appear dangerous because the output message is likely to be meaningless. Nevertheless, a signature scheme, which is not existentially unforgeable, does not guarantee by itself the identity of the signer. For example, it cannot be used to certify randomly looking elements, such as keys or compressed data. Furthermore, it cannot formally guarantee the so-called non-repudiation property, since anyone may be able to produce a message with a valid signature.

In terms of resources, the setting can also vary. We focus on two specific attacks against signature schemes: the *no-message attacks* and the *known-message attacks*. In the first scenario, the attacker only knows the public key of the signer. In the second, the attacker has access to a list of valid message/signature pairs. Again, many sub-cases appear, depending on how the adversary gains knowledge. The strongest is the *adaptive chosen-message attack* (CMA), where the attacker can require the signer to sign any message of its choice, where the queries are based upon previously obtained answers. When signature generation is not deterministic, there may be several signatures corresponding to a given message. A slightly weaker security model, which we call *single-occurrence adaptive chosen-message attack* (SO-CMA), allows the adversary at most one signature query for each message. In other words the adversary cannot submit the same message twice for signature.

In chosen-message attacks, one should point out that existential forgery becomes the ability to forge a fresh message/signature pair that has not been obtained from queries asked during the attack. Again there is a subtle point here, related to the context where several signatures may correspond to a given message. We actually adopt the stronger rule that the attacker needs to forge the signature of message, whose signature was not queried.

When designing a signature scheme, one wishes to rule out existential forgeries, even under adaptive chosen-message attacks. More formally, one requires that the success probability of any adversary $\mathcal{A}$, whose running time remains below some security bound t, is negligible, where the success probability is defined by:

$$\mathsf{Succ}^{\mathsf{cma}}(\mathcal{A}) = \Pr\left[(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathcal{K}(1^k), (m, \sigma) \leftarrow \mathcal{A}^{\Sigma_{\mathsf{sk}}}(\mathsf{pk}) : V_{\mathsf{pk}}(m, \sigma) = 1\right].$$

In the above, note the superscript Σ_{sk}, indicating adaptive calls to the signing algorithm: this is consistent with the framework of relativized complexity theory, and we will use the wording *signing oracle* in this setting. When dealing with single-occurrence attacks, $\mathsf{Succ}^{\mathsf{cma}}(\mathcal{A})$ is replaced by an appropriately defined variant $\mathsf{Succ}^{\mathsf{so-cma}}(\mathcal{A})$.

1.4 The random oracle model

Ideally, one would like to establish the security of a cryptographic scheme based on the sole assumption that some widely studied mathematical problem is hard. Such problems include factoring, inverting the RSA function, and solving the discrete logarithm problem or the Diffie-Hellman problem. Unfortunately, very few schemes are know that allow such a proof (see however [17, 18]), and none is based on RSA.

Thus, the best one can hope for is a proof carried in a non-standard computational model, as proposed by Bellare and Rogaway [2], following an earlier suggestion by Fiat and Shamir [21]. In this model, called the random oracle model, concrete objects such that hash functions [37] are treated as random objects. This allows to carry through the usual reduction arguments to the context of relativized computations, where the hash function is treated as an oracle returning a random answer for each new query. A reduction still uses an adversary as a subroutine of a program that contradicts a mathematical assumption, such as the assumption that RSA is one-way. However, probabilities are taken not only over coin tosses but also over the random oracle.

Of course, the significance of proofs carried in the random oracle is debatable. Firstly, hash functions are deterministic and therefore do not return random answers. Along those lines, Canetti *et al.* [11] gave an example of a signature scheme which is secure in the random oracle model, but insecure under any instantiation of the random oracle. Secondly, proofs in the random oracle model cannot easily receive a quantitative interpretation. One would like to derive concrete estimates - in terms of key sizes - from the proof: if a reduction is efficient, the security "loss" is small and the existence of an efficient adversary leads to an algorithm for solving the underlying mathematical problem, which is almost as efficient. Thus, key sizes that outreach the performances of the known algorithms to break the underlying problem can be used for the scheme as well.

Despite these restrictions, the random oracle model has proved extremely useful to analyze many encryption and signature schemes [2–4, 48, 43, 49, 9, 58, 15, 6, 22, 31, 59]. It clearly provides an overall guarantee that a scheme is not flawed, based on the intuition that an attacker would be forced to use the hash function in a non generic way.

Recently, several authors have proposed to use yet another model to argue in favour of the security of cryptographic schemes, that could not be tackled by the random oracle model. This is the so-called *black-box* group model, or *generic* model [56, 10, 40]. In particular, paper [10] considered the security of ECDSA in this model. Generic algorithms had been earlier introduced by Nechaev and Shoup [41, 57], to encompass group algorithms that do not exploit any special property of the encodings of group elements other than the property that each group element is encoded by a unique string. We will not further comment on this method since it is not used in the examples that we wish to cover.

2 A Case Study: the OAEP Formatting Scheme

As already noted, the famous RSA cryptosystem has been proposed by Rivest, Shamir and Adleman [51]. The key generation algorithm of RSA chooses two large primes p, q of equal size and issues the so-called modulus $n = pq$. The sizes of p, q are set in such a way that the binary length $|n|$ of n equals the security parameter k. Additionally, en exponent e, relatively prime to $\varphi(n) = (p-1)(q-1)$ is chosen, so that the public key is the pair (n, e). The private key d is the inverse of e modulo $\varphi(n)$. Variants allow the use of more than two prime factors.

Encryption and decryption are defined as follows:

$$\mathcal{E}_{n,e}(m) = m^e \bmod n \qquad \mathcal{D}_{n,d}(c) = c^d \bmod n.$$

Note that both operations are deterministic and are mutually inverse to each other. Thus, the RSA encryption function is a permutation. It is termed a *trapdoor permutation* since decryption can only be applied given the private key.

The basic security assumption on which the RSA cryptosystem relies is its *one-wayness* (OW): using only public data, an attacker cannot recover the plaintext corresponding to a given ciphertext. In the general formal setting provided above, an encryption scheme is one-way if the success probability of any adversary $\mathcal{A}$ attempting to invert $\mathcal{E}$ (without the help of the private key), is negligible, i.e. asymptotically smaller than the inverse of any polynomial function of the security parameter. Probabilities are taken over the message space $\mathcal{M}$ and the

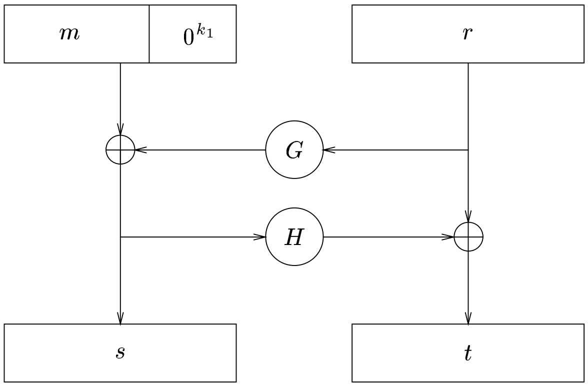

Fig. 2. Optimal Asymmetric Encryption Padding

randoin coins Ω. These include both the random coins r used for the encryption scheme, and the internal random coins of the adversary. In symbols:

$$\mathsf{Succ}^{\mathsf{ow}}(\mathcal{A}) = \Pr[(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathcal{K}(1^k), m \stackrel{R}{\leftarrow} \mathcal{M} : \mathcal{A}(\mathsf{pk}, \mathcal{E}_{\mathsf{pk}}(m)) = m].$$

Clearly, the factorization of n allows to invert the RSA encryption function, since d can be computed from p and q. It is unknown whether the converse is true, i.e. whether factoring and inverting RSA are computationnally equivalent. There are indications that it might not be true (see [7]). Thus, the assumption that RSA is one-way might be a stronger assumption than the hardness of factoring. Still, it is a widely believed assumption and the only method to assess the strength of RSA is to check whether the size of the modulus n outreaches the current performances of the various factoring algorithms.

As already stated, the naive RSA algorithm defined in the previous section cannot be used as it stands. More precisely RSA by itself cannot provide a secure encryption scheme for any of the security notions considered in the previous section: semantic security fails because encryption is deterministic and non-malleability cannot hold due to the homomorphic property:

$$\mathcal{E}_{n,e}(m_1) \cdot \mathcal{E}_{n,e}(m_2) = \mathcal{E}_{n,e}(m_1 m_2 \bmod n) \bmod n.$$

Therefore, any RSA-based cryptosystems has to use a padding or encoding method before applying the RSA primitive.

The OAEP padding scheme (optimal asymmetric encryption padding) was proposed by Bellare and Rogaway [3] in 1994. It is depicted on figure 2 . For a long time it was implicitly believed that OAEP achieved CCA2 security for any trapdoor function, based on a proof in the random oracle model, relying on the one-wayness of the permutation.

However, Victor Shoup [58] recently showed, by means of a subtle counter-example in a relativized model of computation, that it is quite unlikely that such a security proof exists, at least under the sole one-wayness of the permutation. He also proposed a modified version of OAEP, called OAEP+, which can be proven secure, under the one-wayness of the permutation. What went wrong here is that the proof of Bellare and Rogaway only applied in the restricted attack setting where the adversary can query the decryption oracle before it receives the challenge ciphertext c, referred above as CCA1. It did not mean at all that the security proof provided by Bellare and Rogaway was incorrect, since they never claimed to cover CCA2 security. It did not mean either that RSA-OAEP was flawed. It only meant that a new proof was needed.

Surprisingly, the repaired proof appeared shortly afterwards in [22]. Albeit based on the same methodology, it was significantly different, using additional algebraic tools, notably the reduction of two-dimensional lattices in the style of [39], which did not appear in the earlier proof. Thus, the multiplicative properties of RSA, which motivated the quest for formatting schemes, ultimately helped for the security proof. It should also be noted that alternative formatting schemes with a more direct security proof such as OAEP+, have been recently designed. However, OAEP is a widely used standard [52] and it is unclear whether it will be replaced by these challengers.

3 Another Case Study: The ESIGN Signature Scheme

Soon after the appearance of the celebrated RSA cryptosystem [51], a lot of effort was devoted to finding alternative schemes. In the area of signature, researchers faced the challenge of reducing the computing effort needed from the signer, since it is well known that RSA requires a full-size modular exponentiation. Among the potential candidates to answer this challenge is the ESIGN signature scheme, that has been proposed in the early nineties (see [42]). While RSA generates signatures by computing an e-th root of a hash value, ESIGN only requests to find an element whose e-th power is close enough to the hash value.

A precise specification of ESIGN appears in [43]. We refer to this papers for details. The key generation algorithm chooses two large primes p, q of equal size k and computes the modulus $n = p^2q$. The sizes of p, q are set in such a way that the binary length $|n|$ of n equals $3k$. Additionally, an exponent $e > 4$ is chosen.

Signature generation is performed as follows, using a hash function $\mathcal{H}$, outputting strings of length $k-1$:

1. Pick at random r in $\mathbb{Z}^{\star}_{pq}$.
2. Convert $(0\|\mathcal{H}(m)\|0^{2k})$ into an integer y.
3. Compute and output an element s in the interval $I_k(y) = \{u | y \leq u < y + 2^{2k-1}\}$.

Signature verification converts integer $s^e \bmod n$ into a bit string S of length $3k$ and checks that $[S]^k = 0\|\mathcal{H}(m)$, where $[S]^k$ denotes the k leading bits of S.

The basic paradigm of ESIGN is that the arithmetical progression $r^e \bmod n + tpq$ consists of e-th powers of easily computed integers: the third step of signature generation adjusts t so as to fall into a prescribed interval $I_k(y)$ of length

2^{2k-1}. This allows a very efficient way to sign, with a computing time essentially equivalent to a single exponentiation to the e-th power. This is especially attractive when e is small, and in particular a small power of two.

Thus, the mathematical assumption underlying ESIGN is that, given an element y of $\mathbb{Z}_n^\star$, it is hard to find x such that $x^e \bmod n$ lies in the interval $I_k(y) = \{u | y \leq u < y + 2^{2k-1}\}$, where the bit-size of n is $3k$. This is called the approximate e-th root problem (AERP) in [43]. In a more formal setting, denote by $\mathsf{Succ}^{\mathsf{aerp}}(\tau, k)$ the probability for any adversary $\mathcal{A}$ to find an element whose e-th power lies in the prescribed interval, within time τ, in symbols:

$$\mathsf{Succ}^{\mathsf{aerp}}(\tau, k) = \Pr[(n, e) \leftarrow \mathcal{K}(1^k), y \leftarrow \mathbb{Z}_N, x \leftarrow \mathcal{A}(N, e, y) : (x^e \bmod n) \in I_k(y)].$$

then, the hardness assumption is the statement that, for large enough moduli, this probability is extremely small. Variants of the above can be considered, where the length of the interval is replaced by 2^{2k} or 2^{2k+1}.

Of course, the factorization of n allows to solve the AERP problem. It is unknown whether the converse is true, i.e. whether AERP and inverting RSA are computationally equivalent. As most proposed cryptosystems, ESIGN has attracted cryptanalytic effort. Papers [8, 60] described several attacks against the underlying problem, for $e = 2, 3$. Still, It is fair to say that there is no known attack against AERP when e is ≥ 4.

Recently, in connection with several standardization efforts such as IEEE P1363, Cryptrec and NESSIE, an effort was made to lay ESIGN on firm foundations, using the methodology of provable security. A security proof in the random oracle model, along the lines of [4], formally relating the security of ESIGN with the AERP problem, appeared in [43]. However, several unexpected difficulties were found. Firstly, it was observed in [59] that the proof from [43] holds in a more restricted model of security than claimed: this model, termed single occurrence chosen message attack $\mathsf{SO\text{-}CMA}$ in section 1.3 above, is very similar to the usual chosen message attack scenario but does not allow the adversary to submit the same message twice for signature. This observation does not endanger the scheme in any way, and furthermore, it is quite easy to restore the usual CMA security, as suggested in [29]. Still, it shows that the methodology of security proofs should unambiguously define the attack model. Secondly, it was found that the proof needs the additional assumption that e is prime to $\varphi(n)$, thus exluding some very attractive parameter choices, notably powers of two advocated in the original proposal. The difficulty here comes from the simulation of the random oracle. As far as we know, this is the only example where this part is not straightforward, and the underlying difficulty may be easily overlooked. In other words, it may appear obvious that picking x at random and suitably truncating $x^e \bmod n$ yields an almost uniform distribution. However, it is not, at least when e is not prime to $\varphi(n)$ since it relies on the distribution of e-th powers, which is not completely understood from a mathematical point of view. Thus, removing the restriction that e is prime to $\varphi(n)$ is not a side issue. Besides allowing to make e a power of two in order to take full advantage of computational efficiency of ESIGN, it once again shows that security proofs have many subtleties.

In an unpublished paper [46], Tatsuaki Okamoto and the author proved that the set of e-th power modulo an RSA modulus n, is almost uniformly distributed on any large enough interval. The proof borrows from analytic number theory and uses Dirichlet characters and the Polya-Vinogradov inequality [47, 62]. This yield concrete estimates that are enough to complete the security proof of ESIGN.

4 Conclusions

There are several lessons to learn form the above. Firstly, the methodology of provable security has become unavoidable in designing, analyzing and evaluating new schemes. Despite its limitations, the random oracle model needs to be used in order to cover schemes that remain attractive for the practioner. On the other hand, due to these limitations, the estimates that are drawn from proofs in the random oracle model should be interpreted with care: considering SHA1 as a random oracle is a methodological step, averaging on random oracles to derive security margins is a further step.

Secondly, cryptography and provable security proceed at their pace and meet many hidden subtleties. Twenty-five centuries were needed before the discovery of public key cryptography by Diffie and Hellman. It took twenty-five more years to understand how RSA could be correctly practiced. No cryptographic algorithm can be designed and validated in twenty-five minutes, twenty-five hours, or twenty-five days, not even twenty-five months.

5 Acknowledgements

The present paper describes the author's view of provable security. No one else should be held responsible for any inaccuracy it includes. The author is grateful to the program committee and of Eurocypt 2003 and to the program chair for allowing him to present these views. The author also wishes to mention his debt towards many colleagues that developped the subject, by whom he was influenced, or with whom he had discussions. To name a few, Mihir Bellare, Dan Boneh, Don Coppersmith, Ron Cramer, Shafi Goldwasser, Whit Diffie, Silvio Micali, David Naccache, Moni Naor, Tatsuaki Okamoto, Ron Rivest, Phil Rogaway, Claus Schnorr, Adi Shamir, Victor Shoup, Moti Yung. He also wants to express his gratitude to his former PhD students whose contributions helped building his own view: Olivier Baudron, Philippe Béguin, Emmanuel Bresson, Florent Chabaud, Jean-Sébastien Coron, Jean-Bernard Fischer, Louis Granboulan, Antoine Joux, David Mraïhi, Phong Nguyen, David Pointcheval, Thomas Pornin, Guillaume Poupard, Christophe Tymen, Serge Vaudenay. Finally, many of the author's ideas on provable security and its relation to cryptographic standards stemmed from discussions with the NESSIE team and from the result of evaluations requested by the Japanese Cryptrec program. This author wishes to thank both NESSIE and Cryptrec.

References

1. M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway. Relations among Notions of Security for Public-Key Encryption Schemes. In *Crypto '98*, Lecture Notes in Computer Science 1462, Springer-Verlag, Berlin, 1998, 26–45.
2. M. Bellare and P. Rogaway. Random Oracles Are Practical: a Paradigm for Designing Efficient Protocols. In *Proc. of the 1st CCS*, ACM Press, New York, 1993, 62–73.
3. M. Bellare and P. Rogaway. Optimal Asymmetric Encryption – How to Encrypt with RSA. In *Eurocrypt '94*, Lecture Notes in Computer Science 950, Springer-Verlag, Berlin, 1995, 92–111.
4. M. Bellare and P. Rogaway. The Exact Security of Digital Signatures – How to Sign with RSA and Rabin. In *Eurocrypt '96*, Lecture Notes in Computer Science 1070, Springer-Verlag, Berlin, 1996, 399–416.
5. D. Bleichenbacher. A Chosen Ciphertext Attack against Protocols based on the RSA Encryption Standard PKCS #1. In *Crypto '98*, Lecture Notes in Computer Science 1462, Springer-Verlag, Berlin, 1998, 1–12.
6. D. Boneh. Simplified OAEP for the RSA and Rabin Functions. In *Crypto '2001*, Lecture Notes in Computer Science 2139, Springer-Verlag, Berlin, 2001, 275–291.
7. D. Boneh and R. Venkatesan. Breaking RSA may not be equivalent to factoring. In *Eurocrypt '98*, Lecture Notes in Computer Science 1402, Springer-Verlag, Berlin, 1998, 59–71,.
8. E. Brickell and J. M. DeLaurentis. An Attack on a Signature Scheme proposed by Okamoto and Shiraishi. In *Crypto '85*, Lecture Notes in Computer Science 218, 28–32, Springer-Verlag, Berlin, 1986, 28–32.
9. E. Brickell, D. Pointcheval, S. Vaudenay, and M. Yung. Design Validations for Discrete Logarithm Based Signature Schemes. In *PKC '2000*, Lecture Notes in Computer Science 1751, Springer-Verlag, Berlin, 2000, 276–292 .
10. D. R. L. Brown. The Exact Security of ECDSA, January 2001. Available from `http://grouper.ieee.org/groups/1363/`.
11. R. Canetti, O. Goldreich, and S. Halevi. The Random Oracles Methodology, Revisited. In *Proc. of the 30th STOC*, ACM Press, New York, 1998, 209–218.
12. D. Coppersmith. Finding a Small Root of a Univariate Modular Equation. In *Eurocrypt '96*, Lecture Notes in Computer Science 1070, Springer-Verlag, Berlin, 1996, 155–165.
13. D. Coppersmith, S. Halevi and C. Jutla. ISO 9796-1 and the New Forgery Strategy. Presented at the Rump session of *Crypto '99*.
14. J.-S. Coron. On the Exact Security of Full-Domain-Hash. In *Crypto '2000*, Lecture Notes in Computer Science 1880, Springer-Verlag, Berlin, 2000, 229–235.
15. J.-S. Coron. Optimal Security Proofs for PSS and other Signature Schemes. In *Eurocrypt '2002* Lecture Notes in Computer Science 2332, Springer-Verlag, Berlin, 2002, 272–287. Also appeared in the Cryptology ePrint Archive 2001/062, June 2001, available from `http://eprint.iacr.org/`, 2001.
16. J.-S. Coron, D. Naccache and J. P. Stern. On the Security of RSA Padding. In *Crypto '99*, Lecture Notes in Computer Science 1666, Springer-Verlag, Berlin, 1999, 1–18.
17. R. Cramer and V. Shoup. A Practical Public key Cryptosystem Provably Secure Against Adaptive Chosen Ciphertext Attacks. In *Crypto'98*, Lecture Notes in Computer Science 1462, 1998, 13–25.

18. R. Cramer and V. Shoup. Universal Hash Proofs and a Paradigm for Adaptive Chosen Ciphertext Secure Public Key Encryption. In *Eurocrypt'2002*, Lecture Notes in Computer Science 2332, 45–64.
19. W. Diffie and M.E. Hellman. New Directions in Cryptography, *IEEE Transactions on Information Theory*, v. IT-22, 6, Nov 1976, 644–654.
20. D. Dolev, C. Dwork, and M. Naor. Non-Malleable Cryptography. *SIAM Journal on Computing*, 30(2), 2000, 391–437 .
21. A. Fiat and A.Shamir. How to Prove Yourself: Practical Solutions of Identification and Signature Problems. In *Crypto '86*, Lecture Notes in Computer Science 263, Springer-Verlag, Berlin, 1987, 186–194.
22. E. Fujisaki, T. Okamoto, D. Pointcheval, and J. Stern. RSA–OAEP is Secure under the RSA Assumption. In *Crypto '2001*, Lecture Notes in Computer Science 2139, Springer-Verlag, Berlin, 2001, 260–274. Also appeared in the Cryptology ePrint Archive 2000/061, November 2000, available from `http://eprint.iacr.org/`.
23. T. El Gamal. A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. *IEEE Transactions on Information Theory*, IT–31(4), 1985, 469–472.
24. M. Girault, P. Toffin and B. Vallée. Computation of Approximate L-th Roots Modulo n and Application to Cryptography. In *Crypto '88*, Lecture Notes in Computer Science 403, Springer-Verlag, Berlin, 1989, 100–118.
25. S. Goldwasser and S. Micali. Probabilistic Encryption. *Journal of Computer and System Sciences*, 28, 1984, 270–299.
26. S. Goldwasser, S. Micali, and C. Rackoff. The Knowledge Complexity of Interactive Proof Systems. In *Proc. of the 17th STOC*, ACM Press, New York, 1985, 291–304.
27. S. Goldwasser, S. Micali, and R. Rivest. A "Paradoxical" Solution to the Signature Problem. In *Proc. of the 25th FOCS*, IEEE, New York, 1984, 441–448.
28. S. Goldwasser, S. Micali, and R. Rivest. A Digital Signature Scheme Secure Against Adaptative Chosen-Message Attacks. *SIAM Journal of Computing*, 17(2), 1988, 281–308.
29. L. Granboulan. How to repair ESIGN. NESSIE internal document, May 2002. See `http://www.cryptonessie.org`, Document NES/DOC/ENS/WP5/019.
30. C. Hall, I. Goldberg, and B. Schneier. Reaction Attacks Against Several Public-Key Cryptosystems. In *Proc. of ICICS'99*, Lecture Notes in Computer Science, Springer-Verlag, 1999, 2–12.
31. J. Jonsson. Security Proofs for RSA–PSS and Its Variants. Cryptology ePrint Archive 2001/053, June 2001. Available from `http://eprint.iacr.org/`.
32. IEEE P1363. Standard Specifications for Public Key Cryptography, August 1998. Available from `http://grouper.ieee.org/groups/1363/`.
33. B. Kaliski and J. Staddon. IETF RFC 2437: PKCS #1: RSA Cryptography Specifications Version 2.0. October 1998.
34. J. Manger. A Chosen Ciphertext Attack on RSA Optimal Asymmetric Encryption Padding (OAEP) as Standardized in PKCS #1. In *Crypto '2001*, Lecture Notes in Computer Science 2139, Springer-Verlag, Berlin, 2001, 230–238.
35. M. Naor and M. Yung. Public-Key Cryptosystems Provably Secure against Chosen Ciphertext Attacks. In *Proc. of the 22nd STOC*, ACM Press, New York, 1990, 427–437.
36. NIST. Digital Signature Standard (DSS). Federal Information Processing Standards Publication 186, November 1994.
37. NIST. Secure Hash Standard (SHS). Federal Information Processing Standards Publication 180–1, April 1995.

38. NIST. Digital Signature Standard (DSS). Federal Information Processing Standards Publication 186–2, January 2000.
39. A. K. Lenstra, H. W. Lenstra and L. Lovász. Factoring Polynomials with Rational Coefficients, *Mathematische Ann.*, 261, 1982, 513–534.
40. D. Naccache, D. Pointcheval, and J. Stern. Twin Signatures: an Alternative to the Hash-and-Sign Paradigm. In *Proc. of the 8th CCS*, ACM Press, New York, 2001 20–27.
41. V. I. Nechaev. Complexity of a Determinate Algorithm for the Discrete Logarithm. *Mathematical Notes*, 55(2), 1994, 165–172.
42. T. Okamoto. A Fast Signature Scheme Based on Congruential Polynomial Operations. *IEEE Transactions on Information Theory*, IT–36 (1), 1990, 47–53.
43. T. Okamoto, E. Fujisaki and H. Morita. TSH-ESIGN: Efficient Digital Signature Scheme Using Trisection Size Hash, Submission to P1363a, 1998.
44. T. Okamoto and D. Pointcheval. REACT: Rapid Enhanced-security Asymmetric Cryptosystem Transform. In *CT – RSA '2001*, Lecture Notes in Computer Science 2020, Springer-Verlag, Berlin, 2001, 159–175.
45. T. Okamoto and A. Shiraishi. A Fast Signature Scheme Based on Quadratic Inequalities. Proc. of the ACM Symp. Security and Privacy, ACM Press, 1985, 123–132.
46. T. Okamoto and J. Stern. Almost uniform density of power residues and the security proof of ESIGN. Unpublished manuscript.
47. G. Pólya. Über die Verteilung des quadratischen Reste und Nichtreste. *Göttinger Nachtrichten* (1918), 21–26.
48. D. Pointcheval and J. Stern. Security Proofs for Signature Schemes. In *Eurocrypt '96*, Lecture Notes in Computer Science 1070, Springer-Verlag, Berlin, 1996, 387–398.
49. D. Pointcheval and J. Stern. Security Arguments for Digital Signatures and Blind Signatures. *Journal of Cryptology*, 13(3), 2000, 361–396.
50. C. Rackoff and D. R. Simon. Non-Interactive Zero-Knowledge Proof of Knowledge and Chosen Ciphertext Attack. In *Crypto '91*, Lecture Notes in Computer Science 576, Springer-Verlag, Berlin, 1992, 433–444.
51. R. Rivest, A. Shamir, and L. Adleman. A Method for Obtaining Digital Signatures and Public Key Cryptosystems. *Communications of the ACM*, 21(2), 1978, 120–126.
52. RSA Laboratories. PKCS # 1 Version 1.5: RSA Cryptography Standard. November 1993, `http://www.rsa.com/rsalabs/pubs/PKCS/`.
53. RSA Laboratories. PKCS # 1 Version 2.1: RSA Cryptography Standard. Draft, January 2001, `http://www.rsa.com/rsalabs/pubs/PKCS/`.
54. C. P. Schnorr. Efficient Identification and Signatures for Smart Cards. In *Crypto '89*, Lecture Notes in Computer Science 435, Springer-Verlag, Berlin, 1990, 235–251.
55. C. P. Schnorr. Efficient Signature Generation by Smart Cards. *Journal of Cryptology*, 4(3), 1991, 161–174.
56. C. P. Schnorr and M. Jakobsson. Security of Signed ElGamal Encryption. In *Asiacrypt '2000*, Lecture Notes in Computer Science 1976, Springer-Verlag, Berlin, 2000, 458–469.
57. V. Shoup. Lower Bounds for Discrete Logarithms and Related Problems. In *Eurocrypt '97*, Lecture Notes in Computer Science 1233, Springer-Verlag, Berlin, 1997, 256–266.

58. V. Shoup. OAEP Reconsidered. In *Crypto '2001*, Lecture Notes in Computer Science 2139, Springer-Verlag, Berlin, 2001, 239–259. Also appeared in the Cryptology ePrint Archive 2000/060, November 2000, available from `http://eprint.iacr.org/`.
59. J. Stern, D. Pointcheval, J. Malone-Lee, and N. Smart. Flaws in Applying Proof Methodologies to Signature Schemes. In *Crypto '02*, Lecture Notes in Computer Science 2442, Springer-Verlag, Berlin, 2002, 93–110.
60. B. Vallée, M. Girault, and P. Toffin. How to break Okamoto's Cryptosystem by Reducing Lattice Bases. In *Eurocrypt '88*, Lecture Notes in Computer Science 330, Springer-Verlag, Berlin, 1988, 281–292.
61. B. Vallée, M. Girault and P. Toffin. How to Guess ℓth Roots Modulo n by Reducing Lattice Bases. In *AAECC-6*, Lecture Notes in Computer Science 357, Springer-Verlag, Berlin, 1988, 427–442.
62. I.M. Vinogradov. Sur la distributions des résidus et des non-résidus des puissances. *J. Phys.-Math. Soc. Perm.* 1 (1918), 94–96.