The Gap-Problems: A New Class of Problems for the Security of Cryptographic Schemes

Tatsuaki Okamoto[1] and David Pointcheval[2]

[1] NTT Labs, 1-1 Hikarinooka, Yokosuka-shi 239-0847 Japan.
okamoto@isl.ntt.co.jp.
[2] Dépt d'Informatique, ENS – CNRS, 45 rue d'Ulm, 75230 Paris Cedex 05, France.
David.Pointcheval@ens.fr – http://www.di.ens.fr/~pointche.

Abstract. This paper introduces a novel class of computational problems, the *gap problems*, which can be considered as a dual to the class of the *decision problems*. We show the relationship among inverting problems, decision problems and gap problems. These problems find a nice and rich practical instantiation with the Diffie-Hellman problems.
Then, we see how the gap problems find natural applications in cryptography, namely for proving the security of very efficient schemes, but also for solving a more than 10-year old open security problem: the Chaum's undeniable signature.

1 Introduction

1.1 Motivation

It is very important to prove the security of a cryptographic scheme under a reasonable computational assumption. A typical reasonable computational assumption is the intractability of an inverting problem such as factoring a composite number, inverting the RSA function [33], computing the discrete logarithm problem, and computing the Diffie-Hellman problem [12]. Here, an inverting problem is, given a problem, x, and relation f, to find its solution, y, such that $f(x, y) = 1$.

Another type of reasonable computational assumptions is the intractability of a decision problem such as the decision Diffie-Hellman problem. Such a decision problem is especially useful to prove the semantical security of a public-key encryption (e.g., El Gamal and Cramer-Shoup encryption schemes [13,11]). Although we have several types of decision problems, a typical decision problem is, given (x, y) and f, to decide whether the pair (x, y) satisfies $f(x, y) = 1$ or not. Another typical example of decision problems is, given x and f, to decide a hard core bit, $H(y)$, of x with $f(x, y) = 1$.

After having studied some open problems about the security of several primitive cryptographic schemes in which we have not found any flaw, we have realized that the existing computational assumptions (or primitive problems) are not sufficient to prove the security of these schemes. For example, Chaum's undeniable signature scheme [9,7] based on the discrete logarithm is the most typical scheme to realize an undeniable signature scheme and is often used for cryptographic

K. Kim (Ed.): PKC 2001, LNCS 1992, pp. 104–118, 2001.
© Springer-Verlag Berlin Heidelberg 2001

protocols (e.g., Brands' restrictive blind signatures [6,5]), however, we cannot prove the security of Chaum's undeniable signature scheme under any existing computational assumption. That is, we have realized that a new family of computational assumptions (or problems) are necessary to prove several important cryptographic schemes.

1.2 Achievement

To prove the security of these primitive cryptographic schemes, this paper introduces a new family of problems we called the *gap problems*. Intuitively speaking, a gap problem is to solve an inverting problem with the help of the oracle of a related decision problem. For example, a gap problem of f is, given problem x and relation f, to find y satisfying $f(x, y) = 1$, with the help of the oracle of, given question (x', y'), answering whether $f(x', y') = 1$ or not.

Indeed, in some situations, an adversary has to break a specific computational problem to make fail the security, while having a natural access to an oracle which answers a yes/no query, and therefore leaking one bit. For example, in an undeniable signature, an adversary tries to forge a signature (i.e., solve an inverting problem) with being allowed to ask a signer (i.e., oracle) of whether a pair of signature s and message m is valid or not.

We show that the class of gap problems is dual to the class of decision problems. We then prove Chaum's undeniable scheme is secure under the assumption of the related gap problem. Here note that it has been open for more than 10 years to prove the security of Chaum's undeniable scheme.

1.3 Outline of the Paper

This paper has the following organization. First, we formally define this new family of gap-problems, in a general setting and for the particular situation of the random self-reducible problems. Then, we present some interesting examples, derived from the classical problems used in cryptography. Finally, we prove that the security of some very old protocols (undeniable signatures and designated confirmer signatures) is equivalent to some gap problems, while it has been an open problem for a long time.

2 Gap Problems

This section is devoted to the presentation of this new class of problems which can be seen as the dual to the decisional problems. Some theoretical results are proposed together with some practical examples.

2.1 Definitions

Let $f : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}$ be any relation. The inverting problem of f is the classical computational version, while we introduce a generalization of the

decision problem, by the R-decision problem of f, for any relation

$$R : \{0,1\}^* \times \{0,1\}^* \times \{0,1\}^* \to \{0,1\},$$

- the *inverting problem* of f is, given x, to compute any y such as $f(x,y) = 1$ if it exists, or to answer Fail.
- the *R-decision problem* of f is, given (x,y), to decide whether $R(f,x,y) = 1$ or not. Here y may be the null string, $\perp$.

Let us see some examples for the relation, R_1, R_2, R_3, R_4:

- $R_1(f,x,y) = 1$ iff $f(x,y) = 1$, which formalizes the classical version of decision problems (*cf.* the Decision Diffie-Hellman problem [4,26]).
- $R_2(f,x,\perp) = 1$ iff there exists any z such that $f(x,z) = 1$, which simply answers whether the inverting problem has a solution or not.
- $R_3(f,x,\perp) = 1$ iff z is even, when z such that $f(x,z) = 1$ is uniquely defined. This latter example models the least-significant bit of the pre-image, which is used in many hard-core bit problems [1,14].
- $R_4(f,x,\perp) = 1$ iff all the z such that $f(x,z) = 1$ are even.

It is often the case that the *inverting problem* is strictly stronger than the *R-decision problem*, namely for all the classical examples we have for cryptographic purpose. However, it is not always the case, and the R-decision problem can even be strictly stronger than the inverting one (the latter R_4-relation above gives the taste of such an example). In this section, we define the *R-gap problem* which deals with the gap of difficulty between these problems.

Definition 1 (Gap Problem). *The R-gap problem of f is to solve the inverting problem of f with the help of the oracle of the R-decision problem of f.*

2.2 Winning Probabilities

For a computational problem (the inverting or the gap problem), the winning probability is the probability of finding the correct solution on input an instance I and a random tape r. While for a decision problem, the winning probability expresses the advantage the algorithm has in guessing the output bit of the relation R above flipping a coin, on input an instance I and a random tape r.

Computational Problems. For an algorithm $\mathcal{A}$ against a computational problem P, we define winning probabilities as follows:

$$\begin{array}{ll} \text{for any instance } I \in P, & \mathsf{Win}^P_{\mathcal{A}}(I) = \Pr_r[\mathcal{A}(I;r) \text{ wins}], \\ \text{in general,} & \mathsf{Win}^P_{\mathcal{A}} = \Pr_{I,r}[\mathcal{A}(I;r) \text{ wins}]. \end{array}$$

Decision Problems. For an algorithm $\mathcal{A}$ against a decision problem P, we define winning probabilities as follows, which consider the advantage an adversary gains above flipping a coin:

$$\begin{array}{ll} \text{for any instance } I \in P, & \mathsf{Win}^P_{\mathcal{A}}(I) = 2 \times \Pr_r[\mathcal{A}(I;r) \text{ wins}] - 1, \\ \text{in general,} & \mathsf{Win}^P_{\mathcal{A}} = 2 \times \Pr_{I,r}[\mathcal{A}(I;r) \text{ wins}] - 1. \end{array}$$

2.3 Tractability

Let us now define some specific notions of tractability which will be of great interest in the following:

- a problem P is *tractable* if there exists a probabilistic polynomial time Turing machine $\mathcal{A}$ which can win with non-negligible probability, over the instances and the internal coins of $\mathcal{A}$.

$$\exists \mathcal{A}, \mathsf{Win}_{\mathcal{A}}^{P} \text{ is non-negligible.}$$

- a problem P is *strongly tractable* if there exists a probabilistic polynomial time Turing machine $\mathcal{A}$ which can win, for any instance I, with overwhelming probability, over the internal coins of $\mathcal{A}$.

$$\exists \mathcal{A}, \forall I \in P, \mathsf{Win}_{\mathcal{A}}^{P}(I) \text{ is overwhelming.}$$

Therefore, we have the negation:

- a problem P is *intractable* if it is not *tractable*
- a problem P is *weakly intractable* if it is not *strongly tractable*.

Finally, to compare the difficulty of problems, we use the notion of polynomial time reductions:

- a problem P is *reducible* to problem P' if there exists a probabilistic polynomial time oracle Turing machine $\mathcal{A}^{P'}$ (with an oracle of the problem P') that wins P with non-negligible probability.
- a problem P is *strongly reducible* to problem P' if there exists a probabilistic polynomial time oracle Turing machine $\mathcal{A}^{P'}$ (with an oracle of the problem P') that wins any instance I of P with overwhelming probability.

We can easily obtain the following proposition,

Proposition 2. *Let f and R be any relations.*

- *If the R-gap problem of f is tractable (resp. strongly tractable), the inverting problem of f is reducible (resp. strongly reducible) to the R-decision problem of f.*
- *If the R-decision problem of f is strongly tractable, the inverting problem of f is reducible to the R-gap problem of f.*

Proof. The first claim directly comes from the definition of the gap problem and the definitions of tractability and reducibility. Let us consider the second claim, with a probabilistic polynomial time Turing machine $\mathcal{B}$ that solves the R-decision problem of f, with overwhelming probability. Let us also assume that we have a probabilistic polynomial time oracle Turing machine $\mathcal{A}^{D}$ that solves the inverting problem of f with the help of a R-decision oracle D. Since $\mathcal{B}$ solves any instance of the R-decision problem with overwhelming probability, it perfectly simulates the D oracle, after polynomially many queries, with non-negligible probability. For this non-negligible fraction of cases, the machine $\mathcal{A}$ can invert f. But one has to remark that after polynomially many calls to $\mathcal{B}$, the success probability cannot be proven more than non-negligible, hence the classical reducibility, and not the *strong* one. □

This proposition implies a duality between the gap and the decision problems.

2.4 The Random Self-Reducible Problems

Definition 3 (Random Self-Reducibility). *A problem $\mathcal{P} : P \mapsto S$, where P defines the set of the instances and S the set of the possible solutions ($S = \{0, 1\}$ for a decision problem) is said* random self-reducible *(see figure 1) if there exist two probabilistic polynomial time Turing machines $A : P \mapsto P$ and $B : S \mapsto S$, with random tape $\omega \in \Omega$, such that*

- *for any $I \in P$, $A(I; \omega)$ is uniformly distributed in P while ω is randomly drawn from Ω,*
- *for any $s' \in S$, $B(s'; \omega)$ is uniformly distributed in S while ω is randomly drawn from Ω.*
- *for any instance $I \in P$ and any random tape $\omega \in \Omega$, if $I' = A(I; \omega)$ and s' is a solution to I', then $s = B(s'; \omega)$ is a solution to I.*

For such problems, the weak intractability is equivalent to the classical intractability.

Proposition 4. *Let P be any random self-reducible problem:*

- *this problem P is strongly tractable if and only if it is tractable;*
- *this problem P is intractable if and only if it is weakly intractable.*

Proof. It is clear that both claims are equivalent, and furthermore in each, one of the directions is trivial, since any strongly tractable problem is *a fortiori* tractable. For the remaining direction, one can simply use Shoup's construction [35] to obtain the result. □

Corollary 5. *Let f and R be any relations. Let us assume that both the inverting problem of f and the R-decision problem of f are random self-reducible.*

- *If the R-gap problem of f is tractable, the inverting problem of f is reducible to the R-decision problem of f.*
- *If the R-decision problem of f is tractable, the inverting problem of f is reducible to the R-gap problem of f.*

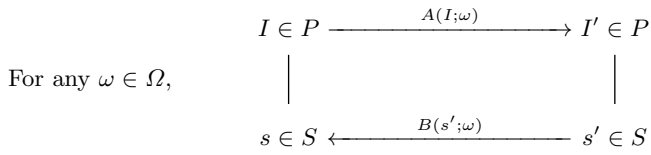

Fig. 1. Random Self-Reducible Problems

Proof. To complete the proof, one just has to remark that if the inverting problem is random self-reducible, then the gap problem is so too. □

Remark 6. Almost all the classical problems used in cryptography are *random self-reducible*: RSA [33] for fixed modulus n and exponent e, the discrete logarithm and the Diffie-Hellman problems [12] for a fixed basis of prime order, or even over the bases if the underlying group is a cyclic group of prime order, etc.

3 Examples of Gap Problems

Let us review some of these classical problems, with their gap variations. Let us begin with the most famous problem used in cryptography, the RSA problem.

3.1 The RSA Problems

Let us consider $n = pq$ and e relatively prime with $\varphi(n)$, the totient function of n. We have the classical *Inverting RSA problem*: given y, find the e-th root of y modulo n. This corresponds to the relation

$$f(y, x) \stackrel{\mathsf{def}}{=} \left(y \stackrel{?}{=} x^e \bmod n\right),$$

which is a polynomially computable function. Therefore, the default decision problem, $R(f, y, x) = 1$ iff $f(y, x) = 1$, is trivial.

A more interesting relation is the least-significant bit of the e-th root of y:

Definition 7 (The lsb-D-RSA(n, e) Problem). *Given y, decide whether the least-significant bit of the e-th root of y, $x = y^{1/e} \bmod n$, is 0 or 1:*

$$R(f, y) \stackrel{\mathsf{def}}{=} \mathsf{lsb}(x \textit{ such that } f(y, x) = 1) = \mathsf{lsb}(y^{1/e} \bmod n).$$

Then, one can define the related gap problem, the lsb-G-RSA(n, e) problem. And therefore, with the results about hard-core bits of RSA [1,14], we know that the lsb-D-RSA is equivalent to the RSA problem, therefore the lsb-G-RSA problem is tractable (and even strongly tractable because of the random self-reducibility of the inverting problem).

3.2 The Diffie-Hellman Problems

The most famous family of problems is definitely the Diffie-Hellman problems [12]. Indeed, it already provides multiple variations (decision and computational versions) as well as interesting environments. Then let us consider any group $\mathcal{G}$ of prime order q. We define three problems as follows:

- *The Inverting Diffie-Hellman Problem (C-DH)* (a.k.a. the Computational Diffie-Hellman problem): given a triple of $\mathcal{G}$ elements (g, g^a, g^b), find the element $C = g^{ab}$.

- *The Decision Diffie-Hellman Problem (D-DH)*: given a quadruple of $\mathcal{G}$ elements (g, g^a, g^b, g^c), decide whether $c = ab \bmod q$ or not.
- *The Gap–Diffie-Hellman Problem (G-DH)*: given a triple (g, g^a, g^b), find the element $C = g^{ab}$ with the help of a Decision Diffie-Hellman Oracle (which answers whether a given quadruple is a Diffie-Hellman quadruple or not).

Note that the decision problem is the default one, when the relation f is defined by

$$f((g, A, B), C) \stackrel{\text{def}}{=} \left(\log_g C \stackrel{?}{=} \log_g A \times \log_g B \bmod q\right),$$

which is *a priori* not a polynomially computable function.

There also exist many possible variations of those problems where the first component, and possibly the second one are fixed:

$$\star\text{-DH}_g(\cdot) = \star\text{-DH}(g, \cdot) \text{ and } \star\text{-DH}_{g,h}(\cdot) = \star\text{-DH}(g, h, \cdot).$$

About the inverting problem, it is believed intractable in many groups (prime subgroups of the multiplicative groups $\mathbb{Z}_n^\star$ or $\mathbb{Z}_p^\star$ [18,23], prime subgroups of some elliptic curves [20], or of some Jacobians of hyper-elliptic curves [21,22]). The decision problem is also believed so in many cases. For example, in generic groups, where only generic algorithms [28] can be used, because of a non-manageable numeration, the discrete logarithm, the inverting Diffie-Hellman and the decision Diffie-Hellman problems have been proven to require the same amount of computation [35]. However, no polynomial time reduction has ever been proposed, excepted in groups with a smooth order [24,25,26]. Therefore, in all these groups used in cryptography, intractability of the gap problem is a reasonable assumption.

However, because of some dual properties in Abelian varieties, the decision Diffie-Hellman problem is easy over the Jacobians of some (hyper)-elliptic curves: namely, in [16], it has been stated the following result

Proposition 8. *Let m be an integer relatively prime to q, and let $\mu_m(\mathbb{F}_q)$ be the group of roots of unity in $\mathbb{F}_q$ whose order divides m. We furthermore assume that the Jacobian $J(\mathbb{F}_q)$ contains a point of order m. Then there is a surjective pairing*

$$\phi_m : J_m(\mathbb{F}_q) \times J(\mathbb{F}_q)/mJ(\mathbb{F}_q) \to \mu_m(\mathbb{F}_q)$$

which is furthermore computable in $\mathcal{O}(\log q)$ (where $J_m(\mathbb{F}_q)$ is the group of m-torsion points).

This pairing, the so-called Tate-pairing, can be used to relate the discrete logarithm in the group $J_m(\mathbb{F}_q)$ to the discrete logarithm in $\mathbb{F}_q^\star$, if $q - 1$ is divisible by m. A particular application [15] is over an elliptic curve, with a trace of the Frobenius endomorphism congruent to 2 modulo m. Indeed, for example, with an elliptic curve $J(\mathbb{F}_q) = E$ of trace $t = 2$ and $m = \#E = q + 1 - t = q - 1$, we have $J_m(\mathbb{F}_q) = J(\mathbb{F}_q)/mJ(\mathbb{F}_q) = E$ and $\mu_m(\mathbb{F}_q) = \mathbb{F}_q^\star$. Then,

$$\phi : E \times E \to \mathbb{F}_q^\star.$$

Let us consider a Diffie-Hellman quadruple, P, $A = a \cdot P$, $B = b \cdot P$ and $C = c \cdot P$,

$$\phi(A, B) = \phi(a \cdot P, b \cdot P) = \phi(P, P)^{ab} = \phi(P, ab \cdot P) = \phi(P, C).$$

And the latter equality only holds with the correct candidate C.

3.3 The Rabin Problems

Let us consider $n = pq$. We define three problems as follows:

- *The Inverting Rabin Problem* (a.k.a. the Factoring Problem): given y, find $x = y^{1/2} \bmod n$ if x exists. This corresponds to the relation

$$f(y, x) \stackrel{\mathsf{def}}{=} \left(x^2 \stackrel{?}{=} y \bmod n\right).$$

- *The Decision Rabin Problem* (a.k.a the Quadratic Residuosity Problem): given y, decide whether x exists or not.
- *The Gap–Rabin Problem*: given a pair y, find $x = y^{1/2} \bmod n$ if x exists, with the help of a Decision Rabin Oracle.

Note that these decision and gap problems correspond to the R relation

$$R(f, y) \stackrel{\mathsf{def}}{=} (\exists x \text{ such that } f(y, x) = 1).$$

Since no polynomial time reduction is known from the Factorization to the Quadratic-Residuosity problem, the Gap–Rabin assumption seems as reasonable as the Quadratic-Residuosity assumption.

It is worth remarking that like in the RSA case, the lsb-G-Rabin problem would be tractable because of hard-core bit result about the least-significant bit [1,14].

4 Application to Cryptography

This notion of gap-problem is eventually not new because it is involved in many practical situations. This section deals with undeniable signatures and designated confirmer signatures. More precisely we show that the security of some old and efficient such schemes is equivalent to a gap-problem, whereas it was just known weaker than the computational version.

4.1 Signatures

An important tool in cryptography is the authentication of messages. It is provided using digital signatures [17]. The basic property of a signature scheme, from the verifier point of view, is the easy verification of the relation between a message and the signature, whereas it should be intractable for anybody, excepted the legitimate signer, to produce a valid signature for a new message: the relation $f(m, \sigma)$, with input a message m and a signature σ, must be computable, while providing an intractable inverting problem. Therefore, an intractable gap-problem is required, with an easy decision problem.

4.2 Undeniable Signatures

In undeniable signatures [9,7], contrarily to plain signatures, the verification process must be intractable without the help of the signer (or a confirmer [8]). And therefore, the confirmer (which can be the signer himself) can be seen as a decision oracle.

Let us study the first example of undeniable signatures [9,7] whose security proof has been an open problem for more than 10 years. We will prove that the full-domain hash [3] variant of this scheme is secure under the Gap-DH problem, in the random oracle model [2].

Definition. First, we just define informally an undeniable signature scheme. For more details, the reader is referred to the original papers [9,7]. An undeniable signature scheme consists of 3 algorithms/protocols:

- key generation algorithm, which on input a security parameter produces a pair of secret/public keys $(\mathsf{sk}, \mathsf{pk})$ for the signer.
- signature protocol. It is a, possibly interactive, protocol in which, on input a message m and a signer secret key sk_s, the verifier gets a certificate s on m for which he is convinced of the validity, without being able to transfer this conviction to anybody.
- confirmation/disavowal protocol. It is a, possibly interactive, protocol in which, on input a message m and an alleged certificate s, the signer convinces the verifier whether the certificate s is actually related to m and pk or not, using his secret key sk (in a non-transferable way).

The security notions are similar to the plain signature setting [17]. One wants to prevent existential forgeries under chosen-message attacks. Then, an existential forgery is a certificate that the signer cannot repudiate whereas he did not produce it. But in such a context, the verification protocol can be called many times, on any message-certificate pair chosen by the adversary. We have to take care about this kind of oracle access, hence the gap-problems.

Description. The first proposal was a very nice and efficient protocol. It consists of a non-interactive signature process and an interactive confirmation protocol.

- Setting: g is a generator of a group $\mathcal{G}$ of prime order q. The secret key of the signer is a random element $x \in \mathbb{Z}_q$ while his public key is $y = g^x$.
- Signature of m: in order to sign a message m, the signer computes and returns $s = m^x$.
- Confirmation/Disavowal of (m, s): an interactive proof is used to convince the verifier whether

$$\log_g y = \log_m s \bmod q.$$

  In the first paper [9], this proof was not zero-knowledge, but it has been quickly improved in [7].

But we further slightly modify this scheme to prevent existential forgeries, namely by ruling out the basic multiplicative attacks: one uses the classical full-domain hash technique [3,10]. If this hash function is furthermore assumed to behave like a random oracle [2], this scheme can be proven secure. Moreover, to make the analysis easier, we replace the zero-knowledge interactive proof by a non-interactive but non-transferable proof. There are well-known techniques using trapdoor commitments [19] which are perfectly simulatable in the random oracle model [31].

Therefore, we analyze the following variant.

- Setting: g is a generator of a group $\mathcal{G}$ of prime order q. The secret key of the signer is a random element $x \in \mathbb{Z}_q$ while his public key is $y = g^x$. We furthermore need a hash function H which outputs random elements in the whole group $\mathcal{G}$.
- Signature of m: in order to sign a message m, the signer computes $h = H(m)$ and returns $s = h^x$.
- Confirmation/Disavowal of (m, s): the signer uses non-transferable NIZK proofs of either the equality or inequality between

$$\log_g y \text{ and } \log_h s \bmod q, \text{ where } h = H(m).$$

  Thus, the confirmation proof answers positively to the $\mathsf{D\text{-}DH}(g, y, h, s)$ problem whereas the disavowal proof answers negatively.

Security Analysis. Before providing such an analysis, one can state the following theorem:

Theorem 9. *An existential forgery under adaptively chosen-message attacks is equivalent to the Gap Diffie-Hellman problem.*

Proof. For this equivalence, one can easily see that if one can break the $\mathsf{C\text{-}DH}_{g,y}$ problem, possibly with access to a $\mathsf{D\text{-}DH}_{g,y}$ oracle (which means that the two first components are fixed to g and y), then one can forge a signature in a universal way: first, a $\mathsf{D\text{-}DH}_{g,y}$ oracle is simulated (with overwhelming probability) by the confirmation/disavowal protocols. Then, for any message m, one computes $h = H(m)$ as well as $\mathsf{C\text{-}DH}_{g,y}(m)$. Therefore, the security of this undeniable signature scheme is weaker than the $\mathsf{G\text{-}DH}_{g,y}$ problem.

In the opposite way, one can use the same techniques as in [3,10] for the security of the full-domain hash signature. Let us consider an adversary that is able to produce an existential forgery with probability ε within time t after q_h queries to the signing oracle, where g is the basis of $\mathcal{G}$ and y the public key of the signer. Then, we will use it to break the $\mathsf{G\text{-}DH}_{g,y}$ problem. Given $\alpha \in \mathcal{G}$, one tries to extract $\beta = \mathsf{C\text{-}DH}_{g,y}(\alpha) = \mathsf{C\text{-}DH}(g, y, \alpha)$. For that, we simulate any interaction with the adversary in an indistinguishable setting from a real attack:

- confirmation/disavowal queries are perfectly simulated by simulating the appropriate proof, correctly chosen thanks to the $\mathsf{D\text{-}DH}_{g,y}$ oracle.

- any hash query m is answered in a probabilistic way. More precisely, one chooses a random exponent $r \in \mathbb{Z}_q$ and then, with probability p, $H(m)$ is answered by α^r, otherwise it is answered by g^r.
- any signing query m (assumed to has already been asked to H) is answered as follows: if $H(m)$ has been defined as g^r, then $s = y^r$ is a valid signature for m since $s = y^r = g^{xr} = H(m)^x$, for x satisfying $y = g^x$. Otherwise, the simulation aborts.

Finally, the adversary outputs a forgery s for a new message m (also assumed to have already been asked to H). If $H(m)$ has been defined as α^r then $s = H(m)^x = \alpha^{rx}$. Consequently, $s^{1/r} = \mathsf{C\text{-}DH}(g, y, \alpha) = \mathsf{C\text{-}DH}_{g,y}(\alpha)$.

The success probability is exactly the same as for the full-domain hash technique [10]

$$\varepsilon' = \varepsilon(1-p)^{q_h} p \geq \exp(-1) \times \frac{\varepsilon}{q_h}, \text{ while simply choosing } p = \frac{1}{q_h + 1}.$$

□

4.3 Designated Confirmer Signatures

In 1994, Chaum [8] proposed a new kind of undeniable signatures where the signer is not required to confirm the signature, but a designated confirmer, who owns a secret. Furthermore, he proposed a candidate. The same year, Okamoto [29] proved that the existence of such schemes is equivalent to the existence of public-key encryption schemes. He furthermore gave an example, based on the Diffie-Hellman problem [12] (on which relies the security of the El Gamal encryption scheme [13]).

Let us first give a quick definition of this new cryptographic object together with the security notions. Then we study the Okamoto's example, using the Schnorr signature [34], in the random oracle model.

Definition. As for undeniable signatures, we just give an informal definition of designated confirmer signatures. For more details, the reader is referred to [8]. A designated confirmer signature scheme consists of 3 algorithms/protocols:

- key generation algorithm, which on input a security parameter produces two pairs of secret/public keys, the pair $(\mathsf{sk}_s, \mathsf{pk}_s)$ for the signer and the pair $(\mathsf{sk}_c, \mathsf{pk}_c)$ for the confirmer.
- signature protocol. It is a, possibly interactive, protocol in which, on input a message m, a signer secret key sk_s and a confirmer public key pk_c, the verifier gets a certificate s on m for which he is convinced of the validity, without being able to transfer this conviction.
- confirmation/disavowal protocol. It is a, possibly interactive, protocol in which, on input a message m and an alleged certificate s, the confirmer convinces the verifier whether the certificate s is actually related to m and pk_s or not, using his secret key sk_c (in a non-transferable way).

The security notions are the same as for undeniable signatures, excepted that the confirmer may be a privileged adversary: an existential forgery is a certificate that the confirmer cannot repudiate, whereas the signer never produced it. Once again, the confirmation protocol can be called many times, on any message-certificate pair chosen by the adversary. However this kind of oracle is of no help for the confirmer, in forging a certificate.

Description. Let us describe the original Okamoto's example [29], using the Schnorr signature [34]. Because of a flaw remarked by Michels and Stadler [27], one cannot prove the security of this scheme against attacks performed by the confirmer. Then we focus on standard adversaries.

- Setting: g is a generator of a group $\mathcal{G}$ of prime order q. The secret key of the signer is a random element $x \in \mathbb{Z}_q$ while his public key is $y = g^x$. We furthermore need a hash function H which outputs elements in $\mathcal{G}$ (still full-domain hash). The confirmer also owns a secret key $a \in \mathbb{Z}_q$ associated to the public key $b = g^a$.
- Signature of m: in order to sign a message m, the signer chooses random $r, w \in \mathbb{Z}_q$, computes

$$d = g^r, t = g^w, e = b^r \cdot H(m, t) \text{ and } s = w - ex \bmod q$$

  and returns (d, e, s). The signer can furthermore prove the validity of this signature by proving, in a non-interactive and non-transferable zero-knowledge way, the equality between

$$\log_g d \text{ and } \log_b z \bmod q, \text{ where } z = e/H(m, g^s y^e).$$

- Confirmation/Disavowal of $(m, (d, e, s))$: the verifier and the confirmer, both compute $z = e/H(m, g^s y^e)$ and the confirmer uses non-interactive and non-transferable zero-knowledge proofs of either the equality or inequality between

$$\log_g b \text{ and } \log_d z \text{ modulo } q.$$

Thus, the confirmation proof by the signer answers positively to $\mathsf{D\text{-}DH}(g, d, b, z)$, and the confirmation proof by the confirmer answers positively to $\mathsf{D\text{-}DH}(g, b, d, z)$ whereas the disavowal proof answers negatively.

Therefore, one can get the answer of $\mathsf{D\text{-}DH}(g, b, d, z)$, which is indeed equivalent to $\mathsf{D\text{-}DH}(b, g, z, d)$, for any (d, z) of his choice, which looks like to a $\mathsf{D\text{-}DH}_{b,g}$ oracle.

Security Analysis. Once again, one can state the following theorem:

Theorem 10. *An existential forgery under adaptively chosen-message attacks, for a standard adversary (not the confirmer), is equivalent to the Gap Diffie-Hellman problem.*

Proof. First, if one can break the $\mathsf{C\text{-}DH}_{b,g}$ problem, possibly with access to a $\mathsf{D\text{-}DH}_{b,g}$ oracle, then one can forge a signature in a universal way: indeed, a $\mathsf{D\text{-}DH}_{b,g}$ oracle is simulated, as already seen, by the confirmation/disavowal protocols. Then, for any message m, one chooses random s and e, computes $t = g^s y^e$ and $z = e/H(m,t)$. Then one gets $d = \mathsf{C\text{-}DH}(b,g,z) = \mathsf{C\text{-}DH}_{b,g}(z)$ which completes a valid signature (d,e,s). Therefore, the security of this designated confirmer signature scheme is weaker than the $\mathsf{G\text{-}DH}_{b,g}$ problem.

In the opposite way, one can use a replay technique [32]. Let us consider an adversary that is able to produce an existential forgery with probability ε within time t after q_s queries to the signing oracle and q_h queries to the random oracle H, where g is the basis of $\mathcal{G}$ and b the public key of the confirmer.

Remark 11. We furthermore need to assume that the bit-length k of the notation of $\mathcal{G}$-elements is not too large comparatively to q: $q/2^k$ must be non-negligible.

Then, we will use it to break the $\mathsf{G\text{-}DH}_{b,g}$ problem. Given $\alpha \in \mathcal{G}$, one tries to extract $\beta = \mathsf{C\text{-}DH}_{b,g}(\alpha) = \mathsf{C\text{-}DH}(b,g,\alpha)$. For that, we simulate any interaction with the adversary in an indistinguishable setting from a real attack:

- for setting up the system, we furthermore choose a random $x \in \mathbb{Z}_q$ and define $y = g^x$ to be the public key of the signer.
- confirmation/disavowal queries are perfectly simulated by simulating the appropriate proof, correctly chosen thanks to the $\mathsf{D\text{-}DH}_{b,g}$ oracle.
- any new hash query is answered by a random element in $\mathcal{G}$.
- any signing query m is perfectly simulated thanks to the secret key x of the signer.

Finally, the adversary outputs a forgery (d,e,s) for a new message m. One computes $t = g^s y^e$, stores $h = H(m,t)$ (which has been defined) and replays the adversary with the same random tape, a new random oracle H' which outputs the same answers than H did until the query (m,t) appears. But this latter query is that time answered by e/α^u for a randomly chosen u. With non-negligible probability, the adversary outputs a new forgery (d',e',s') based on the same query (m,t) to the random oracle H. Since $t = g^s y^e = g^{s'} y^{e'}$

- either $s' = s \bmod q$ and $e' = e \bmod q$
- or the adversary can be used to break the discrete logarithm problem (indeed, the signing answers could be simulated without the secret key x, thanks to the random oracle which makes the non-interaction proof simulatable [32]).

Therefore, one may assume that $s' = s \bmod q$ and $e' = e \bmod q$. Since the answer to (m,t) given by the new random oracle H' is totally independent of e, we furthermore have $e' = e$ in the group $\mathcal{G}$, with probability $q/2^k$, which has been assumed non-negligible. Thus,

$$z' = e'/H'(m, g^{s'} y^{e'}) = e/H'(m,t) = \alpha^u.$$

Consequently,

$$d' = \mathsf{C\text{-}DH}(b,g,z'), \text{ and thus, } \beta = {d'}^{1/u} = \mathsf{C\text{-}DH}(b,g,\alpha).$$

□

5 Conclusion

This paper introduced a novel class of computational problems, the *gap problems*, which is considered to be dual to the class of the *decision problems*. We have shown how the gap problems find natural applications in cryptography, namely for proving the security of some primitive schemes like Chaum's undeniable signatures and designated confirmer signatures.

But there are still other clear applications. For example, they appear while considering a new kind of attacks, the *plaintext-checking attacks*, against public-key encryption scheme. And they help us to provide REACT, a Rapid Enhanced-security Asymmetric Cryptosystem Transform [30], which makes into a chosen-ciphertext secure cryptosystem any weakly secure scheme.

Other applications will certainly appear. Anyway, it is worth noting that it had been open for more than 10 years to prove the security of Chaum's undeniable signatures.

References

1. W. Alexi, B. Chor, O. Goldreich, and C. P. Schnorr. RSA and Rabin Functions: Certain Parts are as Hard as the Whole. *SIAM Journal on Computing*, 17:194–209, 1988.
2. M. Bellare and P. Rogaway. Random Oracles Are Practical: a Paradigm for Designing Efficient Protocols. In *Proc. of the 1st CCS*, pages 62–73. ACM Press, New York, 1993.
3. M. Bellare and P. Rogaway. The Exact Security of Digital Signatures – How to Sign with RSA and Rabin. In *Eurocrypt '96*, LNCS 1070, pages 399–416. Springer-Verlag, Berlin, 1996.
4. S. A. Brands. An Efficient Off-Line Electronic Cash System Based on the Representation Problem. Technical Report CS-R9323, CWI, Amsterdam, 1993.
5. S. A. Brands. Off-Line Electronic Cash Based on Secret-Key Certificates. In *LATIN '95*, LNCS 911, pages 131–166. Springer-Verlag, Berlin, 1995.
6. S. A. Brands. Secret-Key Certificates. Technical Report CS-R9510, CWI, Amsterdam, 1995.
7. D. Chaum. Zero-Knowledge Undeniable Signatures. In *Eurocrypt '90*, LNCS 473, pages 458–464. Springer-Verlag, Berlin, 1991.
8. D. Chaum. Designated Confirmer Signatures. In *Eurocrypt '94*, LNCS 950, pages 86–91. Springer-Verlag, Berlin, 1995.
9. D. Chaum and H. van Antwerpen. Undeniable Signatures. In *Crypto '89*, LNCS 435, pages 212–216. Springer-Verlag, Berlin, 1990.
10. J.-S. Coron. On the Exact Security of Full-Domain-Hash. In *Crypto '2000*, LNCS 1880, pages 229–235. Springer-Verlag, Berlin, 2000.
11. R. Cramer and V. Shoup. A Practical Public Key Cryptosystem Provably Secure against Adaptive Chosen Ciphertext Attack. In *Crypto '98*, LNCS 1462, pages 13–25. Springer-Verlag, Berlin, 1998.
12. W. Diffie and M. E. Hellman. New Directions in Cryptography. *IEEE Transactions on Information Theory*, IT–22(6):644–654, November 1976.
13. T. El Gamal. A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. *IEEE Transactions on Information Theory*, IT–31(4):469–472, July 1985.

14. R. Fischlin and C. P. Schnorr. Stronger Security Proofs for RSA and Rabin bits. *Journal of Cryptology*, 13(2):221–244, 2000.
15. G. Frey, M. Müller, and H. G. Rück. The Tate-Pairing and the Discrete Logarithm Applied to Elliptic Curve Cryptosystems. *IEEE Transactions on Information Theory*, 45:1717–1719, 1999.
16. G. Frey and H. G. Rück. A Remark Concerning m-Divisibility and the Discrete Logarithm in the Divisor Class Group of Curves. *Mathematics of Computation*, 62:865–874, 1994.
17. S. Goldwasser, S. Micali, and R. Rivest. A Digital Signature Scheme Secure Against Adaptative Chosen-Message Attacks. *SIAM Journal of Computing*, 17(2):281–308, April 1988.
18. D. M. Gordon. Discrete Logarithms in GF(p) Using the Number Field Sieve. *SIAM Journal of Discrete Mathematics*, 6(1):124–138, February 1993.
19. M. Jakobsson, K. Sako, and R. Impagliazzo. Designated Verifier Proofs and Their Applications. In *Eurocrypt '96*, LNCS 1070, pages 143–154. Springer-Verlag, Berlin, 1996.
20. N. Koblitz. Elliptic Curve Cryptosystems. *Mathematics of Computation*, 48(177):203–209, January 1987.
21. N. Koblitz. A Family of Jacobians Suitable for Discrete Log Cryptosystems. In *Crypto '88*, LNCS 403, pages 94–99. Springer-Verlag, Berlin, 1989.
22. N. Koblitz. Hyperelliptic Cryptosystems. *Journal of Cryptology*, 1:139–150, 1989.
23. A. Lenstra and H. Lenstra. *The Development of the Number Field Sieve*, volume 1554 of *Lecture Notes in Mathematics*. Springer-Verlag, 1993.
24. U. M. Maurer and S. Wolf. Diffie Hellman Oracles. In *Crypto '96*, LNCS 1109, pages 268–282. Springer-Verlag, Berlin, 1996.
25. U. M. Maurer and S. Wolf. Diffie-Hellman, Decision Diffie-Hellman, and Discrete Logarithms. In *Proceedings of ISIT '98*, page 327. IEEE Information Theory Society, 1998.
26. U. M. Maurer and S. Wolf. The Diffie-Hellman Protocol. *Designs, Codes, and Cryptography*, 19:147–171, 2000.
27. M. Michels and M. Stadler. Generic Constructions for Secure and Efficient Confirmer Signature Schemes. In *Eurocrypt '98*, LNCS 1403, pages 406–421. Springer-Verlag, Berlin, 1998.
28. V. I. Nechaev. Complexity of a Determinate Algorithm for the Discrete Logarithm. *Mathematical Notes*, 55(2):165–172, 1994.
29. T. Okamoto. Designated Confirmer Signatures and Public Key Encryption are Equivalent. In *Crypto '94*, LNCS 839, pages 61–74. Springer-Verlag, Berlin, 1994.
30. T. Okamoto and D. Pointcheval. REACT: Rapid Enhanced-security Asymmetric Cryptosystem Transform. In *RSA '2001*, LNCS. Springer-Verlag, Berlin, 2001.
31. D. Pointcheval. Self-Scrambling Anonymizers. In *Financial Cryptography '2000*, LNCS. Springer-Verlag, Berlin, 2000.
32. D. Pointcheval and J. Stern. Security Arguments for Digital Signatures and Blind Signatures. *Journal of Cryptology*, 13(3):361–396, 2000.
33. R. Rivest, A. Shamir, and L. Adleman. A Method for Obtaining Digital Signatures and Public Key Cryptosystems. *Communications of the ACM*, 21(2):120–126, February 1978.
34. C. P. Schnorr. Efficient Signature Generation by Smart Cards. *Journal of Cryptology*, 4(3):161–174, 1991.
35. V. Shoup. Lower Bounds for Discrete Logarithms and Related Problems. In *Eurocrypt '97*, LNCS 1233, pages 256–266. Springer-Verlag, Berlin, 1997.