Proceedings of Financial Cryptography '2001
(19–22 february 2001, Grand Cayman Island, British West Indies)
P. Syverson Ed., Springer-Verlag, LNCS 2339, pages 319–338.

The Power of RSA Inversion Oracles and the Security of Chaum's RSA-Based Blind Signature Scheme

M. Bellare[1], C. Namprempre[1], D. Pointcheval[2], and M. Semanko[1]

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

[2] Dépt. d'Informatique-CNRS, École Normale Supérieure
45 rue d'Ulm, 75230 Paris, Cedex 05, France.
Email: David.Pointcheval@ens.fr
URL: http://www.dmi.ens.fr/users/pointche/

Abstract. Blind signatures are the central cryptographic component of digital cash schemes. In this paper, we investigate the security of the first such scheme proposed, namely Chaum's RSA-based blind signature scheme, in the random-oracle model. This leads us to formulate and investigate a new class of RSA-related computational problems which we call the "one-more-RSA-inversion" problems. Our main result is that two problems in this class which we call the chosen-target and known-target inversion problems, have polynomially-equivalent computational complexity. This leads to a proof of security for Chaum's scheme in the random oracle model based on the assumed hardness of either of these problems.

1 Introduction

Blind signatures are the central cryptographic component of digital cash schemes. Withdrawer and Bank run the blind signature protocol to enable the former to obtain the latter's signature on some token without revealing this token to the bank, thereby creating a valid but anonymous ecoin. In this paper, we investigate the security of the first such scheme proposed, namely Chaum's RSA-based blind signature scheme [7]. This leads us to formulate and investigate a new class of RSA-related computational problems which we call the "one-more-RSA-inversion" problems. We begin with a high-level description of our approach and its motivation.

The gap between proofs and practice. Chaum's RSA-based blind signature scheme [7] is simple and practical, and (assuming the underlying hash function is properly chosen) has so far resisted attacks. Yet there seems little hope of proving its security (even in a random oracle model [3]) based on the "standard" one-wayness assumption about the RSA function: it seems that the security of the scheme relies on different, and perhaps stronger, properties of RSA.

This is a common situation. It exhibits a gap created by what assumptions we prefer to make and what schemes we want to validate. The reliance on unproven computational properties of RSA for security naturally inclines us to be conservative and to stick to standard assumptions, of which the favorite is that RSA is one-way. Designers who have worked with RSA know, however, that it seems to have many additional strengths. These are typically exploited, implicitly rather than explicitly, in their designs. The resulting schemes might well resist attack but are dubbed "heuristic" because no proof of security based on the standard assumption seems likely. This leads designers to seek alternative schemes that can be proven under the standard

assumptions. If the alternatives have cost comparable to that of the original scheme then they are indeed attractive replacements for the latter. But often they are more expensive. Meanwhile, the use of the original practical scheme is being discouraged even though it might very well be secure.

We take a different approach. Rather than going "forward" from assumptions to schemes —meaning, trying to find a scheme provable under some given standard assumption— we try to go "backwards" from schemes to assumptions — meaning to distill properties of RSA that are sufficient to guarantee the security of the *given* scheme.

We suggest that practical RSA-based schemes that have resisted attack (in this case, Chaum's RSA-based blind signature scheme) are manifestations of strengths of the RSA function that have not so far been properly abstracted or formalized. We suggest that one should build on the intuition of designers and formulate explicit computational problems that capture the above-mentioned strengths and suffice to prove the security of the scheme. These problems can then be studied to see how they relate to other problems and to what extent we can believe in them as assumptions. Doing so will lead to a better understanding of the security of the schemes. It will also highlight computational problems that might then be recognized as being at the core of other schemes, and enlarge the set of assumptions we might be willing to make, leading to benefits in the design or analysis of other schemes.

In this paper, we formalize a class of computational problems which we call *one-more-RSA-inversion* problems. They are natural extensions of the RSA-inversion problem underlying the notion of one-wayness to a setting where the adversary has access to a decryption oracle, and we show that the assumed hardness of one problem in this class —namely the *chosen-target inversion problem*— suffices to prove the security of Chaum's RSA-based blind signature scheme in the random oracle model. We then study this assumption, taking the standard approach in a domain of conjectures: we try to gain confidence in the assumption by relating it to other assumptions. Below, we first discuss the new computational problems and their properties and then tie this in with the blind signature scheme.

THE RSA SYSTEM. Associated with a modulus N and an encryption exponent e are the RSA function and its RSA-inverse defined by

$$\mathsf{RSA}_{N,e}(x) = x^e \bmod N \text{ and } \mathsf{RSA}^{-1}_{N,e}(y) = y^d \bmod N$$

where $x, y \in \mathsf{Z}^*_N$ and d is the decryption exponent. To *invert* RSA at a point $y \in \mathsf{Z}^*_N$ means to compute $x = \mathsf{RSA}^{-1}_{N,e}(y)$. The commonly made and believed assumption is that the RSA function is one-way. In other words, the following problem is hard:

RSA single-target inversion problem: RSA-STI

Input: N, e and a random target point $y \in \mathsf{Z}^*_N$
Find: $y^d \bmod N$

Hardness (i.e. computational intractability) is measured via the usual convention: the success probability of an adversary, whose time-complexity is polynomial in the length k of the modulus, is negligible, the probability being over the choice of keys N, e, d as well as over any random choices explicitly indicated in the problem, in this case y. A problem is easy if it is not hard.

THE ONE-MORE-RSA-INVERSION PROBLEMS. We are interested in settings where the protocol is such that the legitimate user —and hence the adversary— has access to

an oracle $\mathsf{RSA}_{N,e}^{-1}(\cdot)$ for the inverse RSA function. (The adversary can provide a value $y \in \mathsf{Z}_N^*$ to its oracle and get back $x = \mathsf{RSA}_{N,e}^{-1}(y) = y^d \bmod N$, but it is not directly given d. We will see later how the RSA-blind signature scheme fits this setting.) A security property apparently possessed by RSA is that an adversary can only make "trivial" use of this oracle. We capture this in the following way. The adversary is given some random *target points* $y_1, \ldots, y_n \in \mathsf{Z}_N^*$, and we say it wins if the number of these points whose RSA-inverse it manages to compute exceeds the number of calls it makes to its oracle. That is, it computes "one more RSA-inverse." Within this framework we consider two specific problems. They are parameterized by polynomially-bounded functions $n, m\colon\ \mathsf{N} \to \mathsf{N}$ of the security parameter k satisfying $n(\cdot) > m(\cdot)$–

RSA known-target inversion problem: RSA-KTI[m]

Input:	N, e and random target points $y_1, \ldots, y_{m(k)+1} \in \mathsf{Z}_N^*$
Oracle:	RSA-inversion oracle computing $\mathsf{RSA}_{N,e}^{-1}(\cdot) = (\cdot)^d \bmod N$ but only $m(k)$ calls allowed
Find:	$y_1^d, \ldots, y_{m(k)+1}^d \bmod N$

RSA chosen-target inversion problem: RSA-CTI[n, m]

Input:	N, e and random points $y_1, \ldots, y_{n(k)+1} \in \mathsf{Z}_N^*$
Oracle:	RSA-inversion oracle computing $\mathsf{RSA}_{N,e}^{-1}(\cdot) = (\cdot)^d \bmod N$ but only $m(k)$ calls allowed
Find:	Indices $1 \le i_1 < \cdots < i_{m(k)+1} < n(k)$ and $y_{i_1}^d, \ldots, y_{i_{m(k)+1}}^d \bmod N$

In the first problem, the number of oracle calls allowed to the adversary is just one fewer than the number of target points, so that to win it must compute the RSA-inverse of all target points. In the second version of the problem, the adversary does not have to compute the RSA-inverses of all target points but instead can choose some $m(k) + 1$ points out of $n(k)$ given points and wins if it can find their RSA-inverses using only $m(k)$ oracle calls.

The RSA-KTI[0] problem is identical to the standard RSA-STI problem. (When $m(\cdot) = 0$ the adversary's task is to find the RSA-inverse of one given random point y_1 without making any oracle queries.) In this sense, we consider security against known-target inversion to be a natural extension of one-wayness to a setting where the adversary has access to an RSA-inversion oracle.

We note in Remark 5 that if factoring reduces in polynomial time to RSA inversion then both the above problems are easy. Accordingly, these problems can be hard only if factoring does not reduce to RSA inversion. Some evidence that the latter is true is provided by Boneh and Venkatesan [6].

Relations among one-more-RSA-inversion problems. We note in Remark 4 that if problem RSA-CTI[n, m] is hard then so is problem RSA-KTI[m]. (If you can solve the latter then you can solve the former by RSA-inverting the first $m(k) + 1$ target points.) However, it is conceivable that the ability to choose the target points might help the adversary considerably. Our main result is that this is not so. We show in Theorem 6 that if problem RSA-KTI[m] is hard then so is problem RSA-CTI[n, m], for any polynomially-bounded $n(\cdot)$ and $m(\cdot)$. (This result assumes that the encryption exponent e is prime.) We prove the theorem by showing how given any polynomial-time adversary B that solves RSA-KTI[m] we can design a polynomial-time adversary A that solves RSA-CTI[n, m] with about the same probability. The reduction exploits

linear algebraic techniques which in this setting are complicated by the fact that the order $\phi(N)$ of the group over which we must work is not known to the adversary.

THE RSA-BASED BLIND SIGNATURE SCHEME. The signer's public key is N, e, and its secret key is N, d where these quantities are as in the RSA system. The signature of a message M is

$$x = \mathsf{RSA}_{N,e}^{-1}(H(M)) = H(M)^d \bmod N \tag{1}$$

where $H\colon \{0,1\}^* \to \mathsf{Z}_N^*$ is a public hash function. A message-tag pair (M, x) is said to be *valid* if x is as in Equation (1). The blind signature protocol enables a user to obtain the signature of a message M without revealing M to the signer, as follows. The user picks r at random in Z_N^*, computes $\overline{M} = r^e \cdot H(M) \bmod N$, and sends $\overline{M}$ to the signer. The latter computes $\overline{x} = \mathsf{RSA}_{N,e}^{-1}(\overline{M}) = \overline{M}^d \bmod N$ and returns $\overline{x}$ to the user, who extracts the signature $x = \overline{x} \cdot r^{-1} \bmod N$ of M from it. Two properties are desired, *blindness* and *unforgeability*. Blindness means the signer does not learn anything about M from the protocol that it did not know before, and it is easy to show that this is unconditionally true [7]. Unforgeability in this context is captured via the notion of one-more-forgery of Pointcheval and Stern [18, 19]. (The standard notion of [13] does not apply to blind signatures.) The forger can engage in interactions with the signer in which it might not follow the prescribed protocol for the user. (As discussed further in Section 3 there are, in general, a variety of attack models for these interactions [18, 19, 14, 16], but in the case of the RSA blind signature protocol, all are equivalent.) Nothing prevents it from coming up with one valid message-tag pair per protocol execution (to do this, it just has to follow the user protocol) but we want it to be hard to come up with more. We ask that the number of valid message-tag pairs that a forger can produce cannot exceed the number of executions of the blind-signature protocol in which it engages with the signer.

It is the unforgeability property that has been the open question about the RSA-based blind signature scheme. Michels, Stadler and Sun [15] show that one can successfully obtain one-more forgery if the hash function is poorly implemented. Here, we will assume that the hash function is a random oracle. (The forger and signer both get an oracle for H.) In that case, the signature scheme is the FDH scheme of [4]. This scheme is proven to meet the standard security notion for digital signatures of [13] in the random oracle model assuming that RSA is one-way [4, 8], but this result won't help us here. To date, no attacks against the one-more-forgery goal are known on the blind FDH-RSA signature scheme. We would like to support this evidence of security with proofs.

When the forger interacts with a signer in Chaum's blind signature protocol detailed above, the former effectively has access to an RSA-inversion oracle: it can provide the signer any $\overline{M} \in \mathsf{Z}_N^*$ and get back $\overline{M}^d \bmod N$. It is the presence of this oracle that makes it unlikely that the one-wayness of RSA alone suffices to guarantee unforgeability. However, the one-more-RSA-decryption problems were defined precisely to capture settings where the adversary has an RSA-inversion oracle, and we will be able to base the security of the signature scheme on hardness assumptions about them.

UNFORGEABILITY OF THE FDH-RSA BLIND SIGNATURE SCHEME. In Lemma 13, we provide a reduction of the security against one-more-forgery of the FDH-RSA blind signature scheme, in the random oracle model, to the security of the RSA chosen-target inversion problem. Appealing to Theorem 6 we then get a proof of unforgeability for the blind FDH-RSA scheme, in the random oracle model, under the assumption that

the RSA known-target inversion problem is hard. (Again, this is for prime encryption exponents.) These results simplify the security considerations of the blind FDH-RSA scheme by eliminating the hash function and signature issues from the picture, leaving us natural problems about RSA to study.

PERSPECTIVE. An obvious criticism of the above result is that the proof of security of the blind FDH-RSA signature scheme is under a novel and extremely strong RSA assumption which is not only hard to validate but crafted to have the properties necessary to prove the security of the signature scheme. This is true, and we warn that the assumptions should be treated with caution. But we suggest that our approach and results have pragmatic value. Certainly, one could leave the blind RSA signature scheme unanalyzed until someone proves security based on the one-wayness of RSA, but this is likely to be a long wait. Meanwhile, we would like to use the scheme and the practical thing to do is to understand the basis of its security as best we can. Our results isolate clear and simply stated properties of the RSA function that underlie the security of the blind signature scheme and make the task of the security analyst easier by freeing him or her from consideration of properties of signatures and hash functions. It is better to know exactly what we are assuming, even if this is very strong, than to know nothing at all.

EXTENSIONS. The analogues of the one-more-RSA-inversion problems can be formulated for any family of one-way functions. We can prove that the known-target inversion and chosen-target inversion problems have polynomially-equivalent computational complexity also for the discrete logarithm function in groups of prime order. (That proof is actually a little easier than the one for RSA in this paper because in the discrete log case the order of the group is public information.)

RELATED WORK. Other non-standard RSA related computational problems whose study has been fruitful include strong-RSA [11, 2, 12, 9] and dependent-RSA [17]. For more information about RSA properties and attacks see [5].

2 Complexity of the one-more-RSA-inversion problems

Throughout this paper, $k \in \mathsf{N}$ denotes the security parameter. We let KeyGen be the *RSA key generation algorithm* which takes k as input and returns the values N, e and d where N is a k-bit RSA modulus (product of two $k/2$ bit random primes p_1, p_2) and $e, d \in \mathsf{Z}^*_{\phi(N)}$ with $ed \equiv 1 \bmod \phi(N)$ where $\phi(N) = (p_1 - 1)(p_2 - 1)$. (The public key is N, e and the secret key is N, d.) *The results in this paper will assume that e is prime.*

Below, we provide the formal definitions corresponding to the computational problems discussed in Section 1. In each case, we associate to any given adversary an *advantage* function which on input the security parameter k returns the probability that an associated *experiment* returns 1. The problem is *hard* if the advantage of any adversary of time-complexity $\mathrm{poly}(k)$ is negligible, and we say that a problem is *easy* if it is not hard. Furthermore, we adopt the convention that the time-complexity of the adversary refers to the function which on input k returns the execution time of the full associated experiment including the time taken to compute answers to oracle calls, plus the size of the code of the adversary, in some fixed model of computation. This convention will simplify concrete security considerations.

ONE-WAYNESS OF RSA. We recall the standard notion, couching it in a way more suitable for comparison with the new notions.

Definition 1. (Single-Target Inversion Problem: RSA-STI) Let $k \in \mathsf{N}$ be the security parameter. Let A be an adversary. Consider the following experiment:

Experiment $\mathbf{Exp}^{\text{rsa-sti}}_{A}(k)$

$(N, e, d) \xleftarrow{R} \text{KeyGen}(k)$
$y \xleftarrow{R} \mathsf{Z}^*_N \,;\; x \leftarrow A(N, e, k, y)$
If $x^e \equiv y \pmod N$ then return 1 else return 0

We define the advantage of A via

$$\mathbf{Adv}^{\text{rsa-sti}}_{A}(k) = \Pr[\,\mathbf{Exp}^{\text{rsa-sti}}_{A}(k) = 1\,]\,.$$

The RSA-STI problem is said to be *hard* —in more standard terminology, RSA is said to be *one-way*— if the function $\mathbf{Adv}^{\text{rsa-kti}}_{A,m}(\cdot)$ is negligible for any adversary A whose time-complexity is polynomial in the security parameter k. ▮

The Known-Target Inversion Problem. We denote by $(\cdot)^d \bmod N$ the oracle that takes input $y \in \mathsf{Z}^*_N$ and returns its RSA-inverse y^d. An adversary solving the known-target inversion problem is given oracle access to $(\cdot)^d \bmod N$ and is given $m(k)+1$ targets where $m : \mathsf{N} \to \mathsf{N}$. Its task is to compute the RSA-inverses of *all* the targets while submitting at most $m(k)$ queries to the oracle.

Definition 2. (Known-Target Inversion Problem: RSA-KTI[m]) Let $k \in \mathsf{N}$ be the security parameter, and let $m : \mathsf{N} \to \mathsf{N}$ be a function of k. Let A be an adversary with access to an RSA-inversion oracle $(\cdot)^d \bmod N$. Consider the following experiment:

Experiment $\mathbf{Exp}^{\text{rsa-kti}}_{A,m}(k)$

$(N, e, d) \xleftarrow{R} \text{KeyGen}(k)$
For $i = 1$ to $m(k)+1$ do $y_i \xleftarrow{R} \mathsf{Z}^*_N$
$(x_1, \ldots, x_{m(k)+1}) \leftarrow A^{(\cdot)^d \bmod N}(N, e, k, y_1, \ldots, y_{m(k)+1})$
If the following are both true then return 1 else return 0
- $\forall i \in \{1, \ldots, m(k)+1\} : x_i^e \equiv y_i \pmod N$
- A made at most $m(k)$ oracle queries

We define the advantage of A via

$$\mathbf{Adv}^{\text{rsa-kti}}_{A,m}(k) = \Pr[\,\mathbf{Exp}^{\text{rsa-kti}}_{A,m}(k) = 1\,]\,.$$

The RSA-KTI[m] problem is said to be *hard* if the function $\mathbf{Adv}^{\text{rsa-kti}}_{A,m}(\cdot)$ is negligible for any adversary A whose time-complexity is polynomial in the security parameter k. The known-target inversion problem is said to be hard if RSA-KTI[m] is hard for all polynomially-bounded $m(\cdot)$. ▮

Notice that RSA-KTI[0] is the same as RSA-STI. That is, the standard assumption that RSA is one-way is exactly the same as saying that RSA-KTI[0] is hard.

The Chosen-Target Inversion Problem. An adversary solving the chosen-target inversion problem is given access to an RSA-inversion oracle as above, and $n(k)$ targets where $n : \mathsf{N} \to \mathsf{N}$. Its task is to compute $m(k)+1$ RSA-inversions of the given targets, where $m : \mathsf{N} \to \mathsf{N}$ and $m(k) < n(k)$, while submitting at most $m(k)$ queries to the oracle. The choice of which targets to compute the RSA-inversion is up to the adversary. This choice is indicated by the range of the injective map π. (Notationally, this is different from the definition provided in Section 1. There, indices for elements chosen by the adversary are explicitly indicated. These indices constitute the range of the map π used here.)

Definition 3. (Chosen-Target Inversion Problem: RSA-CTI$[n, m]$) Let $k \in \mathsf{N}$ be the security parameter, and let $m, n : \mathsf{N} \to \mathsf{N}$ be functions of k such that $m(\cdot) < n(\cdot)$. Let B be an adversary with access to an RSA-inversion oracle $(\cdot)^d \bmod N$. Consider the following experiment:

Experiment $\mathbf{Exp}^{\text{rsa-cti}}_{B,n,m}(k)$
$(N, e, d) \xleftarrow{R} \text{KeyGen}(k)$
For $i = 1$ to $n(k)$ do $\overline{y}_i \xleftarrow{R} \mathsf{Z}_N^*$
$(\pi, \overline{x}_1, \ldots, \overline{x}_{m(k)+1}) \leftarrow B^{(\cdot)^d \bmod N}(N, e, k, \overline{y}_1, \ldots, \overline{y}_{n(k)})$
If the following are all true then return 1 else return 0
- $\pi\colon \{1, \ldots, m(k)+1\} \to \{1, \ldots, n(k)\}$ is injective
- $\forall i \in \{1, \ldots, m(k)+1\} : \overline{x}_i^e \equiv \overline{y}_{\pi(i)} \pmod N$
- A made at most $m(k)$ oracle queries

We define the advantage of A via

$$\mathbf{Adv}^{\text{rsa-cti}}_{B,n,m}(k) = \Pr[\,\mathbf{Exp}^{\text{rsa-cti}}_{B,n,m}(k) = 1\,]\,.$$

The RSA-CTI$[n, m]$ problem is said to be *hard* if the function $\mathbf{Adv}^{\text{rsa-cti}}_{B,n,m}(\cdot)$ is negligible for any adversary A whose time complexity is polynomial in the security parameter k. The chosen-target inversion problem is said to be hard if RSA-CTI$[n, m]$ is hard for all polynomially-bounded $n(\cdot)$ and $m(\cdot)$. ▮

Relations amongst the problems. We note a few simple relations before going to the main result.

Remark 4. Let $n, m\colon \mathsf{N} \to \mathsf{N}$ be polynomially-bounded functions of the security parameter k. If the RSA-CTI$[n, m]$ problem is hard then so is the RSA-KTI$[m]$ problem. This is justified as follows: given an adversary A for RSA-KTI$[m]$, we let B be the adversary for RSA-CTI$[n, m]$ that runs A on input the first $m(k) + 1$ of B's target points and returns the values returned by A. Then B's advantage is the same as A's.

Remark 5. If factoring reduces to RSA inversion then there exists a polynomially-bounded function $m\colon \mathsf{N} \to \mathsf{N}$ such that RSA-KTI$[m]$ is easy. (So the assumption that either the known-target or chosen-target inversion problems is hard is at least as strong as the assumption that factoring does not reduce to RSA inversion.) Let us briefly justify this. Assume that factoring reduces to RSA inversion. This means there is a polynomial-time algorithm R such that the probability that the following experiment returns 1 is non-negligible:

$(N, e, d) \xleftarrow{R} \text{KeyGen}(k)$
$(p_1, p_2) \leftarrow R^{(\cdot)^d \bmod N}(N, e, k)$
If p_1, p_2 are prime and $p_1 p_2 = N$ then return 1 else return 0.

Let m be the number of oracle queries made by R. We define adversary A as follows:

Adversary $A^{(\cdot)^d \bmod N}(N, e, k, y_1, \ldots, y_{m(k)+1})$
$(p_1, p_2) \leftarrow R^{(\cdot)^d \bmod N}(N, e, k)$
Compute d from p_1, p_2
Compute and return $y_1^d, \ldots, y_{m(k)+1}^d \bmod N$

The adversary A runs the algorithm R, answering to its inversion queries with the answers from its own oracle. It uses the fact that possession of the prime factors of N enables computation of the decryption exponent d, and having computed d, it can of course compute the RSA-inversions of as many points as it pleases.

Our main result is a converse to the claim of Remark 4.

Theorem 6. *Let $n, m\colon \mathsf{N} \to \mathsf{N}$ be polynomially-bounded functions of the security parameter k. If the RSA-KTI$[m]$ problem is hard then so is the RSA-CTI$[n, m]$ problem. Concretely, for any adversary B, there exists an adversary A so that*

$$\mathbf{Adv}^{\text{rsa-cti}}_{B,n,m}(k) \;\le\; \frac{9}{5} \cdot \mathbf{Adv}^{\text{rsa-kti}}_{A,m}(k) \tag{2}$$

and A has time-complexity

$$T_A(k) \;=\; T_B(k) + O\left(k^3 n(k)m(k) + k^4 m(k) + k^2 m(k)^5 + k m(k)^6\right) \tag{3}$$

where $T_B(\cdot)$ is the time-complexity of B.

We will now present some technical lemmas, and then proceed to the proof of Theorem 6. The reader might prefer to begin with Section 2.2 and refer to Section 2.1 as needed.

2.1 Technical lemmas

Before proving our main result we state and prove some relevant technical lemmas.

Lemma 7. *Let $s \ge 1$ be an integer, let I_s be the s by s identity matrix, and let*

$$C = \begin{bmatrix} c_{1,1} \cdots c_{1,s} \\ \vdots \quad\quad \vdots \\ c_{s,1} \cdots c_{s,s} \end{bmatrix} \quad \text{and} \quad D = \begin{bmatrix} d_{1,1} \cdots d_{1,s} \\ \vdots \quad\quad \vdots \\ d_{s,1} \cdots d_{s,s} \end{bmatrix}$$

be integer matrices such that $C \cdot D = \det(C) \cdot I_s$. Suppose N, e is an RSA public key and N, d is the corresponding secret key. Suppose $y_i, \overline{y}_i, v_i \in \mathsf{Z}^*_N$ for $i = 1, \ldots, s$ are related via

$$\overline{y}_i \;\equiv\; v_i^{-e} \cdot \prod_{j=1}^{s} y_j^{c_{j,i}} \pmod{N}\,. \tag{4}$$

Let $\overline{x}_i = \overline{y}_i^d \bmod N$ for $i = 1, \ldots, s$. Then, for $j = 1, \ldots, s$, we have

$$(y_j^d)^{\det(C)} \;\equiv\; \prod_{i=1}^{s} (v_i \cdot \overline{x}_i)^{d_{i,j}} \pmod{N}\,. \tag{5}$$

Proof (Lemma 7). Let $\delta_{l,j} = 1$ if $l = j$ and 0 otherwise. Since $C \cdot D = \det(C) \cdot I_s$ we know that

$$\sum_{i=1}^{s} c_{l,i} d_{i,j} \;=\; \det(C) \cdot \delta_{l,j} \tag{6}$$

for all $l, j = 1, \ldots, s$. We now verify Equation (5). Suppose $1 \le j \le s$. In the following, computations are all mod N. From Equation (4), we have

$$\prod_{i=1}^{s} (v_i \cdot \overline{x}_i)^{d_{i,j}} \;=\; \prod_{i=1}^{s} \left[v_i \cdot \left(v_i^{-e} \cdot \prod_{l=1}^{s} y_l^{c_{l,i}} \right)^d \right]^{d_{i,j}} \;=\; \prod_{i=1}^{s} \left[v_i \cdot v_i^{-1} \cdot \prod_{l=1}^{s} (y_l^d)^{c_{l,i}} \right]^{d_{i,j}}\,.$$

Simplifying the last expression, we obtain

$$\prod_{i=1}^{s}\prod_{l=1}^{s}(y_l^d)^{c_{l,i}d_{i,j}} \;=\; \prod_{l=1}^{s}\prod_{i=1}^{s}(y_l^d)^{c_{l,i}d_{i,j}} \;=\; \prod_{l=1}^{s}(y_l^d)^{\sum_{i=1}^{s} c_{l,i}d_{i,j}} \;=\; \prod_{l=1}^{s}(y_l^d)^{\det(C)\cdot\delta_{l,j}}$$

where the last equality is by Equation (6). Finally, we use the fact that $\delta_{l,j} = 1$ if $l = j$ and 0 otherwise. This tells us that the above is $(y_j^d)^{\det(C)}$ as desired. ∎

Lemma 8. *Let N, e be an RSA public key and N, d the corresponding secret key. Let $\alpha \in \mathsf{N}$ and $y, z \in \mathsf{Z}_N^*$. If $\gcd(\alpha, e) = 1$ and $(y^d)^\alpha \equiv z \pmod N$ then $(z^a y^b)^e \equiv y \pmod N$ where a, b are the unique integers such that $a\alpha + be = 1$.*

Proof (Lemma 8). This is a standard calculation:

$$(z^a y^b)^e = (y^{d\alpha})^{ae} y^{be} = y^{\alpha a + be} = y^1 = y$$

where the computations are all mod N. ∎

Next, we consider a question in probabilistic linear algebra.

Definition 9. Let $q \geq 2$ be a prime, and let $s \geq 1$ be an integer. We define $\mathrm{SProb}(q, s)$ to be the probability that $\det(M) \equiv 0 \pmod q$ when M is an s by s matrix formed by choosing all entries uniformly and independently from Z_q. ∎

It is tempting to think that the determinant of a random matrix is a random value and hence that $\mathrm{SProb}(q, s) = 1/q$. This, however, is not true. For example, a simple computation shows that $\mathrm{SProb}(q, 2) = 1/q + 1/q^2 - 1/q^3$. There is actually a standard formula (whose proof we will recall later) for this quantity–

$$\mathrm{SProb}(q, s) \;=\; 1 - \prod_{i=1}^{s}\left(1 - \frac{q^{i-1}}{q^s}\right) \;. \tag{7}$$

This formula, however, does not lend itself well to estimates. We would like a simple upper bound on $\mathrm{SProb}(q, s)$. We prove the following. (We don't use the lower bound in this paper but include it for completeness.)

Lemma 10. *Let $q \geq 2$ be a prime, and let $s \geq 1$ be an integer. Then*

$$\frac{1}{q} \;\leq\; \mathrm{SProb}(q, s) \;\leq\; \frac{1}{q} + \frac{1}{q^2} \;. \tag{8}$$

Proof (Lemma 10). View the matrix M as formed by successively choosing random row vectors from Z_q^s. Let M_i denote the vector which is the i-th row of M, and let LI_i denote the event that the vectors $M_1, \ldots, M_i$ are linearly independent over Z_q, for $i = 1, \ldots, s$. It is convenient to let LI_0 be the event having probability one. Let $\mathrm{SProb}(q, s, i) = \Pr[\,\neg\mathrm{LI}_i\,]$ for $i = 0, \ldots, s$ and note that $\mathrm{SProb}(q, s) = \mathrm{SProb}(q, s, s)$.

We briefly recall the justification for Equation (7) and use it to derive the lower bound. (The upper bound is derived by a separate inductive argument.) We have

$$1 - \mathrm{SProb}(q, s) \;=\; \prod_{i=1}^{s} \Pr\left[\,\mathrm{LI}_i \mid \mathrm{LI}_{i-1}\,\right] \;=\; \prod_{i=1}^{s} \frac{q^s - q^{i-1}}{q^s} \;=\; \prod_{i=1}^{s}\left(1 - \frac{q^{i-1}}{q^s}\right)$$

which is Equation (7). We derive the lower bound by upper bounding the product term of Equation (7) by the biggest term of the product:

$$\mathrm{SProb}(q, s) \geq 1 - \left(1 - \frac{1}{q}\right) = \frac{1}{q} \;.$$

For the upper bound, we first claim that the following recurrence is true for $i = 0, \ldots, s$:

$$\mathrm{SProb}(q,s,i) \;=\; \begin{cases} 0 & \text{if } i = 0 \\ \dfrac{q^{i-1}}{q^s} + \left(1 - \dfrac{q^{i-1}}{q^s}\right) \cdot \mathrm{SProb}(q,s,i-1) & \text{if } i \geq 1 \end{cases} \tag{9}$$

The initial condition is simply by the convention we adopted that $\Pr[\,\mathrm{LI}_0\,] = 1$. The recurrence is justified as follows for $i \geq 1$:

$$\begin{aligned} &\mathrm{SProb}(q,s,i) \\ &= \Pr[\,\neg\mathrm{LI}_i\,] \\ &= \Pr[\,\neg\mathrm{LI}_i \mid \mathrm{LI}_{i-1}\,] \cdot \Pr[\,\mathrm{LI}_{i-1}\,] + \Pr[\,\neg\mathrm{LI}_i \mid \neg\mathrm{LI}_{i-1}\,] \cdot \Pr[\,\neg\mathrm{LI}_{i-1}\,] \\ &= \Pr[\,\neg\mathrm{LI}_i \mid \mathrm{LI}_{i-1}\,] \cdot (1 - \mathrm{SProb}(q,s,i-1)) + 1 \cdot \mathrm{SProb}(q,s,i-1) \\ &= \Pr[\,\neg\mathrm{LI}_i \mid \mathrm{LI}_{i-1}\,] + (1 - \Pr[\,\neg\mathrm{LI}_i \mid \mathrm{LI}_{i-1}\,]) \cdot \mathrm{SProb}(q,s,i-1) \\ &= \frac{q^{i-1}}{q^s} + \left(1 - \frac{q^{i-1}}{q^s}\right) \cdot \mathrm{SProb}(q,s,i-1)\,. \end{aligned}$$

We claim that

$$\mathrm{SProb}(q,s,i) \;\leq\; \frac{q^i}{q^s} \cdot \frac{1}{q-1} \qquad \text{for } i = 0, \ldots, s\,. \tag{10}$$

This will be justified below. It already gives us an upper bound on $\mathrm{SProb}(q,s) = \mathrm{SProb}(q,s,s)$, namely $1/(q-1)$, but this is a little worse than our claimed upper bound. To get the latter, we use the recurrence for $i = s$ and use Equation (10) with $i = s-1$. This give us

$$\begin{aligned} \mathrm{SProb}(q,s) \;=\; \mathrm{SProb}(q,s,s) &= \frac{q^{s-1}}{q^s} + \left(1 - \frac{q^{s-1}}{q^s}\right) \cdot \mathrm{SProb}(q,s,s-1) \\ &\leq \frac{q^{s-1}}{q^s} + \left(1 - \frac{q^{s-1}}{q^s}\right) \cdot \frac{q^{s-1}}{q^s} \frac{1}{q-1} \end{aligned}$$

Simplifying this further, we get

$$\mathrm{SProb}(q,s) \;\leq\; \frac{1}{q} + \left(1 - \frac{1}{q}\right) \cdot \frac{1}{q} \frac{1}{q-1} \;=\; \frac{1}{q} + \frac{1}{q-1} \cdot \left(\frac{1}{q} - \frac{1}{q^2}\right) \;=\; \frac{1}{q} + \frac{1}{q^2}\,.$$

This is the claimed upper bound. It remains to justify Equation (10) which we do by induction on i. When $i = 0$, Equation (10) puts a positive upper bound on $\mathrm{SProb}(q,s,0)$, and hence, is certainly true. So assume $i \geq 1$. Substituting into the recurrence of Equation (9), we get

$$\begin{aligned} \mathrm{SProb}(q,s,i) &= \frac{q^{i-1}}{q^s} + \left(1 - \frac{q^{i-1}}{q^s}\right) \cdot \mathrm{SProb}(q,s,i-1) \\ &\leq \frac{q^{i-1}}{q^s} + \mathrm{SProb}(q,s,i-1)\,. \end{aligned}$$

Using the inductive hypothesis and simplifying, we have

$$\mathrm{SProb}(q,s,i) \;\leq\; \frac{q^{i-1}}{q^s} + \frac{q^{i-1}}{q^s} \frac{1}{q-1} \;=\; \frac{q^{i-1}}{q^s} \left(1 + \frac{1}{q-1}\right) \;=\; \frac{q^i}{q^s} \frac{1}{q-1}$$

as desired. ∎

Algorithm $A^{(\cdot)^d \bmod N}(N, e, k, y_1, \ldots, y_{m(k)+1})$

1 $q \leftarrow e$; $s \leftarrow m(k) + 1$
2 For $i = 1$ to $n(k)$ do
3 $\quad v[i] \xleftarrow{R} \mathsf{Z}_N^*$
4 $\quad$ For $j = 1$ to s do $c[j, i] \xleftarrow{R} \mathsf{Z}_q$
5 $\quad \overline{y}_i \leftarrow v[i]^{-e} \prod_{j=1}^{s} y_j^{c[j,i]} \bmod N$
6 $(\pi, \overline{x}_1, \ldots, \overline{x}_s) \leftarrow B^{(\cdot)^d \bmod N}(N, e, k, \overline{y}_1, \ldots, \overline{y}_{n(k)})$
7 For $j = 1, \ldots, s$ do
$\quad v_j \leftarrow v[\pi(j)]$
$\quad$ For $l = 1, \ldots, s$ do $c_{j,l} \leftarrow c[j, \pi(l)]$
8 $C \leftarrow \begin{bmatrix} c_{1,1} & \ldots & c_{1,s} \\ \vdots & & \vdots \\ c_{s,1} & \ldots & c_{s,s} \end{bmatrix}$
9 $\alpha \leftarrow \det(C)$
10 If $\alpha = 0$ then abort
11 Compute a matrix
$\quad D = \begin{bmatrix} d_{1,1} & \ldots & d_{1,s} \\ \vdots & & \vdots \\ d_{s,1} & \ldots & d_{s,s} \end{bmatrix}$
$\quad$ with integer entries such that $C \cdot D = \det(C) \cdot I_s$
12 For $j = 1$ to s do
13 $\quad z_j \leftarrow \prod_{i=1}^{s} (v_i \cdot \overline{x}_i)^{d_{i,j}} \bmod N$
14 If $\gcd(\alpha, e) \neq 1$ then abort
15 Compute $a, b \in \mathsf{Z}$ such that $a\alpha + be = 1$ via extended Euclid algorithm
16 For $j = 1$ to s do
17 $\quad x_j \leftarrow z_j^a \cdot y_j^b \bmod N$
18 Return $x_1, \ldots, x_s$

Fig. 1. Adversary A of the proof of Theorem 6.

2.2 Proof of Theorem 6

Overview. The adversary A is depicted in Figure 1. Its input is $(N, e, k$ and$)$ $s = m(k) + 1$ target points $y_1, \ldots, y_s$. Its goal is to compute $y_1^d, \ldots, y_s^d \bmod N$.

Adversary A will begin by computing $n(k)$ points $\overline{y}_1, \ldots, \overline{y}_{n(k)}$ as a (randomized) function of the given points $y_1, \ldots, y_s$. The property we want these to have is that, given the RSA-inverses of *any* s of the points $\overline{y}_1, \ldots, \overline{y}_{n(k)}$, it is possible to extract in polynomial time the RSA-inverses of the original target points, at least with high probability. If such a "reversible embedding" can be implemented then A's work is complete since invoking B on the points $\overline{y}_1, \ldots, \overline{y}_{n(k)}$ will cause the RSA-inverses of some s of these points to be returned. The question is, thus, how to compute and later reverse this "reversible embedding."

Lines 2–5 of Figure 1 show how to compute it. For each j, the point $\overline{y}_j$ is created by first raising each of $y_1, \ldots, y_s$ to a random power and then multiplying the obtained quantities. (This product is then multiplied by a random group element of which A knows the RSA-inverse in order to make sure that $\overline{y}_1, \ldots, \overline{y}_{n(k)}$ are uniformly and independently distributed and thus are appropriate to feed to B.) A detail worth remarking here is the choice of the range from which the exponents $c[j, i]$ are chosen. This is Z_q where we have set q equal to the encryption exponent e. We will see the reasons for this choice later.

Once the points $\overline{y}_1, \ldots, \overline{y}_{n(k)}$ have been defined, B is invoked. In executing B, adversary A will invoke its own oracle to answer RSA-inversion oracle queries of B. Notice that this means that the number of oracle queries made by A is exactly equal to the number made by B which is $s - 1 = m(k)$. Assuming that B succeeds, A is in possession of $\overline{x}_j \equiv \overline{y}_{\pi(j)}^d \pmod N$ for $j = 1, \ldots, s$ where $\pi(j)$ are indices of B's choice that A could not have predicted beforehand. The final step is to recover the RSA-inverses of the original target points.

To this end, A creates the matrix C shown in line 8 of the code. If this matrix has zero determinant then A will not be able to reverse its embedding and aborts. Assuming a non-zero determinant, A would like to invert matrix C. Since the entries are exponents, A would like to work modulo $\phi(N)$ but A does not know this value. Instead, it works over the integers. A can compute a "partial" RSA-inverse, namely an integer matrix D such that $C \cdot D$ is a known integer multiple of the s by s identity matrix I_s. The integer multiple in question is the determinant of C, and thus the matrix D is the adjoint of C. (We will discuss the computation of D more later.) Lines 12–18 show how A then computes $x_1, \ldots, x_s$ which we claim equal $y_1^d, \ldots, y_s^d$. We now proceed to the detailed analysis.

Analysis. Let NS be the event that $\det(C) \not\equiv 0 \pmod q$. (If this is true then not only is $\det(C) \neq 0$, meaning C is non-singular, but also $\gcd(\det(C), e) = 1$ because $q = e$ is prime.) Let "A succeeds" denote the event that $x_i = y_i^d$ for all $i = 1, \ldots, s$. Let "B succeeds" denote the event that $\overline{x}_j = \overline{y}_{\pi(j)}^d$ for all $j = 1, \ldots, s$. Then,

$$\begin{aligned}
&\Pr[\, A \text{ succeeds} \,] \\
&\quad \geq \Pr[\, A \text{ succeeds} \wedge B \text{ succeeds} \wedge \mathrm{NS} \,] \\
&\quad = \Pr[\, A \text{ succeeds} \mid B \text{ succeeds} \wedge \mathrm{NS} \,] \cdot \Pr[\, B \text{ succeeds} \wedge \mathrm{NS} \,] \,. \qquad (11)
\end{aligned}$$

We claim that

$$\Pr[\, A \text{ succeeds} \mid B \text{ succeeds} \wedge \mathrm{NS} \,] = 1 \qquad (12)$$

$$\Pr[\, B \text{ succeeds} \wedge \mathrm{NS} \,] \geq \frac{5}{9} \cdot \mathbf{Adv}^{\text{rsa-cti}}_{B,n,m}(k) \,. \qquad (13)$$

Equations (11), (12), and (13) imply Equation (2). So it remains to verify Equations (12), (13) and the time-complexity claimed in Equation (3). We begin with Equation (12). Lemma 7 tells us that, assuming B succeeds and $\det(C) \neq 0$, after line 13 of Figure 1, we have

$$(y_j^d)^{\det(C)} \equiv z_j \pmod N \qquad (14)$$

for $j = 1, \ldots, s$. Assume $\gcd(\alpha, e) = 1$. Then Equation (14) and Lemma 8 imply that at line 17 we have $x_j^e = y_j$ for all $j = 1, \ldots, s$, in other words, A succeeds. Now note that event NS implies that $\det(C) \neq 0$ and that $\gcd(\det(C), e) = 1$ because $q = e$ and e is prime. This completes the proof of Equation (12).

We now move on to the proof of Equation (13). Due to the random choice of $v[1], \ldots, v[n(k)]$, the points $\overline{y}_1, \ldots, \overline{y}_{n(k)}$ computed at line 5 and then fed to B are uniformly and independently distributed over Z_N^* regardless of the choices of $c[j, i]$. This means that the events "B succeeds" and NS are independent and also that the probability of the former is the advantage of B. Thus, we have

$$\Pr[\, B \text{ succeeds} \wedge \mathrm{NS} \,] \;=\; \Pr[\, \mathrm{NS} \,] \cdot \Pr[\, B \text{ succeeds} \,] \;=\; \Pr[\, \mathrm{NS} \,] \cdot \mathbf{Adv}^{\text{rsa-cti}}_{B,n,m}(k) \,.$$

So to complete the proof of Equation (13), it suffices to show that

$$\Pr[\, \mathrm{NS} \,] \;\geq\; \frac{5}{9} \,. \qquad (15)$$

Code	**Cost**
"For" loop at line 2	$O(k^3) \cdot n(k) \cdot s$
$\det(C)$	$O(s^4k + s^3k^2)$
Matrix D	$s^2 \cdot O(s^4k + s^3k^2)$
"For" loop at line 12	$O(k^2s) \cdot O(sk)$
Lines 14, 15	$O(sk) \cdot O(k)$
Line 17	$O(k^2) \cdot O(k^2s)$
Total	$O(k^3n(k)s + k^4s + k^2s^5 + ks^6)$

Fig. 2. Costs of computations of the algorithm of Figure 1. Recall that $s = m(k) + 1$

Recall that our adversary A sets $q = e$ (line 1 in Figure 1) and that $e \geq 3$ for RSA. We now apply Lemma 10 to get

$$\Pr[\,\mathrm{NS}\,] \;=\; 1 - \mathrm{SProb}(q, s) \;\geq\; 1 - \left(\frac{1}{q} + \frac{1}{q^2}\right) \;=\; 1 - \frac{1}{e} - \frac{1}{e^2} \;\geq\; 1 - \frac{1}{3} - \frac{1}{3^2} \;=\; \frac{5}{9}\,.$$

This proves Equation (15) and, hence, completes the proof of Equation (13). To complete the proof of Theorem 6, it remains to justify the claim of Equation (3) about the time complexity. The costs of various steps of the algorithm of the adversary A are summarized in Figure 2. We now briefly explain them.

As in the code, we let $s = m(k) + 1$. The "For" loop beginning at line 2 involves $n(k) \cdot s$ exponentiations of k-bit exponents which has the cost shown. Computation of determinants is done using the algorithm of [1]. This takes $O(r^4(\log(r) + k) + r^3k^2)$ time to compute the determinant of an r by r integer matrix each of whose entries is at most k-bits long. (Although somewhat faster algorithms are known [10], they are randomized, and for simplicity, we use a deterministic algorithm.) We use this algorithm in Step 9. In the worst case, e (and hence q) is k-bits long. So the entries of C are at most k-bits long, and the cost of computing $\det(C)$ is $O(s^4(\log(s)+k)+s^3k^2)$, which is $O(s^4k + s^3k^2)$ since $\log(s) = O(k)$. The matrix D is the adjoint matrix of C, namely the transpose of the co-factor matrix of C. We compute it by computing the co-factors using determinants. This involves computing s^2 determinants of submatrices of C so the cost is at most s^2 times the cost of computing the determinant of C. Line 13 involves computing exponentiations modulo N with exponents of the size of entries in D. The Hadamard bound tells us that the entries of D are bounded in size by $O(s(\log(s) + k)$, which simplifies to $O(sk)$, so the cost is this many k-bit multiplications. Euclid's algorithm used for lines 14, 15 runs in time the product of the lengths of α and e. Finally, the lengths of a, b cannot exceed this time, and they are the exponents in line 17.

3 The RSA Blind Signature Scheme

The RSA blind signature scheme [7] consists of three components: the key generation algorithm KeyGen described in Section 2; the *signing protocol* depicted in Figure 3; and the *verification algorithm*. The signer has public key N, e and secret key N, d. Here $H\colon\ \{0,1\}^* \to \mathsf{Z}_N^*$ is a public hash function which in our security analysis will be modeled as a random oracle [3]. In that case, the signature schemes is the FDH-RSA scheme of [4]. A message-tag pair (M, x) is said to be valid if $x^e \bmod N$ is equal to $H(M)$. The verification algorithm is the same as that of FDH-RSA: to verify the

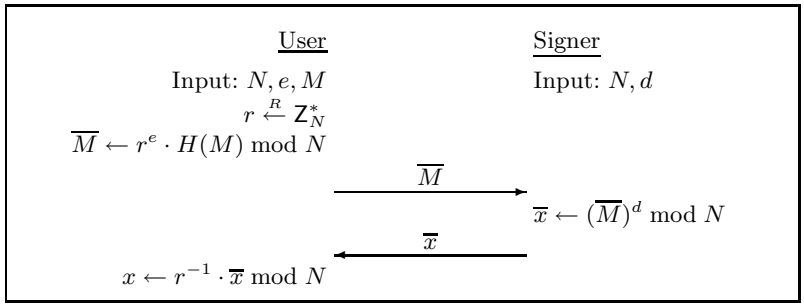

Fig. 3. Blind signing protocol for FDH-RSA

message-tag pair (M, x) using a public key (N, e), one simply checks if the message-tag pair is valid.

Unforgeability. In the standard formalization of security of a digital signature scheme —-namely unforgeability under adaptive chosen-message attack [13]— the adversary gets to submit messages of its choice to the signer and obtain their signature, and is then considered successful if it can forge the signature of a new message. This formalization does not apply for blind signatures because here nobody submits any messages to the signer to sign, and in fact the user is supposed to use the signer to compute a signature on a message which the signer does not know. Instead, we use the notion of security against one-more-forgery introduced in [18, 19]. The adversary (referred to as a *forger* in this context) is allowed to play the role of the user in the blind signature protocol. After some number of such interactions, it outputs a sequence of message-tag pairs. It wins if the number of these that are valid exceeds the number of protocol instances in which it engaged.

There are numerous possiblities with regard to the manner in which the adversary is allowed to interact with the signer, giving rise to different attack models. Some that have been considered are the sequential [18, 19] (where the adversary must complete one interaction before beginning another), the parallel [18, 19] or adaptive-interleaved [14] (where the adversary can engage the signer in several concurrent interactions), and a restricted version of the latter called synchronized-parallel [16]. However, in the blind signature protocol for FDH-RSA, the signer has only one move, and in this case the power of all these different types of attacks is the same.

Notice that in its single move the signer simply inverts the RSA function on the value supplied to it by the user in the previous move. Thus, the signer is simply an RSA inversion oracle. With this simplification we can make the following definition for security against one-more forgery which will cover all types of attacks.

Below, we let $[\{0,1\}^* \to \mathsf{Z}_N^*]$ denote the set of all maps from $\{0,1\}^*$ to Z_N^*. It is convenient to let the notation $H \xleftarrow{R} [\{0,1\}^* \to \mathsf{Z}_N^*]$ mean that we select a hash function H at random from this set. The discussion following the definition clarifies how we implement this selection of an object at random from an infinite space.

Definition 11. [Unforgeability of the blind FDH-RSA signature scheme] Let $k \in \mathsf{N}$ be the security parameter, and let $m, h : \mathsf{N} \to \mathsf{N}$ be functions of k. Let F be a forger with access to an RSA-inversion oracle and a hash oracle, denoted $(\cdot)^d \bmod N$ and $H(\cdot)$, respectively. Consider the following experiment:

Experiment $\mathbf{Exp}^{\text{rsa-omf}}_{F,h,m}(k)$

$H \xleftarrow{R} [\{0,1\}^* \to \mathsf{Z}_N^*]$
$(N, e, d) \xleftarrow{R} \text{KeyGen}(k)$
$((M_1, x_1), \ldots, (M_{m(k)+1}, x_{m(k)+1})) \leftarrow F^{(\cdot)^d \bmod N, H(\cdot)}(N, e, k)$
If the following are all true, then return 1 else return 0:

1. $\forall i \in \{1, \ldots, m(k)+1\} : H(M_i) \equiv x_i^e \bmod N$
2. Messages $M_1, \ldots, M_{m(k)+1}$ are all distinct
3. F made at most $m(k)$ queries to its RSA-inversion oracle
4. The number of hash-oracle queries made in this experiment is at most $h(k)$

We define the *advantage* of the forger F via

$$\mathbf{Adv}^{\text{rsa-omf}}_{F,h,m}(k) = \Pr[\,\mathbf{Exp}^{\text{rsa-omf}}_{F,h,m}(k) = 1\,]\,.$$

The FDH-RSA blind signature scheme is said to be *polynomially-secure against one-more forgery* if the function $\mathbf{Adv}^{\text{rsa-omf}}_{F,h,m}(\cdot)$ is negligible for any forger F whose time-complexity is polynomial in the security parameter k. ▮

Several conventions used here need to be detailed. The count of hash-oracle queries refers to the entire experiment, not just those made directly by the adversary, meaning those made in verifying the signatures in Step 3 are included in the count. We also need a convention regarding choosing the function H since it is an infinite object. The convention is that we do not actually view it as being chosen all at once, but rather view it as being built dynamically and stored in a table. Each time a query of M to the hash oracle is made, we charge the cost of the following: check whether a table entry $H(M)$ exists and if so return it; otherwise, pick an element y of Z_N^* at random, make a table entry $H(M) = y$, and return y. Recall that the time-complexity refers to the entire experiment as per conventions already stated in Section 2. In this regard, the cost of maintaining this table-based implementation of the hash function is included.

Security. We show that the FDH-RSA blind signature scheme is secure as long as the RSA known-target inversion problem is hard.

Theorem 12 (Unforgeability of the FDH-RSA blind signature scheme). *If the RSA known-target inversion problem is hard, then the FDH-RSA blind signature scheme is polynomially-secure against one-more forgery. Concretely, for any functions $m, h : \mathsf{N} \to \mathsf{N}$ and forger F, there exists an adversary A so that*

$$\mathbf{Adv}^{\text{rsa-omf}}_{F,h,m}(k) \;\le\; \frac{9}{5} \cdot \mathbf{Adv}^{\text{rsa-kti}}_{A,m}(k)$$

and the time-complexity of A is

$$T_A(k) \;=\; T_F(k) + O(k^3 n(k) m(k) + k^4 m(k) + k^2 m(k)^5 + k m(k)^6)$$

where $T_F(k)$ is the time-complexity of the forger F.

Theorem 12 follows directly from Theorem 6 and the following lemma saying that the FDH-RSA blind signature scheme is secure if the RSA *chosen*-target inversion problem is hard.

Lemma 13. *If the RSA chosen-target inversion problem is hard, then the FDH-RSA blind signature scheme is polynomially-secure against one-more forgery. Concretely, for any functions $m, h : \mathsf{N} \to \mathsf{N}$ and any forger F, there exists an adversary B so that*

$$\mathbf{Adv}^{\text{rsa-omf}}_{F,h,m}(k) \;\le\; \mathbf{Adv}^{\text{rsa-cti}}_{B,h,m}(k)$$

```
Algorithm B^{(·)^d mod N}(N, e, k, y_1, ..., y_{n(k)})
    count ← 0 ; s ← m(k) + 1
    Initialize associative arrays Hash and Ind to empty
    Initialize arrays Msg, X to empty
    Run F on input N, e, k replying to its oracle queries as follows:
        When F submits a hash query M do
            If Hash[M] is undefined then
                count ← count + 1 ; Hash[M] ← y_count ; Msg[count] ← M
            Return Hash[M]
        When F submits an RSA-inversion query y do
            Submit y to the RSA-inversion oracle (·)^d mod N and
            return its response.
    ((M_1, x_1), ..., (M_s, x_s)) ← F
    For j = 1 to s, do
        If Hash[M_j] is undefined then
            count ← count + 1 ; Hash[M_j] ← y_count ; Msg[count] ← M_j
        Ind[j] ← Find(Msg, M_j) ; X[Ind[j]] ← x_j
    Return (Ind, X[Ind[1]], ..., X[Ind[s]])
```

Fig. 4. Adversary B for the proof of Lemma 13.

and the time-complexity of B is

$$T_B(k) \;=\; T_F(k)$$

where $T_F(k)$ is the time-complexity of the forger F.

Proof (Lemma 13). Adversary B uses the forger F to achieve its goal by running F and providing answers to F's oracle queries. In response to hash-oracle queries, B simply returns its own targets to F. RSA-Inversion oracle queries of F are forwarded by B to its own RSA-inversion oracle and the results returned to F.

A detailed description of B is in Figure 4. It uses a subroutine $Find$ that looks for a given value in a given array. Specifically, it takes as its inputs an array of values A and a target value a assumed to be in the array, and returns the least index i such that $a = A[i]$.

The simulation is a largely straightforward use of random oracle techniques [3, 4] so we confine the analysis to a few remarks. Note that B simulates hash-oracle queries corresponding to the messages in the message-tag pairs output by F in case these are not already made. This ensures that the advantages of the two algorithms are identical. The time spent by B to maintain the hash-oracle table is the same as that spent in $\mathbf{Exp}^{\text{rsa-omf}}_{F,h,m}(k)$ as per the conventions discussed following Definition 11. We omit the details. ∎

References

1. J. Abbott, M. Bronstein, and T. Mulders. Fast deterministic computation of determinants of dense matrices. In *Proceedings of ACM International Symposium on Symbolic and Algebraic Computation*, pages 197–204. ACM Press, 1999.
2. N. Barić and B. Pfitzmann. Collision-free accumulators and fail-stop signature schemes without trees. In W. Fumy, editor, *Advances in Cryptology – EUROCRYPT ' 97*, volume 1233 of *Lecture Notes in Computer Science*, pages 480–494. Springer-Verlag, Berlin Germany, May 1997.
3. M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In V. Ashby, editor, *1st ACM Conference on Computer and Communications Security*. ACM Press, Nov. 1993.

4. M. Bellare and P. Rogaway. The exact security of digital signatures—how to sign with RSA and Rabin. In U. Maurer, editor, *Advances in Cryptology – EUROCRYPT ' 96*, volume 1070 of *Lecture Notes in Computer Science*, pages 399–416. Springer-Verlag, Berlin Germany, 12–16 May 1996.
5. D. Boneh. Twenty years of attacks on the RSA cryptosystem. *Notices of the American Mathematical Society*, 46(2):203–213, Feb. 1999.
6. D. Boneh and R. Venkatesan. Breaking RSA may not be equivalent to factoring. In K. Nyberg, editor, *Advances in Cryptology – EUROCRYPT ' 98*, volume 1233 of *Lecture Notes in Computer Science*, pages 59–71. Springer-Verlag, Berlin Germany, 1998.
7. D. Chaum. Blind signatures for untraceable payments. In D. Chaum, R. Rivest, and A. Sherman, editors, *Advances in Cryptology – CRYPTO ' 82*, Lecture Notes in Computer Science, pages 199–203. Plenum Press, New York and London, 1983, Aug. 1982.
8. J. Coron. On the exact security of full domain hash. In M. Bellare, editor, *Advances in Cryptology – CRYPTO ' 2000*, volume 1880 of *Lecture Notes in Computer Science*, pages 229–235. Springer-Verlag, Berlin Germany, Aug. 2000.
9. R. Cramer and V. Shoup. Signature schemes based on the strong RSA assumption. In *5th ACM Conference on Computer and Communications Security*, pages 46–51. ACM Press, Nov. 1999.
10. W. Eberly, M. Giesbrecht, and G. Villard. Computing the determinant and Smith form of an integer matrix. In *Proceedings of the 41st Symposium on Foundations of Computer Science*. IEEE, 2000.
11. E. Fujisaki and T. Okamoto. Statistical zero knowledge protocols to prove modular polynomial relations. In B. Kaliski Jr., editor, *Advances in Cryptology – CRYPTO ' 97*, volume 1294 of *Lecture Notes in Computer Science*, pages 16–30. Springer-Verlag, Berlin Germany, 17–21 Aug. 1997.
12. R. Gennaro, S. Halevi, and T. Rabin. Secure hash-and-sign signatures without the random oracle. In J. Stern, editor, *Advances in Cryptology – EUROCRYPT ' 99*, volume 1592 of *Lecture Notes in Computer Science*, pages 123–139. Springer-Verlag, Berlin Germany, May 1999.
13. S. Goldwasser, S. Micali, and R. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal on Computing*, 17(2):281–308, Apr. 1988. Special issue on cryptography.
14. A. Juels, M. Luby, and R. Ostrovsky. Security of blind digital signatures. In B. Kaliski Jr., editor, *Advances in Cryptology – CRYPTO ' 97*, volume 1294 of *Lecture Notes in Computer Science*, pages 150–164. Springer-Verlag, Berlin Germany, 17–21 Aug. 1997.
15. M. Michels, M. Stadler, and H. Sun. The security of some variants of the RSA signature scheme. In Y. Deswarte, editor, *Computer Security – ESORICS ' 98*, volume 1485 of *Lecture Notes in Computer Science*, pages 85–96. Springer-Verlag, Berlin Germany, 1998.
16. D. Pointcheval. Strengthened security for blind signatures. In K. Nyberg, editor, *Advances in Cryptology – EUROCRYPT '98*, volume 1403, pages 391–405. Springer-Verlag, Berlin Germany, 31–4 June 1998.
17. D. Pointcheval. New public key cryptosystems based on the dependent-RSA problems. In J. Stern, editor, *Advances in Cryptology – EUROCRYPT '99*, volume 1592, pages 239–255. Springer-Verlag, Berlin Germany, 1999.
18. D. Pointcheval and J. Stern. Provably secure blind signature schemes. In K. Kim and T. Matsumoto, editors, *Advances in Cryptology – ASIACRYPT ' 96*, Lecture Notes in Computer Science, pages 252–265. Springer-Verlag, Berlin Germany, 1996.
19. D. Pointcheval and J. Stern. Security arguments for digital signatures and blind signatures. *Journal of Cryptology*, 13(3):361–396, 2000.