Traceable Signatures

Aggelos Kiayias[*] Yiannis Tsiounis[†] Moti Yung[‡]

version of January 12, 2004

Abstract

We present, implement and apply a new privacy primitive that we call "Traceable Signatures." To this end we develop the underlying mathematical and protocol tools, present the concepts and the underlying security model, and then realize the scheme and its security proof. Traceable signatures support an extended set of fairness mechanisms (mechanisms for anonymity management and revocation) when compared with the traditional group signature mechanism. We demonstrate that this extended function is needed for proper operation and adequate level of privacy in various settings and applications. For example, the new notion allows (distributed) tracing of all signatures by a single (misbehaving) party without opening signatures and revealing identities of any other user in the system. In contrast, if such tracing is implemented by a state of the art group signature system, such wide opening of all signatures of a single user is a (centralized) operation that requires the opening of *all* anonymous signatures and revealing the users associated with them, an act that violates the privacy of all users.

Our work includes a novel modeling of security in privacy systems that leads to simulation-based proofs. Security notions in privacy systems are typically more complex than the traditional security of cryptographic systems, thus our modeling methodology may find future applications in other settings. To allow efficient implementation of our scheme we develop a number of basic tools, zero-knowledge proofs, protocols, and primitives that we use extensively throughout. These novel mechanisms work directly over a group of unknown order, contributing to the efficiency and modularity of our design, and may be of independent interest. The interactive version of our signature scheme yields the notion of "traceable (anonymous) identification."

[*]Computer Science and Eng. Dept., University of Connecticut, Storrs, CT, USA, `aggelos@cse.uconn.edu`.

[†]Etolian Capital, New York, NY, USA, `yiannist@etolian.com`. Research supported in part by NIST under grant SB1341-02-W-1113

[‡]Computer Science Dept., Columbia University, NY, USA `moti@cs.columbia.edu`

Contents

1 Introduction

A number of basic primitives have been suggested in cryptographic research to deal with the issue of privacy. The most flexible private authentication tool to date is "group-signatures," a primitive where each group member is equipped with a signing algorithm that incorporates a proof of group-membership. Group-signatures were introduced by Chaum and Van Heyst in [10] and were further studied and improved in many ways in [11, 8, 6, 7, 4, 2]. Each signature value is anonymous, in the sense that it only reveals that the issuer is a member of the group, without even linking signatures by the same signer.

Privacy comes at a price. Unconditional privacy seems to be an attractive notion from the user's viewpoint, nevertheless it can potentially be a very dangerous tool against public safety (and can even be abused against the user herself). Undoubtedly everybody understands that privacy is a right of law-abiding citizens, while at the same time a community must be capable of revoking such privacy when illegal behavior (performed under the "mask of privacy") is detected; this balancing act is thus called "fairness". Group-signatures were designed with one embedded fairness mechanism which, in fact, allows for the "opening" of an atomic signature value, revealing the identity of its signer. Such opening capability can be assigned to a special entity, a Trustee, which can also be distributed (to further increase privacy). Such functionality is possible in existing schemes, for example in the very efficient and scalable state of the art scheme of Ateniese et al. [2].

We observe that while group signatures are a very general "private credentials" tool, their opening capability is not a sufficient mechanism to ensure safety and/or privacy in a number of settings. What we need is additional mechanisms for lifting of privacy conditions. It may sound paradoxical that offering more mechanisms for revoking privacy actually contributes to privacy, but consider the following scenario: a certain member of the group is suspected of illegal activity (potentially, its identity was revealed by opening a signature value). It is then crucial to detect which signatures were issued by this particular member so that his/her transactions are traced. The only solution with the existing group signature schemes is to have the Group Manager (GM) open all signatures, thus violating the privacy of all (including law-abiding) group members. Furthermore, this operation is also scalability impairing, since the Group Manager would have to open all signatures in the system and these signatures may be distributed in various locations. What would be desirable, instead, is to have a mechanism that allows the selective linking of the existing signatures of a misbehaving user without violating the privacy of law-abiding group members; this mechanism should be efficient (e.g. done in parallel by numerous agents when required). This capability, in fact, implements an "oblivious data mining" where only signature values of a selected misbehaving user are traced. Such traceability property should be offered in conjunction with the standard opening capability of group signatures.

Another type of traceability, "self-traceability," is helpful to the user and is important in our setting. It suggests that a user should also be capable of claiming that he is the originator of a certain signature value if he wishes (or when a certain application protocol requires this). In other words, a group-member should be capable of stepping out and *claiming* a certain group-signature value as his own, *without* compromising the privacy of the remaining past or future group-signatures that he/she issues. Adding self-traceability to the existing solutions in group-signatures is also far from ideal: at best the user will be required to remember her private random coin-tosses for all the signatures she signed, which is an unreasonable user storage overhead in many settings.

Our Concept: Motivated by the above, in this work we introduce a new basic primitive which we call *Traceable Signatures.* It incorporates the following three different types of traceability: (i) user tracing: check whether a signature was issued by a given user; it can be applied to all signatures by agents running in parallel; (ii) signature opening: reveal the signer of a given signature (as in group signature); and (iii) signature claiming: the signer of a signature provably claims a given signature that it has signed. When recovering all transactions by performing user tracing it may be useful to avoid collecting all signatures to a central location and in order to reduce the burden of the GM (which may be a distributed entity), we divide user tracing into two steps: the first is executed by the GM and reveals some secret information about the user; this is given to a set of designated agents (clerks) that scan all signatures in parallel and reveal those signed by the suspected user. Note that the secret information revealed should not allow the agents to impersonate the user or violate the anonymity of law-abiding users.

Modeling: We model our concepts of traceable signatures and their interactive version (as traceable identification) and define their correctness and security.

We introduce a novel way of modeling privacy systems which is more general than previous models. The model includes the definition of correctness and of security properties of the system. In a security system, like encryption, it is obvious who is the attacker and who tries to defend the encryption device, so adversary modeling is relatively easy. In a privacy system, on the other hand, a protocol between many parties may involve mutually distrusting, malicious users attacking each other from many sides and in various coalitions: e.g., a server (perhaps collaborating with a subset of some users) trying to violate the user's privacy interacting with a user trying to impersonate a group member. Since in privacy systems we deal with mutually adversarial parties, we develop a new model that copes with this situation and is geared towards simulation-based security proofs.

To this effect, we introduce a set of queries by which adversaries can manipulate the system (and the simulator during the security proof). Then we present an "array of security definitions," where each definition is modeled as an adversary with partial access to the queries, representing a capability that the attack captures. This allows us to deal with various notions of simultaneous adversarial behavior within one system, modeling them as an "array of attacks" and proving security against each of them. Specifically in our setting, we classify three general security requirements that capture all possibilities for adversarial activity: misidentification attacks, anonymity attacks and framing attacks. We note that previous security notions that have appeared in the group signature literature such as unforgeability, coalition-resistance and exculpability are subsumed by our classification.

Constructions: Our construction is motivated by the state of the art and in particular by the mathematical assumptions that allow a group of users to generate a multitude of keys modulo a composite number that are private, namely are (partially) unknown even to the group manager who owns a trapdoor (prime factorization of the composite); such an ingenious mathematical setting was presented in [2]. Due to the refined notions of fairness of our model and its extended functionality, we need to introduce a number of new tools as well as employ a number of new cryptographic constructs that enable the various mechanisms that our model and scheme employ. We also note that our scheme is consistent with the present state-of-the-art revocation method for group signatures presented in [9], thus member revocation can be added modularly to our construction. We remark that the user tracing (combined with the GM publishing the user's "tracing trapdoor") can be used to implement a type of "CRL-based revocation" that nullifies all signatures by a private key. This type of revocation has been considered recently in [3].

In order to implement the scheme efficiently, we design a number of basic protocols and primitives that we use extensively throughout (as useful subroutines). A pleasing feature of these novel notions and protocols is that they work directly over a group of unknown order. We show useful properties of such groups of quadratic residues that are required for the security proofs. We then introduce the notion of "discrete-log relation sets" which is a generic way of designing zero-knowledge proof systems that allows an entity to prove efficiently the knowledge of a number of witnesses for any such relation set that involves various discrete-logarithms and satisfies a condition that we call "triangularity." Triangular discrete-log relation sets are employed extensively in our protocols but, in fact, they are a useful as an abstraction that can be used elsewhere and are therefore of independent interest. We then define a notion called "discrete-log representations of arbitrary powers," as well as a mechanism we call "drawing random powers" which is a two party protocol wherein one party gets a secret discrete logarithm whose value she does not control, while at the same time the other party gets the public key version, i.e., the exponentiated value.

Based on the above primitives we present traceable signatures and prove their correctness and security. We remark that our traceable signature scheme adds only a constant overhead to the complexity measures of the state of the art group signature scheme of [2].

Applications: We demonstrate the power of the new notion by presenting a number of applications. One generic application is transforming an anonymous system to one with "fair privacy." Another is a mix-net application where originators of messages or messages of an originator can be opened, while otherwise retaining privacy. A specific application to open-bid auctions is also discussed.

Organization: In section 2 we present basic preliminary technical details regarding the intractability assumptions that are employed in our primitives. The main technical content of this work starts at section 3 that investigates certain mathematical properties of the group of quadratic residues modulo a composite. Section 4 then describes the notion of "discrete-log relation sets", while Section 5 presents the notion of "discrete log representations of arbitrary powers" and Section 6 introduces the basic protocol of "drawing random powers". We then move to the conceptual part and present definitions and modelling of our notion in Section 7 which presents the properties and the careful adversarial model of traceable signatures and identification. Combining the definitions and model and the basic protocol constructions, Section 8 deals with the design of the traceable signature scheme, while Section 9 presents the correctness and security proofs. Section 10, in turn, presents a number of applications.

Notations: The notation $S(a, b)$ (called a sphere of radius b centered at a where $a, b \in \mathbb{Z}$ denotes the set $\{a-b+1, \ldots, a+b-1\}$. A function in w will be called negligible if it holds that it is smaller than any fraction of the form $\frac{1}{w^c}$ for any c and sufficiently large w; we use the notation $\mathsf{negl}(w)$ for such functions. The concatenation of two strings a, b will be denoted by $a||b$. If a is a bitstring we denote by $(a)_{l,\ldots,j}$ the substring $(a)_l|| \ldots ||(a)_j$ where $(a)_i$ denotes the i-th bit of a. For any set A, we will denote by $\#A$ its cardinality. If X and Y are parameterized probability distributions with the same support, we will write $X \approx Y$ if the statistical distance between X, Y is a negligible function in the parameter. Furthermore, if f and g are functions over a variable, we will write $f \approx g$ if their absolute distance is a negligible function in the same variable. Finally note that $\log$ denotes the logarithm base 2, PPT stands for "probabilistic polynomial-time," and $=_{\mathsf{df}}$ means "equal by definition."

2 Preliminaries

Throughout the paper we work (unless noted otherwise) in the group of quadratic residues modulo n, denoted by $QR(n)$, with $n = pq$ and $p = 2p'+1$ and $q = 2q'+1$. All operations are to be interpreted as modulo n (unless noted otherwise). We will employ various related security parameters (as introduced in the sequel); with respect to $QR(n)$ the relevant security parameter is the number of bits needed to represent the order of the group, denoted by $\nu =_{\mathsf{df}} \lceil \log p'q' \rceil$. Next we define the Cryptographic Intractability Assumptions that will be relevant in proving the security properties of our constructions.

The first assumption is the so called Strong-RSA assumption. It is similar in nature to the assumption of the difficulty of finding e-th roots of arbitrary elements in $\mathbb{Z}_n^*$ with the difference that the exponent e is not fixed (part of the instance).

Definition 1 **Strong-RSA**. *Given a composite n (as described above), and $z \in QR(n)$, it is infeasible* *to find $u \in \mathbb{Z}_n^*$ and $e > 1$ such that $u^e = z(\bmod n)$, in time polynomial in ν.*

The second assumption that we will employ is the Decisional Diffie-Hellman Assumption over the quadratic residues modulo n; in stating this assumption we also take into account the fact that the exponents may belong to pre-specified integer spheres $\mathcal{B} \subseteq \{1, \ldots, p'q'\}$.

Definition 2 Decisional Diffie-Hellman *(over $\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3$) Given a generator g of a cyclic group $QR(n)$ where n is as above, a DDH distinguisher $\mathcal{A}$ is a polynomial in ν time* PPT *that distinguishes the family of triples of the form $\langle g^x, g^y, g^z \rangle$ from the family of triples of the form $\langle g^x, g^y, g^{xy} \rangle$, where $x \in_R \mathcal{B}_1$, $y \in_R \mathcal{B}_2$, and $z \in_R \mathcal{B}_3$.*

The maximum distance of these two distributions of triples as quantified over all possible PPT *distinguishers will be denoted by $\mathsf{Adv}^{DDH}_{\mathcal{B}_1,\mathcal{B}_2,\mathcal{B}_3}(\nu)$; if $\mathcal{B}_1 = \mathcal{B}_2 = \mathcal{B}_3 = \{1, \ldots, p'q'\}$ we will write simply $\mathsf{Adv}^{DDH}(\nu)$ instead. The* DDH *assumption suggests that this advantage is a negligible function in ν.*

We remark that when the size of the spheres $\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3$ are sufficiently close to the order of $QR(n)$ it will hold that $\mathsf{Adv}^{DDH}_{\mathcal{B}_1,\mathcal{B}_2,\mathcal{B}_3}(\nu) \approx \mathsf{Adv}^{DDH}(\nu)$. Nevertheless we discover that the spheres can be selected to be much smaller than that without any degradation in security (see the remark at the end of section 3).

Finally, we will employ the discrete-logarithm assumption over the quadratic residues modulo n and a pre-specified sphere $\mathcal{B}$, when the factorization of n is known:

Definition 3 Discrete-Logarithm. *Given two values a, b that belong to the set of quadratic residues modulo n with known factorization, so that $\exists x \in \mathcal{B} : a^x = b$, find in time polynomial in ν the integer x so that $a^x = b$. Again $\mathcal{B}$ is an integer sphere into the set $\{1, \ldots, p'q'\}$.*

Conventions. (i) our proofs of knowledge will only be proven to work properly in the honest-verifier setting. On the one hand, the honest-verifier setting is sufficient for producing signatures. On the other hand, even in the general interactive setting the honest-verifier scenario can be enforced by assuming the existence, e.g., of a beacon, or some other mechanism that can produce trusted randomness; alternatively the participants may execute a distributed coin flipping algorithm (which are by now standard tools for converting random coin honest verifier scenario to a general proof). Such protocols where the randomness that is used to select the challenge is trusted will be called "canonical." (ii) the public parameters employed in our various protocol designs (e.g., the composite modulus n) will be be assumed to be selected honestly (in a system initiation protocol or procedure).

3 Sphere Truncations of Quadratic Residues

Let n be a composite so that $n = pq$ and $p = 2p' + 1$ and $q = 2q' + 1$ with p, q, p', q' all prime. Let a be a generator of the cyclic group of quadratic residues modulo n. Recall that the order of $QR(n)$ is $p'q'$. Let $S(2^\ell, 2^\mu) = \{2^\ell - 2^\mu + 1, \ldots, 2^\ell + 2^\mu - 1\}$ be a sphere for two parameters $\ell, \mu \in \mathbb{N}$. Observe that $\#S(2^\ell, 2^\mu) = 2^{\mu+1} - 1$.

In this section we will prove a basic result that will be helpful later in the analysis of our scheme. In particular we will show that, assuming factoring is hard and the fact the sphere $S(2^\ell, 2^\mu)$ is sufficiently large (but still not very large) the random variable a^x with $x \in_R S(2^\ell, 2^\mu)$ is indistinguishable from the uniform distribution over $QR(n)$; note that the result becomes trivial if the size of the sphere is very close to the order of $QR(n)$; we will be interested in cases where the size of the sphere is exponentially smaller (but still sufficiently large). Intuitively, this means that a truncation of the $QR(n)$ as defined by the sphere $S(2^\ell, 2^\mu)$ is indistinguishable to any probabilistic polynomial-time observer.

Let $\nu = \lceil \log p'q' \rceil$. Consider the function $f_{g,n}(x) = g^x(\mathrm{mod} n)$ defined for all $x < n$. The inverse of this function $f_{g,n}^{-1}$ is defined for any element in $QR(n)$ so that $f_{a,n}^{-1}(y) = x$ where $x \leq p'q'$ and it holds that $a^x = y(\mathrm{mod} n)$. Observe that x can be written as a ν-bitstring. Note that if y is uniformly distributed over $\mathbb{Z}_n^*$ it holds that every bit $(x)_i$ of x with $i = 1, \ldots, \nu$ follows a probability distribution $\mathcal{D}_i^\nu$ with support the set $\{0, 1\}$. Note that for the $\mathcal{O}(\log \nu)$ most significant bits i it holds that the distribution $\mathcal{D}_i^\nu$ is biased towards 0, whereas for the remaining bits the distribution $\mathcal{D}_i^\nu$ is uniform; this bias is due to the distance between 2^ν and $p'q'$. Below we define the simultaneous hardness of the bits of the discrete-logarithm function, (cf. [15]):

Definition 4 *The bits $[l, \ldots, j]$, $l > j$, of $f_{g,n}^{-1}$ are* simultaneously hard *if the following two distributions are* PPT*-indistinguishable:*

- *the $\mathcal{SD}_i^j$ distribution: $\langle (f_{g,n}^{-1}(y))_{i,\ldots,j}, y\rangle$ where $y \in_R QR(n)$.*
- *the $\mathcal{SR}_i^j$ distribution:$\langle r_l || \ldots || r_j, y\rangle$ where $y \in_R QR(n)$ and $r_i \leftarrow \mathcal{D}_i^\nu$ for $i = l, \ldots, j$.*

Hastad et al. [15] studied the simultaneous hardness of of the discrete-logarithm over composite groups and one of their results imply the following theorem:

Theorem 5 *The bits $[\nu, \ldots, j]$ of $f_{g,n}^{-1}$ are simultaneously hard under the assumption that factoring n is hard, provided that $j = \lceil \frac{\nu}{2} \rceil - \mathcal{O}(\log \nu)$.*

Now let us return to the study of the subset of $QR(n)$ defined by the sphere $S(2^\ell, 2^\mu)$. Consider the uniform probability distribution $\mathcal{U}$ over $QR(n)$ and the probability distribution $\mathcal{D}_a^{S(2^\ell,2^\mu)}$ with support $QR(n)$ that assigns the probability $1/(2^{\mu+1} - 1)$ to all elements a^x with $x \in S(2^\ell, 2^\mu)$ and probability 0 to all remaining elements of the support. The main result of this section is the following theorem:

Theorem 6 *The probability distributions $\mathcal{D}_a^{S(2^\ell,2^\mu)}$ and $\mathcal{U}$ with support $QR(n)$ are* PPT*-indistinguishable under the assumption that factoring n is hard, provided that $\#S(2^\ell, 2^\mu) = 2^{\lceil \frac{\nu}{2} \rceil - \mathcal{O}(\log \nu)}$.*

Proof. Let $\mathcal{A}$ be a probabilistic polynomial-time distinguisher for the two distributions $\mathcal{G}$ and $\mathcal{D}_a^{S(2^\ell,2^\mu)}$. Consider the modification of $\mathcal{A}$ called $\mathcal{A}'$ that given the input b it simulates $\mathcal{A}$ on input $ba^{-2^\ell+2^\mu}$.

We will show how $\mathcal{A}'$ can be turned into a distinguisher for the simultaneous hardness of the sequence of bits $[\nu, \ldots, \mu + 2]$ for the discrete-logarithm function. By theorem 5 the result will follow.

Let $\tilde{c} = \langle c_\nu || \ldots || c_{\mu+2}, y\rangle$ be a challenge for the simultaneous hardness of the discrete-log bits $[\nu, \ldots, \mu + 2]$. We compute the following:

$$y^* =_{\mathsf{df}} ya^{-c_\nu 2^{\nu-1} - c_{\nu-1} 2^{\nu-2} - \ldots - c_{\mu+2} 2^{\mu+1}}$$

observe that if $\tilde{c}$ is drawn from the probability distribution $\mathcal{SD}_{\mu+2}^\nu$ it follows that the above operation will cancel all the high order bits of y, and as a result y^* will be an element that is uniformly distributed over the subset $\{1, a, \ldots, a^{2^{\mu+1}-1}\}$ of $QR(n)$. Alternatively, if $\tilde{c}$ is drawn from the probability distribution $\mathcal{SR}_{\mu+2}^\nu$ it follows that y^* is uniformly distributed over $QR(n)$.

It is clear from the above that $\mathcal{A}'$ is a distinguisher between the probability distributions $\mathcal{SD}_{\mu+2}^\nu$ and $\mathcal{RD}_{\mu+2}^\nu$ with the same advantage as the distinguishing advantage of $\mathcal{A}$ between the probability distributions $\mathcal{D}_a^{S(2^\ell,2^\mu)}$ and $\mathcal{U}$. Based on the assumption on the size of the sphere $S(2^\ell, 2^\mu)$ we can employ theorem 5 to complete the proof. □

Remark. The results of this section suggest that we may truncate the range of a random variable a^x, $x \in_R \{1, \ldots, p'q'\}$, into a subset of $QR(n)$ that is of size approximately $\sqrt{p'q'}$; this truncation will not affect the behavior of any polynomial-time bounded observer. In particular, for the case of the Decisional Diffie Hellman assumption in $QR(n)$ over the spheres $\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3$, we may use spheres of size approximately $\sqrt{p'q'}$; under the assumption that factoring is hard, we will still maintain that $\mathsf{Adv}^{DDH}_{\mathcal{B}_1,\mathcal{B}_2,\mathcal{B}_3}(\nu) \approx \mathsf{Adv}^{DDH}(\nu)$. In some few cases we may need to employ the DDH over spheres that are smaller in size than $\sqrt{p'q'}$ (in particular we will employ the sphere $\mathcal{B}_2$ to be of size approximately $\sqrt[4]{p'q'}$). While the DDH over such sphere selection does not appear to be easier it could be possible that this version of DDH is a stronger intractability assumption. Nevertheless we remark that if we assume that factoring remains hard even if $\lceil \nu/4 \rceil$ of bits of the prime factors of n are known[1] then as stated in [15] approximately 3/4 of the bits of $f_{g,n}^{-1}$ are simultaneously hard and thus, using the methodology developed in this section, we can still argue that $\mathsf{Adv}^{DDH}_{\mathcal{B}_1,\mathcal{B}_2,\mathcal{B}_3}(\nu) \approx \mathsf{Adv}^{DDH}(\nu)$, even if $\mathcal{B}_2$ is selected to be of size approximately $\sqrt[4]{p'q'}$.

4 Discrete-log Relation Sets

Discrete-log relation sets are quite useful in planning complex proofs of knowledge for protocols operating over groups of unknown order in general. Below, let G be the unknown order group of quadratic residues modulo n, denoted also by $QR(n)$, where n is an RSA modulus that satisfies $n = pq = (2p'+1)(2q'+1)$ with p, q, p', q' all prime numbers.

Definition 7 *A* discrete-log relation set *R with z relations over r variables and m objects is a set of relations defined over the objects $A_1, \ldots, A_m \in G$ and the free variables $\alpha_1, \ldots, \alpha_r$ with the following specifications: (1) The i-th relation in the set R is specified by a tuple $\langle a_1^i, \ldots, a_m^i \rangle$ so that each a_j^i is selected to be one of the free variables $\{\alpha_1, \ldots, \alpha_r\}$ or an element of $\mathbb{Z}$. The relation is to be interpreted as $\prod_{j=1}^m A_j^{a_j^i} = 1$. (2) Every free variable α_j is assumed to take values in a finite integer range $S(2^{\ell_j}, 2^{\mu_j})$ where $\ell_j, \mu_j \geq 0$.*

We will write $R(\alpha_1, \ldots, \alpha_r)$ to denote the conjunction of all relations $\prod_{j=1}^m A_j^{a_j^i} = 1$ that are included in R.

Below we will design a 3-move honest verifier zero-knowledge proof (see e.g. [12]) that allows to a prover that knows witnesses $x_1, \ldots, x_r$ such that $R(x_1, \ldots, x_r) = 1$ to prove knowledge of these values. We start with a definition:

Definition 8 *A discrete-log relation set R is said to be* triangular, *if for each relation i containing the free variables $\alpha_w, \alpha_{w_1}, \ldots, \alpha_{w_b}$ it holds that the free-variables $\alpha_{w_1}, \ldots, \alpha_{w_b}$ were contained in relations $1, \ldots, i-1$.*

The 3-move proof of knowledge is presented in figure 1. The following auxiliary lemma will be useful in proving the properties of the protocol.

Lemma 9 *Consider a fixed $x \in S(2^\ell, 2^\mu)$ and the random variables $t \in_R \pm\{0,1\}^{\epsilon(\mu+k)}$, $c \in_R \{0,1\}^k$. It holds that the random variable $\hat{s} = t - c(x - 2^\ell)$ is statistically indistinguishable from the random variable $s \in_R \pm\{0,1\}^{\epsilon(\mu+k)}$. The parameter assumption required for statistical indistinguishability (assuming k is the security parameter and ϵ, μ are functions in k) is that $(\epsilon - 1)(\mu + k) = \omega(\log k)$; in particular this forces $\epsilon - 1$ to be an asymptotically larger function in k than any function of the form* $\log k/(\mu(k) + k)$.

[1] Efficient factorization techniques are known when at least $\lceil \nu/3 \rceil$ bits of the prime factors of n are known, [15].

Proof of knowledge for a Discrete-Log Relation Set R

objects $A_1, \ldots, A_m$, r free-variables $\alpha_1, \ldots, \alpha_r$, parameters: $\epsilon > 1, k \in \mathbb{N}$,

Each variable α_j takes values in the range $S(2^{\ell_j}, 2^{\mu_j})$

$\mathcal{P}$ proves knowledge of the witnesses $x_j \in S(2^{\ell_j}, 2^{\epsilon(\mu_j+k)+2})$ s.t. $R(x_1, \ldots, x_r) = 1$

$\mathcal{P}$		$\mathcal{V}$
for $w \in \{1, \ldots, r\}$ select $t_w \in_R \pm\{0,1\}^{\epsilon(\mu_w+k)}$		
for $i \in \{1, \ldots, z\}$ set $B_i = \prod_{j:\exists w, a^i_j = \alpha_w} A_j^{t_w}$	$\xrightarrow{B_1,\ldots,B_z}$	$c \in_R \{0,1\}^k$
	$\xleftarrow{c}$	
for $w \in \{1, \ldots, r\}$ set $s_w = t_w - c \cdot (x_w - 2^{\ell_w})$	$\xrightarrow{s_1,\ldots,s_r}$	Verify:
		for $w \in \{1, \ldots, r\}$
		$s_w \in_? \pm\{0,1\}^{\epsilon(\mu_w+k)+1}$
		for $i \in \{1, \ldots, z\}$
		$\prod_{j:\exists w, a^i_j = \alpha_w} A_j^{s_w} \stackrel{?}{=} B_i(\prod_{j:a^i_j \in \mathbb{Z}} A_j^{a^i_j} \prod_{j:\exists w, a^i_j = \alpha_w} A_j^{2^{\ell_w}})^c$

Figure 1: *Proof of Knowledge for a Discrete-Log relation set R.*

Proof. We will denote by $\mathcal{D}_{\mathsf{a}}$ the distribution of the random variable s and by $\mathcal{D}_{\mathsf{s}}$ the distribution of $\hat{s} = t - c(x - 2^\ell)$. Assume that the support of the two random variables is $\mathbb{Z}$.

- Regarding $\mathcal{D}_{\mathsf{a}}$ observe that a certain s_0 in $\pm\{0,1\}^{\epsilon(\mu+k)}$ has probability of being selected equal to $\frac{1}{2^{\epsilon(\mu+k)+1}}$ (uniform probability distribution). Any $s_0 \notin \pm\{0,1\}^{\epsilon(\mu+k)}$ has probability 0.
- Regarding $\mathcal{D}_{\mathsf{s}}$ observe that a certain s_0 has the following probabilities of being selected:
  1. For $-2^{\epsilon(\mu+k)} + 2^{\mu+k} < s_0 < 2^{\epsilon(\mu+k)} - 2^{\mu+k}$ for each of the 2^k different c_0's we can find a t_0 such that $s_0 = t_0 - c_0(x - 2^\ell)$, as a result the probability of obtaining the given s_0 according to $\mathcal{D}_{\mathsf{s}}$ is $\frac{2^k}{2^k 2^{\epsilon(\mu+k)+1}} = \frac{1}{2^{\epsilon(\mu+k)+1}}$.
  2. For $s_0 \geq 2^{\epsilon(\mu+k)} + 2^{\mu+k}$ or $s_0 \leq -2^{\epsilon(\mu+k)} - 2^{\mu+k}$ the probability of obtaining s_0 according to $\mathcal{D}_{\mathsf{s}}$ is 0 (it is impossible to solve the equation $s_0 = t_0 - c_0 x$ for t_0, c_0 in their respective domains).
  3. For the remaining $s_0 \in \mathbb{Z}$ the probability of selecting them according to $\mathcal{D}_{\mathsf{s}}$ is smaller than $\frac{1}{2^{\epsilon(\mu+k)+1}}$ but potentially higher than 0.

It is clear from the above that the absolute difference between the probability of a certain s_0 according to $\mathcal{D}_{\mathsf{s}}$ and $\mathcal{D}_{\mathsf{a}}$ is 0 for the integer ranges of cases 1 and 2 above. The distributions $\mathcal{D}_{\mathsf{s}}$ and $\mathcal{D}_{\mathsf{a}}$ will accumulate some statistical distance though due to their different behavior for s_0 that belong to the integer range specified in item 3. In this case, for a specific s_0, distribution $\mathcal{D}_{\mathsf{a}}$ assigns probability either 0 or $\frac{1}{2^{\epsilon(\mu+k)+1}}$ whereas distribution $\mathcal{D}_{\mathsf{s}}$ assigns probability that belongs in the real interval $[0, \frac{1}{2^{\epsilon(\mu+k)+1}})$. Clearly, in the worst case the absolute difference will be $\frac{1}{2^{\epsilon(\mu+k)+1}}$ The number of elements s_0 of case 3, are $2^{\mu+k+2}$ thus it follows that the statistical distance of the distributions $\mathcal{D}_{\mathsf{s}}$ and $\mathcal{D}_{\mathsf{a}}$ cannot be greater than

$$\frac{2^{\mu+k+2}}{2^{\epsilon(\mu+k)+1}} = \frac{1}{2^{(\epsilon-1)(\mu+k)-1}}$$

Clearly under the assumption that $(\epsilon - 1)(\mu + k) = \omega(\log k)$ the above distance is negligible in k and as a result the distributions $\mathcal{D}_{\mathsf{actual}}$ and $\mathcal{D}_{\mathsf{sim}}$ are statistically indistinguishable. □

Theorem 10 *For any triangular discrete-log relation set R the 3-move protocol of figure 1 is a honest verifier zero-knowledge proof that can be used by a party (prover) knowing a witness for R to prove knowledge of the witness to a second party (verifier).*

We remark that the proof assumes that the prover is incapable of solving the Strong-RSA problem; under this assumption the cheating probability of the prover is $1/2^k$. Regarding the length of the proof we note that the proof requires the first communication flow from the prover to the verifier to be of size $z\ QR(n)$ elements (where z is the number of relations in R) and the second communication flow from the prover to the verifier to be of total bit-length $\sum_{w=1}^{r}(\epsilon(\mu_w + k) + 1)$.

Proof. Let $\overline{\alpha} = \{\alpha_1, \ldots, \alpha_m\}$ be the set of the free-variables of R and $x_1, \ldots, x_r$ the witness for R that is in the knowledge of the prover; further assume that $\langle a_1^i, \ldots, a_m^i \rangle$ is the i-th relation of R and that there are z relations in R.

In the first move the prover selects $t_w \in_R \pm\{0,1\}^{\epsilon(\mu_w+k)}$ for all $w = 1, \ldots, r$. Then, for each relation $i \in \{1, \ldots, z\}$ The prover computes the value $B_i = \prod_{j:\exists w, a_j^i = \alpha_w} A_j^{t_w}$ and transmits all values $B_1, \ldots, B_z$ to the verifier.

The verifier selects $c \in_R \{0,1\}^k$ and transmits c to the prover. The prover in response prepares the values $s_w = t_w - c(x_w - 2^{\ell_w})$ for $w = 1, \ldots, r$ and transmits them to the verifier.

The verifier performs the following checks in order to accept the proof:

- Checks $s_w \in \pm\{0,1\}^{\epsilon(\mu_w+k)+1}$ for all $w = 1, \ldots, r$.
- Tests the equalities for $i = 1, \ldots, z$,

$$\prod_{j:\exists w, a_j^i = \alpha_w} A_j^{s_w} \stackrel{?}{=} B_i \Big(\prod_{j: a_j^i \notin \overline{\alpha}} A_j^{a_j^i} \prod_{j:\exists w, a_j^i = \alpha_w} A_j^{2^{\ell_w}} \Big)^c$$

Next we argue for the three properties of the above protocol, completeness, soundness and honest verifier zero-knowledge.

1. Completeness follows easily by inspection. In particular observe that if $c \in \{0,1\}^k$, it follows that each $c(x_w - 2^{\ell_w}) \in \pm\{0,1\}^{\mu_w+k}$ (recall that $x_w \in S(2^{\ell_w}, 2^{\mu_w})$), and as a result $t_w - c(x_w - 2^{\ell_w}) \in \pm\{0,1\}^{\epsilon(\mu_w+k)+1}$ always.

2. Regarding soundness we let $\langle B_1, \ldots, B_z, c, s_1, \ldots, s_r \rangle$ and $\langle B_1, \ldots, B_z, c^*, s_1^*, \ldots, s_r^* \rangle$ be two accepting conversations. between a prover and the (honest) verifier with $c \neq c^*$.

   First observe that due to the triangularity property it holds that the first relationship in R involves only a single free variable, say α_{w_0} at locations $\mathcal{J}_{w_0} \subseteq \{1, \ldots, m\}$. Now let us denote by $A = \prod_{j \in \mathcal{J}_{w_0}} A_j$.

   Because the two conversations are accepting it follows that:

$$A^{s_{w_0} - s_{w_0}^*} = (A^{2^{\ell_{w_0}}} \prod_{j \notin \mathcal{J}_{w_0}} A_j^{a_j^i})^{c-c^*}$$

   Next, we compute $\delta = \gcd(s_{w_0} - s_{w_0}^*, c - c^*)$ and α, β such that $\delta = \alpha(s_{w_0} - s_{w_0}^*) + \beta(c - c^*)$. Observe that with very high probability it should hold that δ has no common divisor with the order of G (otherwise we can turn the prover into a factorization algorithm) and as a result it follows that:

$$A^{\frac{s_{w_0}-s^*_{w_0}}{\delta}} = (A^{2^{\ell_{w_0}}} \prod_{j \notin \mathcal{J}_{w_0}} A_j^{a^i_j})^{\frac{c-c^*}{\delta}}$$

Now observe that,

$$A = A^{\alpha\frac{s_{w_0}-s^*_{w_0}}{\delta}+\beta\frac{c-c^*}{\delta}} = ((A^{2^{\ell_{w_0}}} \prod_{j \notin \mathcal{J}_{w_0}} A_j^{a^i_j})^{\alpha} A^{\beta})^{\frac{c-c^*}{\delta}}$$

Observe now that if $c - c^* > \delta$ it follows easily that we can turn the prover into an algorithm that solves a given strong-RSA challenge (indeed, given the Strong-RSA challenge K we would select at random the elements of $\{A_j \mid j \in \mathcal{J}_{w_0}\}$ with the condition $\prod_{j\in\mathcal{J}_{w_0}} A_j = K$ and as shown above we would obtain the $\frac{c-c^*}{\delta}$-root of K).

It follows that $c - c^* = \delta$ and as a result it follows that:

$$(\prod_{j\in\mathcal{J}_{w_0}} A_j)^{\frac{s^*_{w_0}-s_{w_0}}{\delta}+2^{\ell_{w_0}}} \prod_{j\notin\mathcal{J}_{w_0}} A_j^{a^i_j} = 1$$

The above equality implies that we have constructed the witness for the w_0-th free variable $\tilde{x}_{w_0} = \frac{s^*_{w_0}-s_{w_0}}{\delta} + 2^{\ell_{w_0}}$.

Observe that since $s_{w_0}, s^*_{w_0} \in \pm\{0,1\}^{\epsilon(\mu_w+k)+1}$ it follows that $s^*_{w_0} - s_{w_0} \in \pm\{0,1\}^{\epsilon(\mu_w+k)+2}$ and also that $\frac{s^*_{w_0}-s_{w_0}}{\delta} \in \pm\{0,1\}^{\epsilon(\mu_w+k)+2}$ As a result $\tilde{x}_{w_0} \in S(2^{\ell_{w_0}}, 2^{\epsilon(\mu_w+k)+2})$.

Now assume that we have processed all the relations with index less than i. We process the i-th relation as follows: first, observe that due to triangularity, relation i involves the variables $\alpha_{w_0}, \alpha_{w_1}, \ldots, \alpha_{w_b}$ so that the variables $\alpha_{w_1}, \ldots, \alpha_{w_b}$ were contained already in the previous relations. It follows by an inductive argument that we have already constructed witnesses for the free-variables $\tilde{x}_{w_1} = \frac{s^*_{w_1}-s_{w_1}}{\delta} + 2^{\ell_{w_1}}, \ldots, \tilde{x}_{w_b} = \frac{s^*_{w_b}-s_{w_b}}{\delta} + 2^{\ell_{w_b}}$. As before let $\mathcal{J}_{w_0}$ be the set of locations of the i-th relation that involve the α_{w_0} variable and similarly define $\mathcal{J}_{w_1}, \ldots, \mathcal{J}_{w_b}$; we also set $\mathcal{J} = \cup_{\imath=0,\ldots,b}\mathcal{J}_{w_\imath}$. Furthermore let $\hat{A}_{w_\imath} = \prod_{j\in\mathcal{J}_{w_\imath}} A_j$ for $\imath = 0,\ldots,b$. Since the two conversations are accepting, it follows that

$$\prod_{j:\exists w, a^i_j=\alpha_w} A_j^{s_w-s^*_w} = (\prod_{j:a^i_j\in\mathbb{Z}} A_j^{a^i_j} \prod_{j:\exists w, a^i_j=\alpha_w} A_j^{2^{\ell_w}})^{c-c^*}$$

Now observe that due to the conditions that we have for the i-th relation it holds that

$$\prod_{j:\exists w, a^i_j=\alpha_w} A_j^{s_w-s^*_w} = \hat{A}_{w_0}^{s_{w_0}-s^*_{w_0}} \hat{A}_{w_1}^{s_{w_1}-s^*_{w_1}} \ldots \hat{A}_{w_b}^{s_{w_b}-s^*_{w_b}}$$

and

$$(\prod_{j:a^i_j\in\mathbb{Z}} A_j^{a^i_j} \prod_{j:\exists w, a^i_j=\alpha_w} A_j^{2^{\ell_w}})^{c-c^*} = (\hat{A}_{w_0}^{2^{\ell_{w_0}}} \hat{A}_{w_1}^{2^{\ell_{w_1}}} \ldots \hat{A}_{w_b}^{2^{\ell_{w_b}}})^{c-c^*} (\prod_{j\notin\mathcal{J}} A_j^{a^i_j})^{c-c^*}$$

From the above we obtain that:

$$\hat{A}_{w_0}^{s_{w_0}-s^*_{w_0}} = \hat{A}_{w_0}^{(c-c^*)2^{\ell w_0}} \hat{A}_{w_1}^{s^*_{w_1}-s_{w_1}+(c-c^*)2^{\ell w_1}} \ldots \hat{A}_{w_b}^{s^*_{w_b}-s_{w_b}+(c-c^*)2^{\ell w_b}} (\prod_{j\notin\mathcal{J}} A_j^{a^i_j})^{c-c^*}$$

Now due to the fact that for all $\imath \in \{1,\ldots,b\}$ it holds that $\tilde{x}_{w_\imath} = \frac{s^*_{w_\imath}-s_{w_\imath}}{\delta} + 2^{\ell_{w_\imath}}$ with $\delta = c - c^*$ we obtain that $s^*_{w_\imath} - s_{w_\imath} + 2^{\ell_{w_\imath}}(c-c^*) = \tilde{x}_{w_\imath}(c-c^*)$ and as a result,

$$\hat{A}_{w_0}^{s_{w_0}-s^*_{w_0}} = (\hat{A}_{w_0}^{2^{\ell w_0}} \hat{A}_{w_1}^{\tilde{x}_{w_1}} \ldots \hat{A}_{w_b}^{\tilde{x}_{w_b}} \prod_{j\notin\mathcal{J}} A_j^{a^i_j})^{c-c^*}$$

Now we define $\delta' = \gcd(s_{w_0} - s^*_{w_0}, c - c^*)$ and α', β' such that $\delta' = \alpha'(s_{w_0} - s^*_{w_0}) + \beta'(c - c^*)$ and with an identical argument as in the case of the first relation, we conclude that it must be the case that $\delta' = c - c^* = \delta$. Moreover, as in the case of the first relation, we construct the witness $\tilde{x}_{w_0} = \frac{s^*_{w_0}-s_{w_0}}{\delta} + 2^{\ell_{w_0}}$ for the free-variable α_{w_0}; it is easy to verify that $\tilde{x}_{w_0} \in S(2^{\ell_{w_0}}, 2^{\epsilon(\mu_{w_0}+k)+2})$.

3. Regarding honest verifier zero-knowledge, we will describe a simulator for protocol transcripts between the honest prover and the honest verifier. The simulator operates as follows: it selects $c \in_R \{0,1\}^k$ and for $j = 1,\ldots,r$, $\hat{s}_j \in_R \pm\{0,1\}^{\epsilon(\mu_j+k)}$ and it computes for $i = 1,\ldots,z$ the values $\hat{B}_i = \prod_{j:\exists w, a^i_j=\alpha_w} A_j^{s_w} (\prod_{j:a^i_j\in\mathbb{Z}} A_j^{a^i_j} \prod_{j:\exists w, a^i_j=\alpha_w} A_j^{2^{\mu_w}})^{-c}$. The simulator outputs the transcript
$$\langle \hat{B}_1,\ldots,\hat{B}_z, c, \hat{s}_1,\ldots,\hat{s}_r\rangle$$
Then we need to show that the simulated transcripts are statistically indistinsguishable from transcripts that are generated in conversations between the honest prover and the honest verifier. This boils down to calculating the statistical distance between the random variable s computed as $t - c(x - 2^\ell)$ for a fixed $x \in S(2^\ell, 2^\mu)$ and $t \in_R \pm\{0,1\}^{\epsilon(\mu+k)}$ and $c \in_R \{0,1\}^k$ to the random variable $\hat{s} \in_R \pm\{0,1\}^{\epsilon(\mu+k)}$. This follows immediately from lemma 9.

$\square$

4.1 Examples of Discrete-Log Relation Sets

Proving knowledge of a witness for a discrete-log relation sets can be used in a variety of settings. We list some of them below:

Proving knowledge of a discrete-logarithm over a group of unknown order. Consider the base elements A_1, A_2 and the free variable $\alpha \in S(1, 2^{\lfloor \log \#G \rfloor - 1})$; we consider the discrete-log relation set with a single relation $A_1^\alpha A_2^{-1} = 1$. It is immediate that this relation set is triangular. Furthermore, it is easy to see that a proof of knowledge for the above discrete-log relation set allows one to prove knowledge of the discrete-logarithm of A_2 base A_1.

Proving knowledge of a discrete-logarithm inside an interval. Following the above description but the variable α will be restricted to a range $S(2^\ell, 2^\mu)$. Note that the soundness property will only guarantee that the constructed witness lies in the extended sphere $S(2^\ell, 2^{\epsilon(\mu+k)+2})$; tighter intervals can be achieved by increasing the size of the zero-knowledge proof, see e.g. [5].

Proving knowledge of a committed discrete-logarithm representation. Let A_1, A_2, A_3, A_4, A_5 be the objects and consider the free-variables $\alpha_1, \alpha_2 \in S(1, 2^{\lfloor \log \#G \rfloor - 1})$. we define two relations $A_1^{\alpha_1} A_2^{-1} = 1$ and $A_3^{\alpha_1} A_4^{\alpha_2} A_5^{-1} = 1$.

First, note that the above discrete-log relation set is triangular. Furthermore it allows a proof of knowledge for a representation of A_5 over the bases A_3, A_4.

4.2 Spheres and Innerspheres

As demonstrated in the proof of knowledge of a discrete-log relation set, if a witness belongs to a certain sphere $S(2^\ell, 2^\mu)$ we are able to enforce the membership of the witness to an extended sphere $S(2^\ell, 2^{\epsilon(\mu+k)})$ based on the parameters k and ϵ (recall that the parameters k and ϵ calibrate the prover cheating probability and the statistical distance of the zero-knowledge simulator in the zero-knowledge argument, respectively).

Frequently, sphere relationships will be immensely useful in proving properties of our constructs. If proof of discrete-log relation sets are to be employed in proving the security of more complex structures and enforcing various sphere relationships, we should take into account the fact that the witness interval is slightly expanded because of the way the soundness proof is performed. Note that the tightness can be increased by employing more complex interval proofs (see [5]); nevertheless the amount of tightness we achieve is sufficient for our setting. Given a sphere $S(2^\ell, 2^\mu)$ we define its innersphere for parameters ϵ, k as follows $S^k_\epsilon(2^\ell, 2^\mu) =_{\mathsf{df}} S(2^\ell, 2^{\frac{\mu-2}{\epsilon}-k})$. If a witness $x \in S^k_\epsilon(2^\ell, 2^\mu)$ is employed in a proof of discrete-log relation set, then the verifier is guaranteed that the prover possesses a witness in $S(2^\ell, 2^\mu)$.

5 Discrete Log Representations of Arbitrary Powers

In this section we introduce and present some basic facts about "discrete log representations of arbitrary powers" inside the set of Quadratic Residues $QR(n)$ where n is a composite modulus with $n = pq = (2p'+1)(2q'+1)$ where all p, q, p', q' are prime. Let $\nu = \lceil \log p'q' \rceil$.

We will define three spheres $\Lambda, \Gamma, \mathrm{M}$ inside the set $\{0, \ldots, 2^\nu - 1\}$ so that the following conditions are satisfied:

S1. $(\min\Gamma)^2 > \max\Gamma$.

S2. M has size approximately equal to $2^{\lceil \nu/2 \rceil}$.

S3. $\min\Gamma > \max\mathrm{M}\max\Lambda + \max\Lambda + \max\mathrm{M}$

The above set of conditions is attainable as shown by the following possible selection: for simplicity, we assume that ν is divisible by 4:

- $\Lambda = S(2^{\frac{\nu}{4}-1}, 2^{\frac{\nu}{4}-1})$, note that $\#\Lambda = 2^{\frac{\nu}{4}} - 1$ and $\max\Lambda = 2^{\frac{\nu}{4}} - 1$.
- $\mathrm{M} = S(2^{\frac{\nu}{2}-1}, 2^{\frac{\nu}{2}-1})$, note that $\#\mathrm{M} = 2^{\frac{\nu}{2}} - 1$ and $\max\mathrm{M} = 2^{\frac{\nu}{2}} - 1$.
- $\Gamma = S(2^{\frac{3\nu}{4}} + 2^{\frac{\nu}{4}-1}, 2^{\frac{\nu}{4}-1})$, note that $\#\Gamma = 2^{\frac{\nu}{4}} - 1$, $\min\Gamma = 2^{\frac{3\nu}{4}} + 1 > \max\Lambda\max\mathrm{M} + \max\Lambda + \max\mathrm{M} = 2^{\frac{3\nu}{4}} - 1$.

In the exposition below we use some fixed values $a_0, a, b \in QR(n)$.

Definition 11 *A discrete-log representation of an arbitrary power is a tuple $\langle A, e : x, x' \rangle$ so that it holds $A^e = a_0 a^x b^{x'}$ with $x, x' \in \Lambda$ and $e \in \Gamma$.*

In this work we will be interested in the following computational problem:

$\diamond$ *The One-more Representation Problem.* Given n, a_0, a, b and K discrete-log representations of arbitrary powers find "one-more" discrete-log representation of an arbitrary power inside $QR(n)$.

The main result of this section, stated in the theorem below, establishes that solving the One-more representation problem cannot be substantially easier than solving the Strong-RSA problem.

Theorem 12 *Fix $a_0, a, b \in QR(n)$ and spheres $\Lambda, \mathrm{M}, \Gamma$ satisfying the above properties. Let $\mathcal{M}$ be a* PPT *algorithm that given K discrete-log representations of arbitrary powers inside $QR(n)$ it outputs a different discrete-log representation of an arbitrary power inside $QR(n)$ with non-negligible probability. Then, the Strong-RSA problem can be solved with non-negligible probability. In particular if α is the success probability of $\mathcal{M}$, the Strong-RSA problem can be solved with success probability at least $\alpha/2K$.*

Proof. Suppose we are given an instance of the Strong-RSA problem $\langle n, z\rangle$, where n is a composite modulus and $z \in \mathbb{Z}_n^*$; we will show how to use $\mathcal{M}$ to construct a pair $\langle u, e\rangle$ such that $u^e = z(\mathrm{mod} n)$.

Below we will describe an algorithm that solves the given Strong-RSA instance. The algorithm is comprised of four games that are played at random. The major issue in all the games is the construction of the discrete-log representations of random powers that are handed to the adversary and the relationship of the output of the adversary to these representations. All four games construct somehow the discrete-log representations of random powers $\langle A_i, e_i : x_i, x_i'\rangle$ for $i = 1, \dots, K$, and obtain the output representation of the adversary $\langle \hat{A}, \hat{e} : \hat{x}, \hat{x}'\rangle$. Each game may fail according to the following specifications:

1. Game 1 will fail if $\hat{e}$ has a non-trivial common divisor with any of the values $e_1, \dots, e_K$.
2. Game 2 will fail if (i) $\hat{e}$ is relatively prime to all values $e_1, \dots, e_K$, and (ii) it is not possible to find a $j \in \{1, \dots, K\}$ for which it holds that $a^{\hat{x}-x_j} b^{\hat{x}'-x_j'} = 1$ with either $\hat{x} \neq x_j$ or $\hat{x}' \neq x_j'$.
3. Game 3 will fail if (i) $\hat{e}$ is relatively prime to all values $e_1, \dots, e_K$, and (ii) for a pre-selected value j it does not hold that e_j divides $\hat{e}$ and it does not hold that $x_j = \hat{x}$ and $x_j' = \hat{x}'$.
4. Game 4 will fail if (i) $\hat{e}$ is relatively prime to all values $e_1, \dots, e_K$, and (ii) for a pre-selected value j it does not hold that e_j divides $\hat{e}$ and it does not hold that $a^{\hat{x}-x_j} b^{\hat{x}'-x_j'} \neq 1$.

Observe that playing the above games at random covers all possible behaviors of the algorithm $\mathcal{M}$ with respect to the relation of the output discrete-log representation to the given ones. The detailed description of the four games follows below.
Game 1.

1. Select random $x_1, \dots, x_K, x_1', \dots, x_K' \in \Lambda$ and $e_1, \dots, e_K \in \Gamma$.
2. Set $a = z^{e_1 \dots e_K}(\mathrm{mod} n)$, $a_0 = a^r$ and $b = a^{r'}$ where r, r' are random integers in M. Observe that due to theorem 6 and the properties of the selected sphere M (S2) it holds that the values a_0, a, b are indistinguishable from random elements of $QR(n)$.
3. Compute $A_i = z^{(x_i + r + r' x_i')\frac{e_1 \dots e_K}{e_i}}(\mathrm{mod} n)$, for all $i = 1, \dots, K$. Observe that $A_i^{e_i} = a_0 a^{x_i} b^{x_i'}$ for all $i = 1, \dots, K$, i.e., $\langle A_i, e_i : x_i, x_i'\rangle$ are discrete-log representations of arbitrary powers inside $QR(n)$ over a_0, a, b.

4. Simulate $\mathcal{M}$ by providing the K discrete-log representations of arbitrary powers computed above and let $\langle \hat{A}, \hat{e} : \hat{x}, \hat{x}' \rangle$ denote the output of $\mathcal{M}$ which is a discrete-log representation of an arbitrary power (distinct from $\langle A_i, e_i : x_i, x_i' \rangle$ for $i = 1, \ldots, K$).

5. If $\hat{e}$ has a non-trivial common divisor with any of $e_1, \ldots, e_K$ abort.

   Now observe that $\hat{A}^{\hat{e}} = a_0 a^{\hat{x}} b^{\hat{x}'}$; based on our selection of the values a_0, a, b it is easy to see that $a_0 a^{\hat{x}} b^{\hat{x}'} = z^{(r+\hat{x}+r'\hat{x}')e_1 \ldots e_K}$. Set $\tilde{e} := (r + \hat{x} + r'\hat{x}')e_1 \ldots e_K$. Next we compute $\delta := \gcd(\tilde{e}, \hat{e}) = \gcd(r + \hat{x} + r'\hat{x}', \hat{e})$. In the case δ has a non-trivial common divisor with the order of $QR(n)$ we can factor n and thus compute the solution of the given Strong-RSA instance; in the other case, we compute α, β such that $\delta = \alpha\tilde{e} + \beta\hat{e}$ and we have

   $$z = z^{\alpha\frac{\tilde{e}}{\delta} + \beta\frac{\hat{e}}{\delta}} = (z^{\frac{\tilde{e}}{\delta}})^{\alpha} z^{\beta\frac{\hat{e}}{\delta}} = (\hat{A}^{\frac{\hat{e}}{\delta}})^{\alpha} z^{\beta\frac{\hat{e}}{\delta}} = (\hat{A}^{\alpha} z^{\beta})^{\frac{\hat{e}}{\delta}}$$

   Clearly if $\delta < \hat{e}$ we are done, since the above equations reveals a solution to the given Strong-RSA instance. Observe that $\delta \leq r + \hat{x} + r'\hat{x}'$, and due to property S3 of the sphere selection it holds that $\delta < e$ for any $e \in \Gamma$ and as a result $\delta < \hat{e}$. It follows that $u := z^{\alpha}\hat{A}^{\beta}$ and $e := \frac{\hat{e}}{\delta}$ is a solution for the Strong-RSA instance $\langle n, z \rangle$.

Game 2.

1. We flip a random coin bit $\in_R \{0, 1\}$ and if bit $= 0$ we set $v_a = z$ and $v_b = z^{r'}$, otherwise we set $v_a = z^{r'}$ and $v_b = z$, where $r' \in_R$ M.

2. Select random $x_1, \ldots, x_K, x_1', \ldots, x_K' \in \Lambda$ and $e_1, \ldots, e_K \in \Gamma$.

3. Set $a = v_a^{e_1 \ldots e_K} (\mathrm{mod} n)$, $a_0 = a^r$ and $b = v_b^{e_1 \ldots e_K} (\mathrm{mod} n)$ where r is a random integer in M. Due to sphere selection property S2 and theorem 6 it holds that a_0, a, b are indistinguishable from random elements of $QR(n)$.

4. Compute $A_i = (a_0 v_a^{x_i} v_b^{x_i'})^{\frac{e_1 \ldots e_K}{e_i}} (\mathrm{mod} n)$, for all $i = 1, \ldots, K$. Observe that $A_i^{e_i} = a_0 a^{x_i} b^{x_i'}$ for all $i = 1, \ldots, K$, i.e., $\langle A_i, e_i : x_i, x_i' \rangle$ are discrete-log representations of arbitrary powers inside $QR(n)$ over a_0, a, b.

5. Simulate $\mathcal{M}$ by providing the K discrete-log representations of arbitrary powers computed above and let $\langle \hat{A}, \hat{e} : \hat{x}, \hat{x}' \rangle$ denote the output of $\mathcal{M}$ which is a discrete-log representation of an arbitrary power (distinct from $\langle A_i, e_i : x_i, x_i' \rangle$ for $i = 1, \ldots, K$).

6. If $\hat{e}$ is relatively prime with all $e_1, \ldots, e_K$ game 2 aborts.

7. For all j we compute $\Delta = \hat{x} - x_j$ and $\Delta' = -\hat{x}' + x_j'$; We check whether $a^{\Delta} = b^{\Delta'}$. If we find no j for which the test passes or if we find only j's for which it holds that $\Delta = \Delta' = 0$ we abort game 2. Below we assume that j satisfies $a^{\Delta} = b^{\Delta'}$ so that not both Δ, Δ' are 0. Now observe that $v_a^{e_1 \ldots e_K \Delta} = v_b^{e_1 \ldots e_K \Delta'}$ which is equivalent to $v_a^{\Delta} = v_b^{\Delta'}$ since the primes $e_1, \ldots, e_K$ do not divide the order of $QR(n)$.

8. (Case a) suppose that $\Delta \neq 0$ and $\Delta' \neq 0$. We compute $\delta = \gcd(\Delta, \Delta')$ and α, β such that $\delta = \alpha\Delta + \beta\Delta'$.

   Observe that if we come up with a δ that divides the order of $QR(n)$ we can factor n and as a result solve the Strong-RSA instance n, z. In the other case, it will hold that $v_a^{\frac{\Delta}{\delta}} = v_b^{\frac{\Delta'}{\delta}}$

Now observe that,

$$v_a = (v_a^{\frac{\Delta}{\delta}})^{\alpha}(v_a^{\frac{\Delta'}{\delta}})^{\beta} = (v_b^{\frac{\Delta'}{\delta}})^{\alpha}(v_a^{\frac{\Delta'}{\delta}})^{\beta} = (v_b^{\alpha} v_a^{\beta})^{\frac{\Delta'}{\delta}}$$

Now if $\Delta' > \delta$ it follows easily that since v_a is be equal to z with probability $1/2$, the above relation reveals a solution for the Strong-RSA instance n, z.

In the case $\Delta' = \delta$ it follows that Δ' divides Δ. Again if Δ' has a non-trivial common divisor with the order of the group $QR(n)$ we can factor n and thus solve the Strong-RSA instance n, z.

In the other case observe that it will hold that $v_a^{\frac{\Delta}{\Delta'}} = v_b$ and since $v_b = z$ with probability $1/2$ we will solve the given Strong-RSA instance, unless $\Delta = \Delta'$, something hardly possible since this would make $v_a = v_b$, which is a negligible probability event.

9. (Case b). Suppose that $\Delta = 0$. It follows that $v_b^{\Delta'} = 1$ and as a result either Δ' has a non-trivial divisor with the order of $QR(n)$ that allows us to factor n and thus solve the Strong-RSA instance n, z or it also holds that $\Delta' = 0$. But this cannot be true as this case has been excluded. The case when $\Delta' = 0$ is similar.

Game 3.

1. We select random $x_1, \ldots, x_K, x'_1, \ldots, x'_K \in \Lambda$ and $e_1, \ldots, e_K \in \Gamma$.

2. Select $j \in_R \{1, \ldots, K\}$ and set $a = v_a^{\frac{e_1 \cdots e_K}{e_j}} \pmod n$, $a_0 = A_j^{e_j}/(a^{x_j} b^{x'_j})$ and $b = v_b^{\frac{e_1 \cdots e_K}{e_j}} \pmod n$ where $A_j = z^{\frac{e_1 \cdots e_K}{e_j}} (\mathrm{mod} n)$ and $v_a = z^r$ and $v_b = v_a^{r'}$ with r, r' random integers in M. Clearly a, b are indistinguishable from random elements of $QR(n)$ given the sphere selection property S2, theorem 6 and the fact that $e_1, \ldots, e_K$ are all relatively prime to the order of $QR(n)$ (and this fact holds for a fixed z). Moreover since z is a free selected element of $QR(n)$, also a_0 is uniformly distributed over $QR(n)$ (since $e_1, \ldots, e_K$ are all relatively prime to the order of $QR(n)$).

3. Compute $A_i = (z^{e_j} v_a^{x_i - x_j} v_b^{x'_i - x'_j})^{\frac{e_1 \cdots e_K}{e_i e_j}} (\mathrm{mod} n)$, for all $i = 1, \ldots, j-1, j+1, \ldots K$. Observe that $A_i^{e_i} = a_0 a^{x_i} b^{x'_i}$ for all $i = 1, \ldots, K$, It follows that $\langle A_i, e_i : x_i, x'_i \rangle$ are discrete-log representations of arbitrary powers for $i = 1, \ldots, K$.

4. Simulate $\mathcal{M}$ by providing the K discrete-log representations of arbitrary powers computed above and let $\langle \hat{A}, \hat{e} : \hat{x}, \hat{x}' \rangle$ denote the output of $\mathcal{M}$ which is a discrete-log representation of an arbitrary power (distinct from $\langle A_i, e_i : x_i, x'_i \rangle$ for $i = 1, \ldots, K$).

5. If $\hat{e}$ is relatively prime to e_j game 3 aborts.

6. If $x_j \neq \hat{x}$ or $x'_j \neq \hat{x}'$ then game 3 aborts. In the other case, observe that $A_j^{e_j} = \hat{A}^{\hat{e}}$. It follows that, $\hat{A}^{\hat{e}} = z^{e_1 \cdots e_K}$. We compute $\delta = \gcd(\hat{e}, e_1 \ldots e_K)$ and α, β such that $\delta = \alpha \hat{e} + \beta e_1 \ldots e_K$. In the case that δ has a non-trivial common divisor with the order of $QR(n)$ we can factor n and thus solve the given Strong RSA instance. In the other case it holds,

$$z = z^{\alpha \frac{\hat{e}}{\delta} + \beta \frac{e_1 \cdots e_K}{\delta}} = z^{\alpha \frac{\hat{e}}{\delta}} (z^{\frac{e_1 \cdots e_K}{\delta}})^{\beta} = (z^{\alpha} \hat{A}^{\beta})^{\frac{\hat{e}}{\delta}}$$

which yields a solution to the given Strong RSA instance unless $\delta = \hat{e}$. But this in turn means that $\hat{e}$ divides $e_1 \ldots e_K$ which implies that either (i) $\hat{e} = e_j$ which is not possible since in this case the two representations $\langle A_j, e_j : x_j, x'_j\rangle$ and $\langle \hat{A}, \hat{e} : \hat{x}, \hat{x}'\rangle$ are the same or (ii) $\hat{e} = e_j e_{j'}$ which is also not possible since in this case $\hat{e} \notin \Gamma$ (due to sphere selection property S1).

Game 4.

1. Select $x_1, \ldots, x_K, x'_1, \ldots, x'_K \in \Lambda$ and $e_1, \ldots, e_K \in \Gamma$.

2. Select $j \in_R \{1, \ldots, K\}$ and set $a = z^{\frac{e_1 \cdots e_K}{e_j}} \pmod{n}$, $a_0 = A_j^{e_j}/(a^{x_j} b^{x'_j})$ and $b = a^{r'}$ where $A_j = a^r$ and r, r' are random integers in M. As in game 3, we argue that a_0, a, b are indistinguishable from random elements of $QR(n)$.

3. Compute $A_i = z^{(x_i + r'x'_i + re_j - x_j - r'x'_j)\frac{e_1 \cdots e_K}{e_i e_j}} \pmod{n}$, for all $i = 1, \ldots, j-1, j+1, \ldots K$. Observe that $A_i^{e_i} = a_0 a^{x_i} b^{x'_i}$ for all $i = 1, \ldots, K$, It follows that $\langle A_i, e_i : x_i, x'_i\rangle$ are discrete-log representations of arbitrary powers for $i = 1, \ldots, K$.

4. Simulate $\mathcal{M}$ by providing the K discrete-log representations of arbitrary powers computed above and let $\langle \hat{A}, \hat{e} : \hat{x}, \hat{x}'\rangle$ denote the output of $\mathcal{M}$ which is a discrete-log representation of an arbitrary power (distinct from $\langle A_i, e_i : x_i, x'_i\rangle$ for $i = 1, \ldots, K$).

5. If it holds that $\gcd(\hat{e}, e_j) \neq e_j$ then the game aborts (note that if $\hat{e}$ has a non-trivial common divisor with some of $e_1, \ldots, e_K$ game 4 will abort with probability only $1/K$). In any other case, $\gcd(\hat{e}, e_j) = e_j$ (due to the fact that e_j is a prime number) and as a result there exists an integer t such that $\hat{e} = te_j$.

   Next we check whether $\hat{x} - x_j + r'(\hat{x}' - x'_j) \neq 0$ and in this case we proceed as follows:

   Let $Z = \hat{A}^t/A_j$. It cannot be the case that $Z = 1$; indeed if $Z = 1$ this means that $\hat{A}^t = A_j$ or equivalently that $\hat{A}^{\hat{e}} = A_j^{e_j}$ which implies that $a^{\hat{x}} b^{\hat{x}'} = a^{x_j} b^{x'_j}$ and $a^{\hat{x} - x_j + r'(\hat{x}' - x'_j)} = 1$. From this we obtain that $\gamma = \hat{x} - x_j + r'(\hat{x}' - x'_j) = 0$ since γ is a positive integer (due to sphere selection property S1) that is smaller than the order of the group (the only other case that this may happen is in the case that γ has a non-trivial common divisor with the order of the group $QR(n)$ from which we can factor n).

   $$Z^{e_j} = \left(\frac{\hat{A}^t}{A_j}\right)^{e_j} = \frac{\hat{A}^{\hat{e}}}{A_j^{e_j}} = \frac{a_0 a^{\hat{x}} b^{\hat{x}'}}{a_0 a^{x_j} b^{x'_j}} = a^{\hat{x} - x_j + r'(\hat{x}' - x'_j)} = a^{\gamma}$$

   Let $\tilde{e} := \gamma \frac{e_1 \ldots e_K}{e_j}$, it follows that $Z^{e_j} = z^{\tilde{e}}$. Suppose that $\delta := \gcd(e_j, |\tilde{e}|)$; it is easy to see that $\delta = \gcd(e_j, |\gamma|)$ since e_j is relatively prime with $\frac{e_1 \ldots e_K}{e_j}$. Due to the sphere selection property S3 it follows that $e_j > |\gamma|$ and since e_j is a prime number, it holds that $\delta = 1$. It follows then that we can find $\alpha, \beta \in \mathbb{Z}$ such that $1 = \alpha e_j + \beta \tilde{e}$ and as a result $z = z^{\alpha e_j + \beta \tilde{e}} = (z^{\alpha} Z^{\beta})^{e_j}$ It follows that $u := z^{\alpha} Z^{\beta}$ and $e := e_j$ is a solution for the Strong-RSA instance $\langle n, z\rangle$.

As a result, using $\mathcal{M}$ we can construct a probabilistic algorithm for Strong-RSA by playing the above two games. If α is the success probability of $\mathcal{M}$, it is easy to see that the above algorithm will solve the Strong-RSA problem with success probability at least $\alpha/2K$. $\square$

6 Non-adaptive Drawings of Random Powers

Consider the following game between two players A, B: player A wishes to select a random power a^x so that $x \in_R S(2^\ell, 2^\mu)$ where $a \in QR(n)$ with $n = pq = (2p' + 1)(2q' + 1)$. Player B wants to ensure that the value x is selected "non-adaptively" from its respective domain. The output specifications of the game is that player A returns x and that player B returns a^x. Player B is assumed to know the factorization of n. In this section we will carefully model and implement a protocol for achieving this two-player functionality. The reader is referred to [14] for a general discussion of modeling secure two-party computations.

In the ideal world the above game is played by two Interactive TM's (ITM's) A_0, B_0 and the help of a trusted third party ITM T following the specifications below. We note that we use a special symbol $\perp$ to denote failure (or unwillingness to participate); if an ITM terminates with any other output other than $\perp$ we say that it accepts; in the other case we say it rejects. From all the possible ways to implement A_0, B_0 one is considered to be the honest one; this will be marked as A_0^H, B_0^H and is also specified below.

0. The modulus n is available to all parties and its factorization is known to B_0. The sphere $S(2^\ell, 2^\mu)$ is also public and fixed.
1. A_0 sends a message in $\{\mathsf{go}, \perp\}$ to T. A_0^H transmits go.
2. B_0 sends a message in $\{\mathsf{go}, \perp\}$ to T. B_0^H transmits go.
3. If T receives go from both parties, it selects $x \in_R S(2^\ell, 2^\mu)$ and returns x to A_0; otherwise T transmits $\perp$ to both parties.
4. A_0 selects a value $C \in \mathbb{Z}_n^*$ and transmits either C or $\perp$ to T. A_0^H transmits $C = a^x \bmod n$.
5. T verifies that $a^x \equiv C (\mathrm{mod} n)$ and if this is the case it transmits C to both players. Otherwise, (or in the case A_0 transmitted $\perp$ in step 4), T transmits $\perp$ to both players. B_0^H terminates by returning C or $\perp$ in the case of receiving $\perp$ from T. Similarly A_0^H terminates by returning x, or $\perp$ in the case of receiving $\perp$ from T.

Let $\mathsf{Im}_T =_{\mathsf{df}} \langle A_0, B_0 \rangle$ be two ITM's that implement the above protocol with the help of the ITM T. We define by $\mathsf{OUT}_{A_0}^{\mathsf{Im}_T}(\mathsf{init}_A(\nu))$ and $\mathsf{OUT}_{B_0}^{\mathsf{Im}_T}(\mathsf{init}_B(\nu))$ be the output probability distributions of the two players. Note that $\mathsf{init}_A(\nu)$ contains the initialization string of player A which contains the modulus n, and the description of the sphere $S(2^\ell, 2^\mu)$; similarly $\mathsf{init}_B(\nu)$ is defined as $\mathsf{init}_A(\nu)$ with the addition of the factorization of n. Below we will use the notation $\mathsf{IDEAL}^{\mathsf{Im}_T}(in_A, in_B)$ to denote the pair $\langle \mathsf{OUT}_{A_0}^{\mathsf{Im}_T}(in_A), \mathsf{OUT}_{B_0}^{\mathsf{Im}_T}(in_B) \rangle$. Finally, we denote by Im_T^H the pair $\langle A_0^H, B_0^H \rangle$.

The goal of a protocol for non-adaptive drawing of random powers is the simulation of the trusted third party by the two players. Let $\mathsf{Im} = \langle A_1, B_1 \rangle$ be a two-player system of interactive TM's that implement the above game without interacting with the trusted third party T. As above we will denote by $\mathsf{OUT}_{A_1}^{\mathsf{Im}}(in_A)$ the output probability distribution A_1, and likewise for $\mathsf{OUT}_{B_1}^{\mathsf{Im}}(in_B)$. Also we denote by $\mathsf{REAL}^{\mathsf{Im}}(in_A, in_B)$ the concatenation of these two distributions.

Definition 13 *(Correctness) An implementation* $\mathsf{Im} = \langle A_1, B_1 \rangle$ *for non-adaptive drawings of random powers is* correct *if the following is true:*

$$\mathsf{REAL}^{\mathsf{Im}}(in_A, in_B) \approx \mathsf{IDEAL}^{\mathsf{Im}_T^H}(in_A, in_B)$$

where $in_A \leftarrow \mathsf{init}_A(\nu)$ and $in_B \leftarrow \mathsf{init}_B(\nu)$. Intuitively the above definition means that the implementation Im should achieve essentially the same output functionality for the two players as the ideal honest implementation.

Defining security is naturally a bit trickier as the two players may misbehave arbitrarily when executing the prescribed protocol implementation $\mathsf{Im} = \langle A_1, B_1\rangle$.

Definition 14 *(Security) An implementation $\mathsf{Im} = \langle A_1, B_1\rangle$ for non-adaptive drawings of random powers is* secure *if the following is true:*

$$\forall A_1^* \ \exists A_0^* \ \mathsf{REAL}^{\langle A_1^*, B_1\rangle}(in_A, in_B) \approx \mathsf{IDEAL}^{\langle A_0^*, B_0^H\rangle}(in_A, in_B)$$

$$\forall B_1^* \ \exists B_0^* \ \mathsf{REAL}^{\langle A_1, B_1^*\rangle}(in_A, in_B) \approx \mathsf{IDEAL}^{\langle A_0^H, B_0^*\rangle}(in_A, in_B)$$

where $in_A \leftarrow \mathsf{init}_A(\nu)$ and $in_B \leftarrow \mathsf{init}_B(\nu)$. Intuitively the above definition means that no matter what adversarial strategy is followed by either player it holds that it can be transformed to the ideal world setting without affecting the output distribution.

Having defined the goals, we now take on the task of designing an implementation Im without a trusted third party; below we denote by $\tilde{m} =_{\mathsf{df}} \#S(2^\ell, 2^\mu) = 2^{\mu+1} - 1$.

1. The two players read their inputs and initiate a protocol dialog.
2. Player A selects $\tilde{x} \in_R \mathbb{Z}_{\tilde{m}}, \tilde{r} \in_R \{0, \ldots, n^2 - 1\}$ and transmits to player B the value $C_1 = g^{\tilde{x}} h^{\tilde{r}} (\bmod n)$ and $C_2 = y^{\tilde{r}} (\bmod n)$.
3. Player A engages with player B to a proof of knowledge for the discrete-log relation set $\langle -1, 0, \tilde{x}, \tilde{r}, 0\rangle$ and $\langle 0, -1, 0, 0, \tilde{r}\rangle$ over the objects C_1, C_2, g, h, y. Observe that the relation set is triangular.
4. Player B selects $\tilde{y} \in_R \mathbb{Z}_{\tilde{m}}$ and transmits $\tilde{y}$ to A.
5. Player A computes $x' = \tilde{x} + \tilde{y} (\bmod \tilde{m})$ and transmits to player B the value $C_3 = a^{x'}$.
6. Player A engages with player B to a proof of knowledge for the discrete-log relation set $\langle -1, 0, \alpha, \beta, \gamma, 0, 0\rangle$, $\langle 0, -1, 0, 0, 0, 0, 0, \gamma\rangle$, $\langle 0, 0, -1, 0, 0, 0, \alpha, 0\rangle$ over the objects $C_1 g^{\tilde{y}}, C_2, C_3, g, g^{\tilde{m}}, h, a, y$ (observe again, that the relation set is triangular).
7. Player A engages with player B to a tight interval proof for C_3 ensuring that $\log_a C_3 \in \mathbb{Z}_{\tilde{m}}$ (treating $\mathbb{Z}_{\tilde{m}}$ as an integer range); this is done as described in [5].
8. Player A outputs $x := x' + 2^\ell - 2^\mu + 1$ and Player B outputs $C := C_3 a^{2^\ell - 2^\mu + 1}$.

Theorem 15 *The above protocol implementation for non-adaptive drawing of random powers is correct and secure, per definitions 13 and 14, under the Strong-RSA and DDH assumptions.*

Proof. Regarding correctness note that if both players follow the protocol, then x is selected from the uniform distribution over $S(2^\ell, 2^\mu)$; following the protocol steps both parties will obtain the output as specified in the ideal implementation.

Next, we deal with the first equation of definition 14 (essentially the security for player B). Let A_1^* be any ITM for player A in the real protocol execution. We need to construct an ITM A_0^* as an ideal world transformation of A_1^* so that the first equation is satisfied.

A_0^* operates as follows: it simulates A_1^* up to the point that A_1^* initiates the protocol dialog; in this case A_0^* transmits the go message to T and receives the value $x \in S(2^\ell, 2^\mu)$. On the other hand, if A_1^* never initiates the protocol dialog, A_0^* transmits $\perp$ to T. Subsequently, A_0^* continues the simulation of A_1^*. A_0^* stores the values C_1, C_2 as transmitted by A_1^*; then, it selects two challenges c^*, c so that

$c \in_R \{0,1\}^{k_1}$ and $c^* \in_R \{0,1\}^{k_1} - \{c\}$, and simulates A_1^* till step 3 is completed so that the challenge c^* is supplied. Then, A_0^* rewinds A_1^* to the step that A_1^* waits the challenge in the proof of knowledge of step 3 and A_0^* gives to A_1^* the challenge c. Based on the soundness property of the proof of knowledge of step 3, A_0^* is capable of reconstructing $\tilde{x}$ and $\tilde{r}$ (the witnesses). Subsequently it computes $x' = x - (2^\ell - 2^\mu + 1)$ (as an integer) and sets $\tilde{y} = x' - \tilde{x}(\bmod \tilde{m})$; then it transmits $\tilde{y}$ to A_1^*. A_1^* replies by C_3 and the proofs of knowledge of step 5 and step 6. A_0^* verifies the proofs of knowledge that they are correct and in this case, it transmits C_3 to T; in any other case A_0^* transmits $\perp$ to T. A_0^* continues the simulation of A_1^* and terminates by outputting the output of A_1^*.

Let us first consider the distributions $O^1 = \mathsf{OUT}_{A_1^*}^{\langle A_1^*, B_1 \rangle}(in_A)$ and $O^0 = \mathsf{OUT}_{A_0^*}^{\langle A_0^*, B_0^H \rangle}(in_A)$. It is clear that O^1 and O^0 are indistinguishable as A_0^* executes a perfect simulation of A_1^*.

Now we consider the distributions $O^1 = \mathsf{OUT}_{B_1}^{\langle A_1^*, B_1 \rangle}(in_B)$ and $O^0 = \mathsf{OUT}_{B_0^H}^{\langle A_0^*, B_0^H \rangle}(in_B)$.

The probability distributions O^1 and O^0 can be thought of, as mappings from a sequence of coin tosses to an element of $QR(n) \cup \{\perp\}$. Coins^j for $j = 1, 0$ is the set of all possible coin tosses respectively. If $b \in \mathsf{Coins}^j$, it holds that $O^j(b) \in QR(n) \cup \{\perp\}$. Now let k_1 be the number of coin tosses required for selecting the challenge of B_1 in the step 3 of the execution of the protocol in the real world. Let k_2 be the number of coin tosses that the adversary A_1^* requires to complete the final step in the proof proof of step 3 in the real world execution.

Let $b \in \mathsf{Coins}^1$, and $O^1(b) = C \in QR(n)$. Now observe that b can be mapped to a set of $(2^{k_1} - 1)2^{k_2}$ coin tosses $b' \in \mathsf{Coins}^0$ in a straightforward manner; the only point that is interesting is the fact that the coin tosses of T inside b' must be computed based on the coin tosses of B_1 for selecting $\tilde{y}$ and the base g logarithm of the value C_1. Now observe that the coin tosses b that lead B_1 to accept despite the fact that A_1^* does not construct the values C_1, C_2, C_3 properly (and as a result in this case b', B_0^H will reject) constitute a negligible fraction of all possible coin tosses. In any other case observe that it will hold that $O^1(b) = O^0(b') \in QR(n)$. Observe that this mapping can be reversed; indeed, given any $b' \in \mathsf{Coins}^0$ with $O^1(b) = C$ we can construct a string of coin tosses $b \in \mathsf{Coins}^1$ so that $O^1(b) = C$. In this case, the coin tosses from b' that correspond to T will determine the coin tosses of b required to set the coin tosses of $\tilde{y}$ using, again, the base g logarithm of the value C_1. Regarding the case of $\perp$, observe that if b is a sequence of coin tosses for which $O^1(b) = \perp$ then definitely $O^0(b') = \perp$, where $b' \in \mathsf{Coins}^0$ is any of the corresponding coin tosses to b. On other hand if b' is such that $O^1(b') = \perp$ and it holds that $O^1(b) \neq \perp$ for the corresponding coin tosses $b \in \mathsf{Coins}^1$ then this means that A_1^* cheats in some of the proofs of knowledge in steps 6, 7; this can happen with only for a negligible fraction of coin tosses. The result $O^1 \approx O^0$ follows.

Suppose now that B_1^* be an ITM playing the role of B_1 in a real protocol execution (acting as an adversary). We will design B_0^* as an adaptation of B_1^* in the ideal world. The ITM B_0^* operates as follows: in step 1 (of the real world simulation), B_0^* provides to B_1^* the values C_1, C_2 as random elements of $QR(n)$ (this is indistinguishable from real-world executions based on the DDH assumption); then in step 3, B_0^* provides to B_1^* a simulated protocol transcript for the proof of knowledge of step 3 (employing the zero-knowledge property). In step 3, B_0^* receives from B_1^* the value $\tilde{y}$ and ignores it. When B_0^* receives the value C from the trusted party T, B_0^* computes $C_3 = Ca^{-2^\ell + 2^\mu - 1}$; B_0^* gives to B_1^* the value C_3 when simulating step 4. Otherwise, if T transmitted a failure message, B_0^* will select C_3 at random from $QR(n)$ for the simulation of step 4. Subsequently, B_0^* simulates the proofs of knowledge of steps 5 and 6 and gives them to B_1^*. Finally B_0^* continues the simulation of B_1^* and returns the same output.

Next we consider the distributions, $O^1 = \mathsf{OUT}_{A_1}^{\langle A_1, B_1^* \rangle}(in_A)$ and $O^0 = \mathsf{OUT}_{A_0^H}^{\langle A_0^H, B_0^* \rangle}(in_A)$. The indistinguishability of O^1 and O^0 is simple to see: it is clear that O^0 is the uniform distribution over elements a^x with $x \in_R S(2^\ell, 2^\mu)$; the same will hold true in the case of a real execution between A_1 and B_1^* (this holds true, independently of what B_1^* does, as it cannot bias the probability distribution of the output of A_1 since the reduction modulo $\tilde{y}$ will allow the random variable $\tilde{x}$ cancel any possible bias introduced by B_1^*).

Finally, we consider the distributions $O^1 = \mathsf{OUT}_{B_1^*}^{\langle A_1, B_1^* \rangle}(in_B)$ and $O^0 = \mathsf{OUT}_{B_0^*}^{\langle A_0^H, B_0^* \rangle}(in_B)$. Recall that we assume that the challenges of B_1^* should be honest (all our zero-knowledge proofs canonical). This means that the simulations performed by B_0^* while adapting B_1^* in the ideal world are indistinguishable from the ones supplied by player A_1 in real protocol executions. It follows that the protocol views of B_1^* in real executions are indistinguishable compared to the corresponding views in the simulation performed by B_0^* when B_1^* is adapted to the ideal world; thus the two probability distributions O^1 and O^0 will be indistinguishable. $\square$

7 Traceable Signatures and Identification

A traceable signature scheme is comprised of nine protocol procedures $\mathsf{Setup}, \mathsf{Join}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{Open}, \mathsf{Reveal}, \mathsf{Trace}, \mathsf{Claim}, \mathsf{Claim\_Verify}\rangle$ that are executed by the active participants of the system, which are identified by the Group Manager (GM), a set of users and other non-trusted third parties called tracers. A traceable identification scheme is defined in a similar fashion with the difference that the Sign and Verify procedures are substituted by an Identify protocol and the Claim and Claim_Verify procedures are substituted by a Claiming protocol.

Setup (executed by the GM). For a given security parameter ν, the GM produces a publicly-known string $\mathsf{pk}_{\mathcal{GM}}$ and some private string $\mathsf{sk}_{\mathcal{GM}}$ to be used for user key generation.

Join (a protocol between a new user and the GM). In the course of the protocol the GM employs the secret-key string $\mathsf{sk}_{\mathcal{GM}}$. The outcome of the protocol results in a membership certificate cert_i that becomes known to the new user. The whole Join protocol transcript is stored by the GM in a database that will be denoted by transcripts. This is a private database and each Join transcript contains also all the coin tosses that were used by the GM during the execution.

Identify (traceable identification) It is an interactive proof system between a prover and a verifier with the user playing the role of the prover and the verifier played by any non-trusted third party. The Identify protocol is a proof of knowledge of a membership certificate cert_i. We restrict the protocol to operate in 3 rounds, with the verifier selecting a random challenge of appropriate length in the second round.

Sign and Verify. The signing and verification algorithms are derived from the Identify protocol using the Fiat-Shamir heuristics [13] (including the message into hash).

Open (invoked by the Trustee) A p.p.t. TM which, given a signature (or an identification protocol transcript), the secret-key $\mathsf{sk}_{\mathcal{GM}}$ and access to the database of all the transcripts of the Join protocols, it outputs the identity of the signer and a proof that the opening algorithm was executed properly.

Reveal (invoked by the GM) A p.p.t. TM which, given the Join transcript for a user i, it outputs the tracing trapdoor for the user i denoted by trace_i.

Trace (invoked by designated parties, called tracers). A p.p.t. TM which, given a signature σ (or a Identify transcript) and the tracing trapdoor of a certain user, checks if σ was signed by the user.

Claiming. It is an interactive proof system between a prover and a verifier where the role of the prover is played by the user and the role of the verifier is played by the claim recipient. The Claiming protocol is a proof of knowledge that binds to a given Identify protocol transcript (or signature) and employs the membership certificate cert_i of the user. As in the case of Identify protocol we restrict Claiming to be a 3-round protocol so that in round 2 the verifier selects a random challenge of appropriate length.

Claim and Claim_Verify. It is the non-interactive version of the Claiming protocol employing the Fiat-Shamir heuristics [13].

Given the inter-relationship between traceable identification and traceable signatures for simplicity we will define correctness for the signature version of the scheme (the traceable identification will be correct provided that the corresponding signature scheme is correct – note that the correctness condition for identification deals with only the honest verifier case, thus it is safe to say that the identification scheme is correct if the corresponding signature scheme is correct).

Definition 16 (Correctness for a traceable scheme) *A traceable signature scheme with security parameter ν is* **correct** *if the following four conditions are satisfied (with overwhelming probability in ν). Let $\mathsf{Sign}_{\mathcal{U}}$ be the signing mechanism of user $\mathcal{U}$ and $\mathsf{Claim}_{\mathcal{U}}$ its corresponding claiming mechanism.*

(1) **Sign-Correctness:** *It should hold that for all M,* $\mathsf{Verify}(M, \mathsf{pk}_{\mathcal{GM}}, \mathsf{Sign}_{\mathcal{U}}(M)) = \mathsf{true}$.

(2) **Open-Correctness:** *For all M,* $\mathsf{Open}(\mathsf{sk}_{\mathcal{GM}}, \mathsf{transcripts}, \mathsf{Sign}_{\mathcal{U}}(M)) = \mathcal{U}$.

(3) **Trace-Correctness:** *For any M, it should hold that* $\mathsf{Trace}(\mathsf{Reveal}(\mathcal{U}, \mathsf{transcripts}), \mathsf{Sign}_{\mathcal{U}}(M)) = \mathsf{true}$*; on the other hand for any $\sigma \leftarrow \mathsf{Sign}_{\mathcal{U}'}(M)$ with $\mathcal{U}' \neq \mathcal{U}$ it should hold that* $\mathsf{Trace}(\mathsf{Reveal}(\mathcal{U}, \mathsf{transcripts}), \sigma) = \mathsf{false}$.

(4) **Claim-Correctness:** *The* Claim *and* Claim_Verify *satisfy the following property: for all $M, \sigma \leftarrow \mathsf{Sign}_{\mathcal{U}}(M)$ it holds that* $\mathsf{Claim\_Verify}(M, \sigma, \mathsf{Claim}_{\mathcal{U}}(M, \sigma)) = \mathsf{true}$.

7.1 Security Model for Traceable Schemes

In this section we formalize the security model for traceable signature schemes. To claim security we will define the notion of an interface $\mathcal{I}$ for a traceable scheme which is a PTM that simulates the operation of the system. The purpose behind the definition of $\mathcal{I}$ is to capture all possible adversarial activities against a traceable scheme in an intuitive way. We will deal with the security of the interactive version of a traceable scheme, i.e., a traceable identification scheme. We model our security using canonical 3-move proofs of knowledge and passive impersonation-type of attacks; identification security in this type of model facilitates the employment of the Fiat-Shamir transform for proving signature security; thus, proving security for the interactive version will be sufficient for ensuring security of the traceable signature in the random oracle model (see [1]).

We model the security of a traceable identification scheme as an interaction between the adversary $\mathcal{A}$ and an entity called the *interface*. The interface maintains a (private) state denoted by $\mathsf{state}_{\mathcal{I}}$ (or simply state) and communicates with the adversary over a handful of pre-specified *query actions* that allow the adversary to learn information about $\mathsf{state}_{\mathcal{I}}$; these queries are specified below. The initial state of the interface is set to $\mathsf{state}_{\mathcal{I}} = \langle \mathsf{sk}_{\mathcal{GM}}, \mathsf{pk}_{\mathcal{GM}} \rangle$. The interface also employs an "internal user counter" denoted by n which is initialized to 0. Moreover three sets are initialized U^p, U^a, U^b to $\emptyset$. Note that $\mathsf{state}_{\mathcal{I}}$ is also assumed to contain U^p, U^a, U^b and n. Finally the interface employs two other strings denoted and initialized as follows: $\mathsf{transcripts} = \epsilon$ and $\mathsf{Sigs} = \epsilon$. The various query action specifications are listed below:

- $\langle \mathcal{Q}_{\mathsf{pub}} \rangle$. The interface returns the string $\langle \mathsf{n}, \mathsf{pk}_{\mathcal{GM}} \rangle$. This allows to an adversary to learn the public-information of the system, i.e., the number of users and the public-key information.
- $\langle \mathcal{Q}_{\mathsf{key}} \rangle$. The interface returns $\mathsf{sk}_{\mathcal{GM}}$; this query action allows to the adversary to corrupt the group-manager.

- $\langle \mathcal{Q}_{\mathsf{p-join}} \rangle$. The interface simulates the Join protocol in *private*, increases the user count n by 1, and sets $\mathsf{state} := \mathsf{state}_{\mathcal{I}} || \langle \mathsf{n}, \mathsf{transcript}_{\mathsf{n}}, \mathsf{cert}_{\mathsf{n}} \rangle$. It also adds n into U^p and sets $\mathsf{transcripts} := \mathsf{transcripts} || \langle \mathsf{n}, \mathsf{transcript}_{\mathsf{n}} \rangle$.

  This query action allows to the adversary to introduce a new user to the system (that is not adversarially controlled).

- $\langle \mathcal{Q}_{\mathsf{a-join}} \rangle$. The interface initiates an active Join dialog with the adversary; the interface increases the user count n by 1, and assumes the role of the GM where the adversary is assuming the role of the prospective user. If the dialog terminates successfully the interface sets $\mathsf{state}_{\mathcal{I}} := \mathsf{state}_{\mathcal{I}} || \langle \mathsf{n}, \mathsf{transcript}_{\mathsf{n}}, \perp \rangle$. It finally adds n into the set U^a and $\mathsf{transcripts} := \mathsf{transcripts} || \langle \mathsf{n}, \mathsf{transcript}_{\mathsf{n}} \rangle$.

  This query action allows to the adversary it introduce an adversarially controlled user to the system. The adversary has the chance to interact with the GM during the Join transcript.

- $\langle \mathcal{Q}_{\mathsf{t-join}} \rangle$. This query is identical to a $\mathcal{Q}_{\mathsf{p-join}}$ query with the difference that the interface at the end transmits $\mathsf{cert}_{\mathsf{n}}$ to the adversary and adds n to U^a. The query $\mathcal{Q}_{\mathsf{t-join}}$ is weaker than the query $\mathcal{Q}_{\mathsf{a-join}}$ and we include it in the modelling for technical reasons. See lemma 21.

- $\langle \mathcal{Q}_{\mathsf{b-join}} \rangle$. The interface initiates an active Join dialog with the adversary; the interface increases the user count n by 1 and assumes the role of the prospective user and the adversary is assuming the role of the GM. If the dialog terminates successfully the interface sets $\mathsf{state}_{\mathcal{I}} := \mathsf{state}_{\mathcal{I}} || \langle \mathsf{n}, \perp, \mathsf{cert}_{\mathsf{n}} \rangle$. It also adds n into U^b.

  This query allows the adversary to introduce users to the system acting as a GM.

- $\langle \mathcal{Q}_{\mathsf{id}}, i \rangle$. The interface parses $\mathsf{state}_{\mathcal{I}}$ and if it discovers an entry of the form $\langle i, \cdot, \mathsf{cert}_i \rangle$ it produces an $\mathsf{Identify}$ protocol transcript using the certificate cert_i and selecting the verifier challenge at random; if no such entry is discovered or if $i \in U^a$ the interface returns $\perp$. Finally, if σ is the protocol transcript the interface sets $\mathsf{Sigs} = \mathsf{Sigs} || \langle i, \sigma \rangle$.

- $\langle \mathcal{Q}_{\mathsf{reveal}}, i \rangle$. The interface returns the output of $\mathsf{Reveal}(i, \mathsf{transcripts})$. Sometimes we will write $\mathcal{Q}^{\neg A}_{\mathsf{reveal}}$ to restrict the interface from revealing users in A. Note that $\mathsf{Reveal}(i, \mathsf{transcripts}) = \perp$ in case user i does not exist or $i \in U^b$.

Given the above definition of an interface we proceed to characterize the various security properties that a traceable scheme should satisfy. We will use the notation $\mathcal{I}[a, \mathcal{Q}_1, \ldots, \mathcal{Q}_r]$ to denote the operation of the interface with (initial) state a that responds to the query actions $\mathcal{Q}_1, \ldots, \mathcal{Q}_r$ (a subset of the query actions defined above). In general we assume that the interface serves one query at a time: this applies to the queries $\mathcal{Q}_{\mathsf{a-join}}$ and $\mathcal{Q}_{\mathsf{b-join}}$ that require interaction with the adversary (i.e., the interface does not allow the adversary to cascade such queries). For some traceable identification scheme we will denote by iP and iV the prover and verifier algorithms for the $\mathsf{Identify}$ 3-move protocol as well as by cP and cV the prover and verifier algorithms of the $\mathsf{Claiming}$ 3-move protocol.

Our definition of security, stated below, is based on the definitions of the three named security properties in the coming subsections.

Definition 17 *A traceable scheme is said to be* **secure** *provided that it satisfies security against misidentification, anonymity and framing attacks.*

7.1.1 Misidentification Attacks.

In a misidentification attack against a traceable scheme the adversary is allowed to control a number of users of the system (in an adaptive fashion). The adversary is also allowed to observe the operation

$$
\mathsf{Exp}^{\mathcal{A}}_{\mathsf{mis}}(\nu): \left|\begin{array}{l}
\mathsf{state}_{\mathcal{I}} = \langle \mathsf{pk}_{\mathcal{GM}}, \mathsf{sk}_{\mathcal{GM}}\rangle \leftarrow \mathsf{Setup}(1^{\nu});\\
\langle d, \rho_1\rangle \leftarrow \mathcal{A}^{\mathcal{I}[\mathsf{state}_{\mathcal{I}}, \mathcal{Q}_{\mathsf{pub}}, \mathcal{Q}_{\mathsf{p-join}}, \mathcal{Q}_{\mathsf{a-join}}, \mathcal{Q}_{\mathsf{id}}, \mathcal{Q}_{\mathsf{reveal}}]}(\mathsf{first}, 1^{\nu});\\
c \xleftarrow{r} \{0,1\}^k;\\
\rho_2 \leftarrow \mathcal{A}(\mathsf{second}, d, \rho_1, c);\\
\mathtt{if}\ \mathsf{iV}(\mathsf{pk}_{\mathcal{GM}}, \rho_1, c, \rho_2) = \mathtt{true\ and}\\
\qquad \mathtt{if}\ \mathsf{Open}(\mathsf{sk}_{\mathcal{GM}}, \mathsf{transcripts}, \rho_1) \notin U^a\\
\qquad\quad \mathtt{or}\ \wedge_{i\in U^a} \mathsf{Trace}(\mathsf{Reveal}(i, \mathsf{transcripts}), \rho_1) = \mathtt{false}\\
\qquad \mathtt{then\ output\ 1}\\
\mathtt{else\ output\ 0}
\end{array}\right.
$$

Figure 2: The misidentification experiment

$$
\mathsf{Exp}^{\mathcal{A}}_{\mathsf{anon}}(\nu): \left|\begin{array}{l}
\mathsf{state}_{\mathcal{I}} = \langle \mathsf{pk}_{\mathcal{GM}}, \mathsf{sk}_{\mathcal{GM}}\rangle \leftarrow \mathsf{Setup}(1^{\nu});\\
\langle d, i_0, i_1\rangle \leftarrow \mathcal{A}^{\mathcal{I}[\mathsf{state}_{\mathcal{I}}, \mathcal{Q}_{\mathsf{pub}}, \mathcal{Q}_{\mathsf{p-join}}, \mathcal{Q}_{\mathsf{a-join}}, \mathcal{Q}_{\mathsf{id}}, \mathcal{Q}_{\mathsf{reveal}}]}(\mathsf{play}, 1^{\nu});\\
\mathtt{if}\ \ i_0\ \mathtt{or}\ \ i_1\ \mathtt{do\ not\ belong\ in}\ \ U^p\ \mathtt{return}\ \bot.\\
b \xleftarrow{r} \{0,1\}.\\
\mathtt{parse}\ \ \mathsf{state}_{\mathcal{I}}\ \mathtt{and\ find\ the\ entry}\ \ \langle i_b, \mathsf{transcript}_{i_b}, \mathsf{cert}_{i_b}\rangle.\\
\mathtt{simulate\ the}\ \textbf{Identify}\ \mathtt{protocol\ for}\ \ \mathsf{cert}_{i_b}\ \mathtt{to\ obtain}\ \langle \rho_1, c, \rho_2\rangle.\\
b_* \leftarrow \mathcal{A}^{\mathcal{I}[\mathsf{state}_{\mathcal{I}}, \mathcal{Q}_{\mathsf{pub}}, \mathcal{Q}_{\mathsf{p-join}}, \mathcal{Q}_{\mathsf{a-join}}, \mathcal{Q}_{\mathsf{id}}, \mathcal{Q}^{\neg(i_0,i_1)}_{\mathsf{reveal}}]}(\mathsf{guess}, 1^{\nu}, d, \langle \rho_1, c, \rho_2\rangle);\\
\mathtt{if}\ \ b = b_*\ \mathtt{then\ return}\ 1\ \mathtt{else\ return}\ 0.
\end{array}\right.
$$

Figure 3: The anonymity attack experiment

of the system in the way that users are added and they produce identification transcripts. Finally the adversary is required to produce an identification transcript that satisfies either one of the following properties: (a): the adversarial identification transcript does not open to any of the users controlled by the adversary, or (b): the adversarial identification transcript does not trace to any of the users controlled by the adversary. We will formalize this attack using the experiment presented in figure 2.

We will say that a traceable identification scheme satisfies security against misidentification if for any PPT $\mathcal{A}$, it holds that $\mathbf{Prob}[\mathsf{Exp}^{\mathcal{A}}_{\mathsf{mis}}(\nu) = 1] = \mathsf{negl}(\nu)$.

7.1.2 Anonymity Attacks

An anonymity attack is best understood in terms of the following experiment that is played with the adversary $\mathcal{A}$ who is assumed to operate in two phases called play and guess. In the play phase, the adversary interacts with the interface, introduces users in the system, and selects two target users he does not control; then receives an identification transcript that corresponds to one of the two at random; in the guess stage the adversary tries to guess which of the two produced the identification transcript. We remark that we allow the adversary to participate in the system also as a tracer (i.e., one of the clerks that assist in the tracing functionality). The experiment is presented in figure 3.

A traceability scheme is said to satisfy anonymity if for any attacker $\mathcal{A}$ it holds that
$|\mathbf{Prob}[\mathsf{Exp}^{\mathcal{A}}_{\mathsf{anon}}(\nu) = 1] - \frac{1}{2}| = \mathsf{negl}(\nu)$.

7.1.3 Framing Attacks

A user may be framed or two different ways: the authorities and other users may construct a signature that opens or trace to an innocent user, or they may claim a signature that was generated by the user as

$$
\mathsf{Exp}^{\mathcal{A}}_{\mathsf{fra}}(\nu): \left|
\begin{array}{l}
\mathsf{state}_{\mathcal{I}} = \langle \mathsf{pk}_{\mathcal{GM}}, \mathsf{sk}_{\mathcal{GM}} \rangle \leftarrow \mathsf{Setup}(1^{\nu});\\
\langle \mathsf{s}, d, \rho_1 \rangle \leftarrow \mathcal{A}^{\mathcal{I}[\mathsf{state}_{\mathcal{I}}, \mathcal{Q}_{\mathsf{pub}}, \mathcal{Q}_{\mathsf{key}}, \mathcal{Q}_{\mathsf{b-join}}, \mathcal{Q}_{\mathsf{id}}]}(\mathsf{first}, 1^{\nu});\\
c \xleftarrow{r} \{0,1\}^k;\\
\rho_2 \leftarrow \mathcal{A}(\mathsf{second}, d, \rho_1, c);\\
\texttt{if } \mathsf{iV}(\mathsf{pk}_{\mathcal{GM}}, \rho_1, c, \rho_2) = \texttt{true and}\\
\qquad \texttt{if } \mathsf{Open}(\mathsf{sk}_{\mathcal{GM}}, \mathsf{transcripts}, \rho_1) \in U^b\\
\qquad\quad \texttt{or } \exists i \in U^b : \mathsf{Trace}(\mathsf{Reveal}(\mathsf{sk}_{\mathcal{GM}}, \mathsf{transcripts}, i), \rho_1) = \texttt{true}\\
\qquad \texttt{then output 1}\\
\texttt{else if } \mathsf{s} \texttt{ is such that } \langle i, \mathsf{s} \rangle \in \mathsf{Sigs} \texttt{ and } i \in U^b\\
\qquad \texttt{and } \mathsf{cV}(\mathsf{s}, \rho_1, c, \rho_2) = \mathsf{true} \texttt{ then output 1}\\
\texttt{else output 0}
\end{array}\right.
$$

Figure 4: The framing attack experiment

their own. We capture these two framing notions with the experiment described in figure 4 (we remark that "exculpability" of group signatures [2] is integrated in this experiment).

A traceable scheme satisfies security against framing provided that for any probabilistic polynomial-time $\mathcal{A}$ it holds that $\mathbf{Prob}[\mathsf{Exp}^{\mathcal{A}}_{\mathsf{fra}}(\nu) = 1] = \mathsf{negl}(\nu)$.

7.1.4 Comments

(i) In modeling misidentification and anonymity attacks we do not allow the adversary to submit "open" queries to the interface. This models the fact that opening a signature is an internal operation performed by the GM. On the contrary, this is not assumed for the tracing operation, since we model it as a distributed operation whose results are made available to distributed agents (and thus a Reveal oracle query is available to the adversary). Allowing opening oracles to be part of the adversarial control is possible, but will require our encryptions and commitments to be of the chosen ciphertext secure type.
(ii) Framing and misidentification were identified as separate attacks in our model. We could have collapsed them into a single attack that we can express formally. However, we feel that the goals of the two attacks are different and they represent two different ways by which the adversary interacts with the system. As we will show later, the security proofs against them rely on different intractability assumptions, which constitutes supporting evidence for our intuition.

8 Design of a Traceable Scheme

8.1 The Construction

Parameters. The parameters of the scheme are $\epsilon \in \mathbb{R}$ with $\epsilon > 1$, $k \in \mathbb{N}$ as well as three spheres $\Lambda, \mathrm{M}, \Gamma$ satisfying the properties presented in 5; the function ϵ is supposed to satisfy the condition of lemma 9. Below we will denote by Λ^k_ϵ, and Γ^k_ϵ the inner spheres of Λ, M and Γ w.r.t. the parameters ϵ, k, (see section 4.2).
Setup The GM generates two primes p', q' with $p = 2p' + 1$, $q = 2q' + 1$ also primes. The modulus is set to $n = pq$. The spheres $\Lambda, \mathrm{M}, \Gamma$ are embedded into $\{0, \ldots, p'q' - 1\}$. Also the GM selects $a, a_0, b, g, h \in_R QR(n)$ of order $p'q'$. The secret-key $\mathsf{sk}_{\mathcal{GM}}$ of the GM is set to p, q. The public-key of the system is subsequently set to $\mathsf{pk}_{\mathcal{GM}} := \langle n, a, a_0, b, y, g, h \rangle$.
Join (a protocol executed by a new user and the GM). The prospective user and the GM execute the protocol for non-adaptive drawing a random power $x' \in \Lambda^k_\epsilon$ over b (see section 6) with the user playing

the role of player A and the GM playing the role of player B; upon successful completion of the protocol the user obtains x'_i and the GM obtains the value $C_i = b^{x'_i}$.

Subsequently the GM selects a random prime $e_i \in \Gamma_\epsilon^k$ and $x_i \in \Lambda_\epsilon^k$ and then computes $A_i = (C_i a^{x_i} a_0)^{e_i^{-1}} (\mathrm{mod} n)$ and sends to the user the values $\langle A_i, e_i, x_i \rangle$. The user forms the membership certificate as $\mathsf{cert}_i := \langle A_i, e_i, x_i, x'_i \rangle$. Observe that $\langle A_i, e_i : x_i, x'_i \rangle$ is a discrete-log representation of an arbitrary power in $QR(n)$ (see section 5); furthermore observe that the portion of the certificate x_i is known to the GM and will be used as the user's tracing trapdoor.

Identify. To identify herself a user first computes the values,

$$T_1 = A_i y^r,\ T_2 = g^r,\ T_3 = g^{e_i} h^r,\ T_4 = g^{x_i k},\ T_5 = g^k,\ T_6 = g^{x'_i k'},\ T_7 = g^{k'}$$

where $r, k, k' \in_R$ M. Subsequently the user proceeds to execute the proof of knowledge of the following triangular discrete-log relation set defined over the objects $g, h, y, a_0, a, b, T_1^{-1}, T_2^{-1}, T_3, T_4, T_5, T_6, T_7$ and the free variables are $x, x' \in \Lambda_\epsilon^k, e \in \Gamma_\epsilon^k, r, h'$.

$$\left[\begin{array}{rccccccccccccc}
 & g & h & (T_2)^{-1} & T_5 & T_7 & y & (T_1)^{-1} & a & b & a_0 & T_3 & T_4 & T_6 \\
T_2 = g^r : & r & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
T_3 = g^e h^r : & e & r & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\
T_2^e = g^{h'} : & h' & 0 & e & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
T_5^x = T_4 : & 0 & 0 & 0 & x & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\
T_7^{x'} = T_6 : & 0 & 0 & 0 & 0 & x' & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\
a_0 a^x b^{x'} y^{h'} = T_1^e : & 0 & 0 & 0 & 0 & 0 & h' & e & x & x' & 1 & 0 & 0 & 0
\end{array}\right]$$

Observe that the above proof of knowledge ensures that the values $T_1, T_2, T_3, T_4, T_5, T_6, T_7$ are properly formed and "contain" a valid certificate. In particular the above proof not only enforces the certificate condition $A_i^{e_i} = a_0 a^{x_i} b^{x'_i}$ but also the fact that $e_i \in \Gamma$ and $x_i, x'_i \in \Lambda$.

Open. (invoked by the GM) Given a Identify transcript $\langle \rho_1, c, \rho_2 \rangle$ and all Join transcripts the GM does the following: it parses ρ_1 for the sequence $\langle T_1, T_2, T_3, T_4, T_5, T_6, T_7 \rangle$ and computes the value $A = (T_2)^{-x} T_1$. Then it searches the membership certificates $\langle A_i, e_i \rangle$ (available from the Join transcripts) to discover the index i such that $A = A_i$; the index i identifies the signer of the message.

Reveal. (invoked by the GM) Given the Join transcript of the i-th user the GM parses the Join transcript to recover the tracing trapdoor $\mathsf{trace}_i := x_i$.

Trace. (invoked by any agent/clerk) Given the value trace_i and an Identify protocol transcript $\langle \rho_1, c, \rho_2 \rangle$ the agent parses the sequence $\langle T_1, T_2, T_3, T_4, T_5, T_6, T_7 \rangle$ from ρ_1; subsequently it checks whether $T_5^{x_i} = T_4$; if this is the case the agent concludes that user i is the originator of the given Identify protocol transcript.

Claiming. (invoked by the user) Given an Identify protocol transcript that was generated by user i and contains the sequence $\langle T_1, T_2, T_3, T_4, T_5, T_6, T_7 \rangle$, the user i can claim that he is the originator as follows: he initiates a proof of knowledge of the discrete-log of T_6 base T_7 (which is a discrete-log relation set, see section 4). If the proof is directed to a specific entity the proof can be targeted to the receiver using a designated verifier proof, see [16]; such proofs can be easily coupled to our proofs of knowledge for discrete-log relation sets.

9 Security and Correctness of the Protocol

In this section we prove that our construction is correct and secure according to definitions 16 and 17.

Theorem 18 *The traceable scheme of section 8.1 is correct.*

Proof. Sign-correctness follows immediately from the completeness of the zero-knowledge proof that is employed for the signing algorithm.

Open-correctness follows from the fact that any signature $\sigma \leftarrow \mathsf{Sign}_{\mathcal{U}}(M)$ contains the values T_1, T_2 that constitute an ElGamal encryption over $QR(n)$ of the value $A_{\mathcal{U}}$ of the user's membership certificate $\langle A_{\mathcal{U}}, e_{\mathcal{U}}, x_{\mathcal{U}}, x'_{\mathcal{U}}\rangle$. With overwhelming probability this certificate is unique for each user, and thus the comparison with the Join protocol $\mathsf{transcripts}$ database will allow the recovery of the identity of the user $\mathcal{U}$.

Regarding Trace-correctness observe that $\mathsf{Reveal}(\mathcal{U}, \mathsf{transcripts}) = x_{\mathcal{U}}$ where $\langle A_{\mathcal{U}}, e_{\mathcal{U}}, x_{\mathcal{U}}, x'_{\mathcal{U}}\rangle$ is the membership certificate of user $\mathcal{U}$; now observe that any $\sigma \leftarrow \mathsf{Sign}_{\mathcal{U}}(M)$ contains the values T_4, T_5 that satisfy the property $T_4 = T_5^{x_{\mathcal{U}}}$; thus it holds that $\mathsf{Trace}(x_{\mathcal{U}}, \sigma) = \mathsf{true}$. On the other hand, for any other user $\mathcal{U}' \neq \mathcal{U}$ it holds with overwhelming probability that $x_{\mathcal{U}} \neq x_{\mathcal{U}'}$ and thus if $\sigma \leftarrow \mathsf{Sign}_{\mathcal{U}'}(M)$ it holds that the values T_4, T_5 inside σ satisfy the condition $T_4 = g^{kx_{\mathcal{U}'}}$ and $T_5 = g^k$; since $x_{\mathcal{U}} \neq x_{\mathcal{U}'}$ they cannot satisfy the condition $T_4 = T_5^{x_{\mathcal{U}}}$.

Finally, regarding Claim-correctness, observe that the $\mathsf{Claim}_{\mathcal{U}}$ algorithm produces a non-interactive proof of knowledge for the discrete-logarithm of the value T_6 base T_7 inside the signature $\sigma \leftarrow \mathsf{Sign}_{\mathcal{U}}(M)$; based on the completeness property of this proof of knowledge the correctness follows. □

Theorem 19 *The traceable scheme of section 8.1 is secure according to definition 17.*

In particular it satisfies (i) security against misidentification attacks based on the Strong-RSA and the DDH assumptions; (ii) security against anonymity attacks based on the DDH assumption; (iii) security against framing attacks based on the discrete-logarithm problem over $QR(n)$ when the factorization of n is known.

The exposition of the proof of the above theorem will be spread in the following three subsections, one for each of the security properties. We will start with some basic lemmas that will be useful in the main security proofs.

Lemma 20 *Let $\mathsf{pk}_{\mathcal{GM}} = \langle n, g, a_0, a, b, y, g, h\rangle$ be the public-key in the scheme of section 8.1. There exists a PPT $\mathcal{S}_{\mathsf{id}}$ that takes as input $\mathsf{pk}_{\mathcal{GM}}$ and a tuple $\langle A, e, x, x'\rangle \in QR(n) \times \Gamma \times \Lambda \times \Lambda$ (not necessarily satisfying the condition $A^e = a_0 a^x b^{x'}$) that is capable of simulating the valid identification transcripts generated by a single user i with membership certificate $\langle A_i, e_i : x_i, x'_i\rangle$, for which it holds $e_i = e, x_i = x, x'_i = x$ (but potentially $A_i \neq A$).*

In particular the distance between $\mathcal{S}_{\mathsf{id}}$ and real **Identify** *protocol transcripts of the user i is at most $2\mathsf{Adv}^{DDH}(\nu) + \epsilon$ where ϵ is the statistical distance of the simulator $\mathcal{S}$ of the 3-move zero-knowledge proof of knowledge used inside the* **Identify** *protocol.*

Proof. $\mathcal{S}_{\mathsf{id}}$ operates as follows: first it sets,

$$T_1 = Ay^r,\ T_2 = g^r,\ T_3 = g^e h^r,\ T_4 = g^{xk},\ T_5 = g^k\ T_6 = g^{x'k'}\ T_7 = g^{k'}$$

for $r, k, k' \in_R \mathrm{M}$. Then, based on properties of the proof of knowledge we know that there exists a simulator $\mathcal{S}$ for the proof of knowledge of the discrete-log relation-set that corresponds to the **Identify** 3-move proof of knowledge. Thus, $\mathcal{S}_{\mathsf{id}}$ simulates $\mathcal{S}$ over the objects $g, h, y, a_0, a, b, T_1, T_2, T_3, T_4, T_5, T_6, T_7$.

Also let $\mathcal{S}_R$ be a simulator operating as $\mathcal{S}_{\mathsf{id}}$ but using $T'_1 \in_R QR(n)$ instead of T_1.

Suppose now that there exists a distinguisher of between valid protocol transcripts and the output of $\mathcal{S}_{\mathsf{id}}$.

Let $\langle G, X, Y, Z\rangle$ be a challenge for the DDH assumption. Consider the following algorithm $\mathcal{S}^*$. It sets $T_1' := AZ, T_2' := X, g' = G, y' := Y$ and it simulates $\mathcal{S}$ on the input $n, g', a_0, a, b, y', g, h, A, e, x, x'$. If G, X, Y, Z is valid DDH tuple then observe that the output of $\mathcal{S}^*$ is identically distributed to the output of $\mathcal{S}_{\mathsf{id}}$. On the other hand if G, X, Y, Z is a random tuple then the output of $\mathcal{S}^*$ is identically distributed to the output of $\mathcal{S}_R$.

It follows that the probability distributions of $\mathcal{S}_{\mathsf{id}}$ and $\mathcal{S}_R$ have statistical distance at most $\mathsf{Adv}^{DDH}(\nu)$.

Now consider the distribution $\mathcal{D}$ of all valid protocol transcripts generated by a single user i. We modify the **Identify** protocol of the user i to use the simulator $\mathcal{S}$ of the 3-move proof of knowledge of a discrete-log relation set instead. Based on theorem 10 it is easy to see that the modified probability distribution $\mathcal{D}'$ of all valid protocol transcripts generated by a single user i will be statistically indistinguishable from the distribution $\mathcal{D}$ (it will distance ϵ in particular). Next we modify the $\mathcal{D}'$ further so that the value T_1 is substituted by the value T_1' selected at random from $QR(n)$; the modified ditribution will be denoted by $\mathcal{D}''$. It is easy to see that the distance of $\mathcal{D}''$ from $\mathcal{D}'$ is at most $\mathsf{Adv}^{DDH}(\nu)$. Finally observe that the distribution $\mathcal{D}''$ is identical to the distribution generated by $\mathcal{S}_R$. The proof of the theorem follows easily. □

Lemma 21 *For any probabilistic polynomial-time algorithm $\mathcal{A}$ that interacts with the interface as defined in section 7.1,* $\mathbf{Prob}_{\mathsf{state}_\mathcal{I} \leftarrow \mathsf{Setup}(1^\nu)}[\mathcal{A}^{\mathcal{I}[\mathsf{state}_\mathcal{I}, \ldots, \mathcal{Q}_{\mathsf{a-join}}, \ldots]} = \mathcal{A}'^{\mathcal{I}[\mathsf{state}_\mathcal{I}, \ldots, \mathcal{Q}_{\mathsf{t-join}}, \ldots]}] = 1 - \mathsf{negl}(\nu)$.

Proof. The proof is based on the security properties of the Join protocol which is a secure implementation of a non-adaptive drawing of a random power as described in section 6. In particular due to the security from the player B's side we can simulate $\mathcal{A}$ so that the values x' that $\mathcal{A}$ obtains from each instantiation of a $\mathcal{Q}_{\mathsf{a-join}}$ protocol can be chosen externally by a trusted party. According to theorem 15 this does not affect the private output functionality. □

9.1 Security against Misidentification

Theorem 22 *The traceable scheme of section 8.1 satisfies security against misidentification based on the strong-RSA assumption and the DDH assumption over $QR(n)$.*

Proof. We remark that for simplicity we will suppress the use of inner sphere notation, and use the notation Λ and Γ instead of Λ_k^ϵ and Γ_k^ϵ (this does not affect the argumentation of the proof in any way). Let $\mathcal{A}$ be an adversary that violates security against misidentification. It follows that

$$\mathbf{Prob}[\mathsf{Exp}_{\mathsf{imp}}^{\mathcal{A}}(\nu) = 1]$$

is a non-negligible function in w. We will use $\mathcal{A}$ to construct an algorithm that solves the one-more representation problem. First, let K be the number of users that are controlled by the adversary (i.e., introduced in system using $\mathcal{Q}_{\mathsf{a-join}}$).

Now observe that based on lemma 21 there exists an adversary $\mathcal{A}'$ that has the same functionality as $\mathcal{A}$ but whenever he executes $\mathcal{Q}_{\mathsf{a-join}}$ he obtains the value x' through querying an external trusted-third party.

Let n be a composite modulus with unknown factorization according to the specifications of our protocol and $\{\langle A_j, e_j : x_j, x_j'\rangle\}_{j=1}^K$ be an instance of the one-more representation problem over the bases a_0, a, b. Below we describe an algorithm $\mathcal{B}$ that uses $\mathcal{A}'$ to solve the one-more discrete-log representation problem.

First, $\mathcal{B}$ selects y, g, h values as specified in the description of the protocol and sets $\mathsf{pk}_{\mathcal{GM}} := \langle n, a, a_0, b, y, g, h\rangle$. Subsequently $\mathcal{B}$ simulates the adversary $\mathcal{A}'(\mathsf{first}, 1^\nu)$ playing the role of an appropriately modified interface as described below:

- If $\mathcal{A}'$ submits $\langle \mathcal{Q}_{\mathsf{pub}} \rangle$ to the interface, then $\mathcal{B}$ supplies to $\mathcal{A}$ the specified response (the public-key of the system).

- If $\mathcal{A}'$ submits $\langle \mathcal{Q}_{\mathsf{p-join}} \rangle$ to the interface, then $\mathcal{B}$ increments the internal user counter i by one, and selects $\mathsf{cert}_i = A, e, x, x' \in QR(n) \times \Gamma \times \Lambda \times \Lambda$ and stores cert_i in the database by inserting the string $\langle i, \perp, \mathsf{cert}_i \rangle$. Also, $\mathcal{B}$ adds i into the set U^p.

- If $\mathcal{A}'$ initiates a $\mathcal{Q}_{\mathsf{t-join}}$ dialog then $\mathcal{B}$ increases the join dialog counter j by one and the user counter i by one; $\mathcal{A}'$ asks the pair of values x, x' that will be used as part of the certificate. $\mathcal{B}$ supplies the input values x_j, x'_j. Then $\mathcal{A}$ submits to $\mathcal{B}$ the value $a_0 a^{x_j} a^{x'_j}$ and $\mathcal{B}$ returns A_j, e_j to $\mathcal{A}$. In addition, $\mathcal{B}$ enters in the database the entry $\langle i, \perp, \mathsf{cert}_i \rangle$ where $\mathsf{cert}_i = \langle A_j, e_j, x_j, x'_j \rangle$. Finally $\mathcal{B}$ adds i into the set U^a.

- If $\mathcal{A}$ submits $\langle \mathcal{Q}_{\mathsf{id}}, i \rangle$ to the interface, then $\mathcal{B}$ looks into the database to recover the corresponding entry $\langle i, \ldots \rangle$ and the string $\mathsf{cert}_i = \langle A_i, e_i, x_i, x'_i \rangle$ (which observe that it does not necessarily satisfies $A_i^{e_i} = a_0 a^{x_i} b^{x'_i}$) and then $\mathcal{B}$ simulates an **Identify** protocol transcript as described in lemma 20. Note that if $i \notin U^p$, $\mathcal{B}$ returns fail to the adversary.

- If $\mathcal{A}$ submits $\langle \mathcal{Q}_{\mathsf{reveal}}, i \rangle$ to the interface, then $\mathcal{B}$ looks into the database to recover the corresponding entry $\langle i, \ldots \rangle$ and the string $\mathsf{cert}_i = \langle A_i, e_i, x_i, x'_i \rangle$ and returns to the adversary the value x_i.

Observe now that the view that $\mathcal{A}'$ has of its interaction with $\mathcal{B}$ is indistinguishable from the interaction with the interface in the security definition.

At some point $\mathcal{A}'(\mathsf{first}, 1^\nu)$ terminates by returning the values d, ρ_1. Then, $\mathcal{B}$ selects two different c, c' and simulates $\mathcal{A}'(\mathsf{second}, d, \rho_1, c)$ and $\mathcal{A}'(\mathsf{second}, d, \rho_1, c')$ to obtain two outputs ρ_2, ρ'_2.

Observe now that with probability $(\mathbf{Prob}[\mathsf{Exp}^{\mathcal{A}'}_{\mathsf{mis}}(\nu) = 1])^2$ it holds that the identification protocol transcripts $\langle \rho_1, c, \rho_2 \rangle$ and $\langle \rho_1, c', \rho'_2 \rangle$ satisfy the verification function iV. Observe that $\rho_1 = \langle T_1, \ldots, T_7, \ldots \rangle$. Now, using the fact that the **Identify** protocol transcript is sound we can extract a witness x, x', w, e, h' from the two transcripts for which it will hold that $T_2 = g^w, T_3 = g^e h^w, T_2^e = g^{h'}, a_0 a^x b^{x'} y^{h'} = T_1^e, T_5^x = T_4, T_7^{x'} = T_6$.

Now we have two alternative events: (i) $\mathsf{Open}(\mathsf{sk}_{\mathcal{T}}, \mathsf{transcripts}, \rho_1) \notin U^a$ which means that $T_1/T_2^{\log_g y}$ does not equal any A_i for those $i \in U^a$; observe that $A := T_1/T_2^{\log_g y}$ has the property that $A^e = (T_1/T_2^{\log_g y})^e = a_0 a^x b^{x'} y^{h'}/y^{h'} = a_0 a^x b^{x'}$, as a result we constructed a discrete-log representation of an arbitrary power $\langle A, e : x, x' \rangle$ that is different from the ones that were selected by $\mathcal{B}$.

In the second alternative event we have : (ii) $\wedge_{i \in U^a} \mathsf{Trace}(\mathsf{Reveal}(\mathsf{sk}_{\mathcal{T}}, \mathsf{transcripts}, i), \rho_1) = \mathtt{false}$. It follows that (due to lemma 21) $\wedge_{i \in U^a} \mathsf{Trace}(x_i, \rho_1) = \mathtt{false}$ or equivalently that $T_5^{x_i} \neq T_4$ for all $i \in U^a$ and as a result $x \neq x_i$ for all $i \in U^a$.

It turns out that in both of the above cases the algorithm $\mathcal{B}$ is an algorithm that can solve the "one-more representation" problem, something that based on theorem 12, yields an algorithm against the Strong-RSA problem. □

9.2 Anonymity

Theorem 23 *The traceable scheme of section 8.1 satisfies security against anonymity attacks based on the DDH assumption over $QR(n)$.*

In particular we show that $|\mathbf{Prob}[\mathsf{Exp}^{\mathcal{A}}_{\mathsf{anon}}(\nu) = 1] - \frac{1}{2}| \leq n^2(2\mathsf{Adv}^{DDH}(\nu) + 3\mathsf{Adv}^{DDH}_{*,\Lambda,*}(\nu))$ *where* $\mathsf{Adv}^{DDH}_{*,\Lambda,*}(\nu)$ *denotes the maximum advantage of any DDH adversary when the second argument of the DDH challenge is restricted into the sphere* Λ*; this can be further relaxed, see the comment at the end of section 3.*

Proof. We remark that for simplicity we will suppress the use of inner sphere notation, and use the notation Λ and Γ instead of Λ^{ϵ}_k and Γ^{ϵ}_k (this does not affect the argumentation of the proof in any way).

Let $\mathcal{A}$ be an anonymity adversary as described in section 7.1.2 with the modification that he wants to violate the anonymity of users i_0, i_1 always for fixed i_0, i_1. We will show that for such adversary it holds that $|\mathbf{Prob}[\mathsf{Exp}^{\mathcal{A},i_0,i_1}_{\mathsf{anon}}(\nu) = 1] - \frac{1}{2}| = \mathsf{negl}(\nu)$ (assuming that the advantage of DDH distinguishers is a negligible function in w). Then, it will follow that even if $\mathcal{A}$ selects i_0, i_1 adaptively (as stated in the anonymity definition in 7.1.2) it will hold that the probability of success of the experiment will remain negligible since we assume a polynomial number of users. This is so since it is possible to transform any adaptive adversary $\mathcal{A}$ to a non-adaptive one $\mathcal{A}'$ as follows: $\mathcal{A}'$ for fixed i_0, i_1 simulates $\mathcal{A}$ and if $\mathcal{A}$ returns indeed i_0, i_1 as the challenge then $\mathcal{A}'$ proceeds with the simulation as specified, otherwise $\mathcal{A}'$ selects a random bit and returns this instead.

Observe that based on lemma 21 there exists an adversary $\mathcal{A}'$ that has the same functionality as $\mathcal{A}$ but whenever $\mathcal{A}$ executes $\mathcal{Q}_{a-join}$, $\mathcal{A}'$ executes the oracle $\mathcal{Q}_{t-join}$ instead (i.e., $\mathcal{A}'$ obtains the value x' externally). Also let K be the number of $\mathcal{Q}_{\mathsf{t-join}}$ queries executed by $\mathcal{A}'$ (in both the play and guess stages).

Now consider the following game G_1:

Let n be a composite modulus with unknown factorization according to the specifications of our construction; G_1 selects random $x_1, \ldots, x_K, x'_1, \ldots, x'_K \in \Lambda$ and $e_1, \ldots, e_K \in \Gamma$, and set $a = z^{e_1 \cdots e_K}(\mathrm{mod} n)$, $a_0 = a^r$ and $b = a^{r'}$ where r, r' are random integers in Λ. Then $\mathcal{B}$ computes $A_i = z^{(x_i + r + r'x'_i)\frac{e_1 \cdots e_K}{e_i}}(\mathrm{mod} n)$, for all $i = 1, \ldots, K$. Observe that $A_i^{e_i} = a_0 a^{x_i} b^{x'_i}$ for all $i = 1, \ldots, K$, i.e., $\langle A_i, e_i : x_i, x'_i\rangle$ are discrete-log representations of arbitrary powers inside $QR(n)$ over a_0, a, b. Let y, g, h, g_0, N be values as specified in the description of the protocol.

G_1 proceeds to simulate the adversary $\mathcal{A}'(\mathsf{play}, 1^{\nu})$ by answering $\mathcal{A}$'s oracle queries to the interface as follows (in the description below i, j are two counters initialized to 0).

- If $\mathcal{A}'$ poses the query $\mathcal{Q}_{\mathsf{pub}}$, G_1 returns the public-key of the system as defined above.
- If $\mathcal{A}'$ submits $\langle \mathcal{Q}_{\mathsf{p-join}}\rangle$ to the interface, then G_1 increments the internal user counter i by one, and selects $\mathsf{cert}_i = A, e, x, x' \in QR(n) \times \Gamma \times \Lambda \times \Lambda$ and stores cert_i in the database by inserting the string $\langle i, \perp, \mathsf{cert}_i\rangle$. Also, G_1 adds i into the set U^p.
- If $\mathcal{A}'$ initiates a $\mathcal{Q}_{\mathsf{t-join}}$ dialog then G_1 increases the join dialog counter j by one and the user counter i by one; $\mathcal{A}'$ asks the pair of values x, x' that will be used as part of the certificate. G_1 supplies the pre-computed values x_j, x'_j. Then $\mathcal{A}$ submits to G_1 the value $a_0 a^{x_j} a^{x'_j}$ and G_1 returns A_j, e_j to $\mathcal{A}$. In addition, G_1 enters in the database the entry $\langle i, \perp, \mathsf{cert}_i\rangle$ where $\mathsf{cert}_i = \langle A_j, e_j, x_j, x'_j\rangle$. Finally G_1 adds i into the set U^a.
- If $\mathcal{A}'$ submits $\langle \mathcal{Q}_{\mathsf{id}}, i\rangle$ to the interface, then G_1 checks whether $i \in U^p$ and in this case it retrieves from the database the corresponding entry $\langle i, \ldots\rangle$ and the string $\mathsf{cert}_i = \langle A_i, e_i, x_i, x'_i\rangle$ (which observe that it does not necessarily satisfies $A_i^{e_i} = a_0 a^{x_i} b^{x'_i}$) and then G_1 simulates an **Identify** protocol transcript as described in lemma 20.
- If $\mathcal{A}'$ submits the query $\langle \mathcal{Q}_{\mathsf{reveal}}, i\rangle$, G_1 checks whether $i \notin \{i_0, i_1\}$ and in this case it looks into the database for the corresponding entry $\langle i, \ldots\rangle$ and the string $\mathsf{cert}_i = \langle A_i, e_i, x_i, x'_i\rangle$; finally it

returns to the adversary the value x_i. In this case user i is removed from U^p and entered into the set U^r.

When $\mathcal{A}'(\mathsf{play}, \cdot)$ terminates, G_1 receives the values $\langle d, i_0, i_1\rangle$; if $i_0, i_1 \notin U^p$, G_1 terminates and returns 0. In the other case G_1 selects $b \xleftarrow{r} \{0,1\}$, retrieves the entry $\langle i_b, \perp, \mathsf{cert}_{i_b}\rangle$ with $\mathsf{cert}_{i_b} = \langle A_{i_b}, e_{i_b} : x_i, x_{i_b}\rangle$. Then it forms the sequence of values:

$$T_1 = A_{i_b} y^w,\ T_2 = g^w,\ T_3 = g^{e_{i_b}} h^w,\ T_4 = g^{x_{i_b} k},\ T_5 = g^k\ T_6 = g^{x'_{i_b} k'}\ T_7 =^{k'}$$

and simulates the proof of knowledge for the discrete-log relation set of the **Identify** protocol. Then G_1 simulates $\mathcal{A}(\mathsf{guess}, d, \langle \rho_1, c, \rho_2\rangle)$ employing the oracle simulations as described above and obtains the output b^*. Finally G_1 returns 1 if $b = b^*$ or 0 otherwise. Observe that $\mathbf{Prob}[G_1(\cdot) = 1] = \mathbf{Prob}[\mathsf{Exp}^{\mathcal{A}}_{\mathsf{anon}}(\nu) = 1]$.

Then consider game G_2 that operates as game G with the difference that it uses the values in the simulation of the **Identify** protocol transcript,

$$T_1 = A_{i_b} R_1,\ T_2 = R_2,\ T_3 = g^{e_{i_b}} R_2^{\log_g h},\ T_4 = g^{x_{i_b} k},\ T_5 = g^k\ T_6 = g^{x'_{i_b} k'}\ T_7 = g^{k'}$$

where $R_1, R_2 \in_R QR(n)$.

It is easy to see that $|\mathbf{Prob}[G_1(\cdot) = 1] - \mathbf{Prob}[G_2(\cdot) = 1]| \leq \mathsf{Adv}^{DDH}(\nu)$.

Next consider game G_3 that operates as G_2 but with the modification:

$$T_1 = R_1\ T_2 = R_2\ T_3 = g^{e_{i_b}} R_3,\ T_4 = g^{x_{i_b} k},\ T_5 = g^k\ T_6 = g^{x'_{i_b} k'}\ T_7 = g^{k'}$$

where $R_1, R_2, R_3 \in_R QR(n)$. It is easy to verify that $|\mathbf{Prob}[G_2(\cdot) = 1] - \mathbf{Prob}[G_3(\cdot) = 1]| \leq \mathsf{Adv}^{DDH}(\nu)$.

Now consider the following game called G_{4,i_*} that modifies G_3 as follows for $i_* = \{i_0\}$ or $i_* = \{i_0, i_1\}$.

- If $\mathcal{A}'$ poses the query $\mathcal{Q}_{\mathsf{pub}}$, G_{4,i_*} returns the public-key of the system as defined above.
- If $\mathcal{A}'$ submits $\langle \mathcal{Q}_{\mathsf{p-join}}\rangle$ to the interface, then G_1 increments the internal user counter i by one; if $i \notin i_*$ then G_{4,i_*} selects $\mathsf{cert}_i = A, e, x, x' \in QR(n) \times \Gamma \times \Lambda \times \Lambda$ and stores cert_i in the database by inserting the string $\langle i, \perp, \mathsf{cert}_i\rangle$. Now if $i \in i_*$ then G_{4,i_*} selects $\mathsf{cert}^*_i = \langle A_i, e_i, R_i, x'_i\rangle \in QR(n) \times \Gamma \times QR(n) \times \Lambda$ and stores in the database of users the value $\langle i, \perp, \mathsf{cert}^*_i\rangle$.In either case, G_{4,i_*} adds i into the set U^p.
- If $\mathcal{A}'$ submits $\langle \mathcal{Q}_{\mathsf{id}}, i\rangle$ to the interface, then G_1 checks whether $i \in U^p$ and in this case if $i \notin i_*$ it operates identically to G_3. In the case $i \in i_*$ it retrieves from the database the corresponding entry $\langle i, \ldots\rangle$ and the string $\mathsf{cert}^*_i = \langle A_i, e_i, R_i, x'_i\rangle$ and then simulates an **Identify** transcript for the values

$$T_1 = A_i y^w,\ T_2 = g^w,\ T_3 = g^{e_i} h^w\ T_4 = R_i^k,\ T_5 = g^k,\ \ T_6 = g^{x'_i k'}\ T_7 = g^{k'}$$

Finally, at the challenge phase, if $i_b \in i_*$, G_{4,i_*} constructs the **Identify** challenge as:

$$T_1 = R_1\ T_2 = R_2\ T_3 = R_3,\ T_4 = R_4,\ T_5 = R_5,\ \ T_6 = g^{x'_{i_b} k'}\ T_7 = g^{k'}$$

(in the other case G_{4,i_*} operates as game G_3). Now consider the behavior of the games G_3 and $G_{4,\{i_0\}}$. It is clear that they are either identical, or in the case $b = 0$ it holds that the distance of

$\mathbf{Prob}[G_3(\cdot) = 1]$ and $\mathbf{Prob}[G_{4,1}(\cdot) = 1]$ can be at most $\mathsf{Adv}^{DDH}_{*,\Lambda,*}(\nu)$, where $\mathsf{Adv}^{DDH}_{*,\Lambda,*}$ denotes the advantage of any PPT adversary so that the DDH's second argument is restricted over the sphere Λ. In a similar fashion, the same will hold true for the games $G_{4,\{i_0\}}$ and $G_{4,\{i_0,i_1\}}$, i.e., they will have a distance of $\mathsf{Adv}^{DDH}_{*,\Lambda,*}(\nu)$. Finally observe that in the case of $G_{4,\{i_0,i_1\}}$ the challenge will have the form:

$$T_1 = R_1 \;\; T_2 = R_2 \;\; T_3 = R_3, \;\; T_4 = R_4, \;\; T_5 = R_5, \;\; T_6 = g^{x'_{i_b}k'}, \;\; T_7 = g^{k'}$$

with R_1, R_2, R_3, R_4, R_5 random elements of $QR(n)$.

Next we define a sequence of games G_{5,i'_*} in the same fashion as G_{4,i_*}. We observe that in the case of game $G_{5,\{i_0,i_1\}}$ it holds that the challenge is as follows:

$$T_1 = R_1 \;\; T_2 = R_2 \;\; T_3 = R_3, \;\; T_4 = R_4, \;\; T_5 = R_5, \;\; T_6 = R_6, \;\; T_7 = R_7$$

with $R_1, R_2, R_3, R_4, R_5, R_6, R_7$ random elements of $QR(n)$. It is easy to see that the distance of $\mathbf{Prob}[G_{4,\{i_0,i_1\}}(\cdot) = 1]$ and $\mathbf{Prob}[G_{5,\{i_0,i_1\}} = 1]$ is at most $\mathsf{Adv}^{DDH}_{*,\Lambda,*}(\nu)$.

Moreover it is clear that game $G_{5,\{i_0,i_1\}}$ does not retain any information about i_b and as a result it is implied that $\mathbf{Prob}[G_{5,\{i_0,i_1\}}(\cdot) = 1] = 1/2$.

It is easy to see from the above that $|\mathbf{Prob}[\mathsf{Exp}^{\mathcal{A},i_0,i_1}_{\mathsf{anon}}(\nu) = 1] - \frac{1}{2}| \leq 2\mathsf{Adv}^{DDH}(\nu) + 3\mathsf{Adv}^{DDH}_{*,\Lambda,*}(\nu)$, which implies that $|\mathbf{Prob}[\mathsf{Exp}^{\mathcal{A}}_{\mathsf{anon}}(\nu) = 1] - \frac{1}{2}| \leq n^2(2\mathsf{Adv}^{DDH}(\nu) + 3\mathsf{Adv}^{DDH}_{*,\Lambda,*}(\nu))$. □

9.3 Security Against Framing

Theorem 24 *The traceable scheme of section 8.1 satisfies security against framing based the discrete-logarithm assumption over $QR(n)$ with known factorization for n.*

Proof. Let $\mathcal{A}$ be an adversary against framing as described in section 7.1.3. Let $n = pq$ with known factors p, q and a challenge $b, C \in QR(n)$ for which we want to compute the $\log_b B$. We define the following algorithm $\mathcal{B}$ that employs the adversary $\mathcal{A}$. $\mathcal{B}$ generates all the elements of the public-key of the system g, h, y, a_0, a as specified in the protocol for the RSA modulus n with the addition of challenge b as the public-key. $\mathcal{B}$ selects a random member $j \in \{1, \ldots, s\}$ where s is the total number $\mathcal{Q}_{\mathsf{b-join}}$ queries submitted by the adversary (required to be ≥ 1 in a framing attack). $\mathcal{B}$ simulates the adversary $\mathcal{A}$, by answering its interactions with the interface correctly, with the exception of $\mathcal{Q}_{\mathsf{b-join}}$ query for the j-th user that must be handled so that the adversary (playing the role of the GM) should give to the user a certificate A, e so that $A^e = a_0 a^x C$. This requires that $\mathcal{B}$ plugs C during the execution of the Join protocol; this is possible by simulating all the zero-knowledge proofs in the non-adaptive drawing of random powers executed within the Join protocol; see theorem 15. Then if the adversary outputs an identification transcript that either opens to user j traces to the user j it is clear that we can rewind the adversary and obtain a witness for that transcript that will reveal the logarithm of C base b, and thus solving the discrete-logarithm problem. The same is true for the case that the adversary outputs a claim for an identification transcript of user j : $\mathcal{B}$ rewinds $\mathcal{A}$ and obtains the witness for the claiming which, again, is the discrete-logarithm of C base b. □

10 Applications

In this section we demonstrate the potential of traceable signatures and identification in providing conditional anonymity in anonymous systems. The main motivation for our construction is the development of a generic way to transform any system $\mathcal{S}$ that provides anonymity into a system that provides

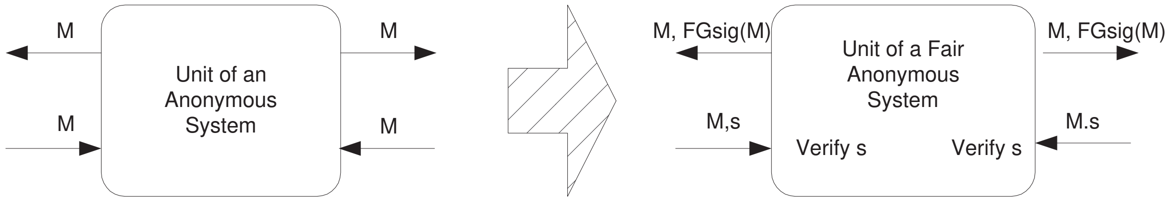

Figure 5: Adding Fairness to any Anonymous System using a traceable signature FGsig

"fair" or conditional anonymity. An anonymity system is comprised of a population of units which, depending on the system's function, exchange messages using anonymous channels. An anonymity system with *fairness* allows the identification of the origin of messages, as well as the tracing of all messages of a suspect unit, if this is mandated by the authorities. Our transformation, illustrated in figure 5, suggests that all systems' units form a group and execute the Join protocol of our traceable signature scheme prior to the initialization of the system's operation. Subsequently, any message sent from a unit is signed using the signing algorithm of our scheme; likewise for any message received, a unit verifies the signature and if it fails the message is rejected. This simple construction is powerful enough to transform an anonymous system based on a population of units to an anonymous system with fairness (conditional anonymity).

To understand the potential of this construction consider the notion of a mix-network: a mix-network is an anonymous message delivery system that allows to a set of users $U_1, \ldots, U_m$ to transmit messages that are delivered to a destination so that the correspondence of each message and sender is lost. This is achieved by employing a series of servers, called a mix-network, that shuffles the messages transmitted by the users. Only the coalition of all servers comprising the mix-network can violate the privacy of this system. Anonymous message delivery will be ensured provided that at least one server will be honest (i.e., refuse to collaborate with malicious servers against the privacy of the users).

Applying our methodology as above, only properly signed messages will be allowed to enter the mix-network. After the mixing procedure terminates the anonymity properties of our traceable signature scheme guarantee that the correspondence between senders and messages is lost. Nevertheless based on our traceability properties, the authorities will be capable of performing the operations:

- Reveal the originator of a specific message (opening).
- Reveal all messages sent by the same user (tracing).

Finally through our claiming protocol a user may claim a message as his own, at the convenience of the user (privacy is a good that should be personally managed).

10.1 Application to Auctions

Mix-nets with conditional privacy have many applications. For example, one can use them to implement an anonymous auction protocol (with open bids). Users submit their bids through the mix-network. In the message delivery point the bids are sorted from the highest to lowest and the identity of the highest bidder is revealed by performing the "open" operation of the traceable signature. Moreover a user can claim that a certain public bid as his own, if he is asked to; for example an employee of a company can prove that he submitted a bid by performing the claiming protocol. On the other hand if a certain user is found to be misbehaving (e.g., he won an auction and he refused to pay) then all his current bids must be identified and invalidated: this is possible by employing the tracing functionality of a traceable scheme.

References

[1] Michel Abdalla, Jee Hea An, Mihir Bellare, and Chanathip Namprempre. From identification to signatures via the fiat-shamir transform: Minimizing assumptions for security and forward-security. In *Advances in Cryptology – EUROCRYPT ' 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 418–433, 2002.

[2] G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik. A practical and provably secure coalition-resistant group signature scheme. In *Crypto 2000, Lecture Notes in Computer Science*, volume 1880, 2000.

[3] G. Ateniese, G. Song, and G. Tsudik. Quasi-efficient revocation of group signatures. In *Financial Crypto 2002, Lecture Notes in Computer Science*, 2002. Available at: http://paris.cs.berkeley.edu/ dawnsong/papers/grpsig-rev.ps.

[4] G. Ateniese and G. Tsudik. Some open issues and new directions in group signatures. In Matthew Franklin, editor, *Financial cryptography: Third International Conference, FC '99, Anguilla, British West Indies, February 22–25, 1999: proceedings*, volume 1648 of *Lecture Notes in Computer Science*, pages 196–211. Springer-Verlag, 1999.

[5] Fabrice Boudot. Efficient proofs that a committed number lies in an interval. In Bart Preneel, editor, *Advances in Cryptology – EUROCRYPT ' 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 431–444, Brugge, Belgium, 2000. Springer-Verlag, Berlin Germany.

[6] J. Camenisch. Efficient and generalized group signatures. In *Proc. International Advances in Cryptology Conference – EUROCRYPT '97*, pages 465–479, 1997.

[7] J. Camenisch and M. Michels. A group signature scheme with improved efficiency. *Lecture Notes in Computer Science*, 1514:160–174, 1998.

[8] J. Camenisch and M. Stadler. Efficient group signature schemes for large groups. *Lecture Notes in Computer Science*, 1294:410–424, 1997.

[9] Jan Camenisch and Anna Lysyanskaya. Dynamic accumulators and application to efficient revocation of anonymous credentials. In Moti Yung, editor, *Advances in Cryptology - CRYPTO 2002, 22nd Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 2002, Proceedings*, volume 2442 of *Lecture Notes in Computer Science*, pages 61–76. Springer, 2002.

[10] D. Chaum and E. van Heyst. Group signatures. In D. W. Davies, editor, *Advances in Cryptology, Proc. of Eurocrypt '91 (Lecture Notes in Computer Science 547)*, pages 257–265. Springer-Verlag, April 1991. Brighton, U.K.

[11] L. Chen and T. P. Pedersen. New group signature schemes (extended abstract). In Alfredo De Santis, editor, *Advances in Cryptology—EUROCRYPT 94*, volume 950 of *Lecture Notes in Computer Science*, pages 171–181. Springer-Verlag, 1995, 9–12 May 1994.

[12] Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In Yvo G. Desmedt, editor, *Advances in Cryptology—CRYPTO '94*, volume 839 of *Lecture Notes in Computer Science*, pages 174–187. Springer-Verlag, 21–25 August 1994.

[13] A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In A. Odlyzko, editor, *Advances in Cryptology, Proc. of Crypto '86 (Lecture Notes in Computer Science 263)*, pages 186–194. Springer-Verlag, 1987. Santa Barbara, California, U. S. A., August 11–15.

[14] Oded Goldreich. Secure multi-party computation. http://www.wisdom.weizmann.ac.il/ oded/, 1998.

[15] Johan Hastad, A. W. Schrift, and Adi Shamir. The discrete logarithm modula a composite hides O(n) bits. *JCSS: Journal of Computer and System Sciences*, 47(3):376–404, 1993.

[16] Markus Jakobsson, Kazue Sako, and Russell Impagliazzo. Designated verifier proofs and their applications. *Lecture Notes in Computer Science*, 1070:143–154, 1996.