J. Cryptol. (2010) 23: 169–186
DOI: 10.1007/s00145-009-9042-5

Journal of
CRYPTOLOGY

The RSA Group is Pseudo-Free*

Daniele Micciancio

Department of Computer Science and Engineering, University of California at San Diego, La Jolla, CA 92093, USA
daniele@cs.ucsd.edu

Communicated by Dan Boneh

Received 17 November 2006 and revised 13 August 2008
Online publication 7 May 2009

Abstract. We prove, under the strong RSA assumption, that the group of invertible integers modulo the product of two safe primes is pseudo-free. More specifically, no polynomial-time algorithm can output (with non negligible probability) an unsatisfiable system of equations over the free Abelian group generated by the symbols $g_1, \ldots, g_n$, together with a solution modulo the product of two randomly chosen safe primes when $g_1, \ldots, g_n$ are instantiated to randomly chosen quadratic residues. Ours is the first provably secure construction of pseudo-free Abelian groups under a standard cryptographic assumption and resolves a conjecture of Rivest (Theory of Cryptography Conference—Proceedings of TCC 2004, LNCS, vol. 2951, pp. 505–521, 2004).

Key words. Cryptographic assumptions, Pseudo-free Abelian group, Strong RSA problem, Safe primes.

1. Introduction

Informally, the notion of "pseudo-free group," put forward by Hohenberger [11] and subsequently refined by Rivest [22], describes a finite computational group (i.e., a group that admits an efficient algorithmic implementation) with the security property that it is computationally hard to find solutions to any nontrivial equation over the group. More specifically, Rivest [22] defines pseudo-free (Abelian) groups as computational (commutative) groups such that no polynomial-time adversary, given random group elements $g_1, \ldots, g_n$ (chosen using an appropriate sampling procedure), can output (with non-negligible probability) an equation which is unsatisfiable over the free Abelian group generated by the symbols $g_1, \ldots, g_n$, together with a solution to the equation in the computational group. As shown in [22], pseudo-freeness is a very strong assumption, and it implies many other computational assumptions typically used in cryptography, like the hardness of computing discrete logarithms and the RSA assumption in its standard

* A preliminary version of this work appeared in *Advances in Cryptology, Proceedings of EUROCRYPT 2005*, LNCS, vol. 3494, pp. 387–493, Springer.

and strong version. Each of these computational assumptions corresponds to a specific class of equations, e.g., the *strong RSA assumption* asserts that it is computationally infeasible to come up with an equation of the form $x^e = g$ (which is unsatisfiable over the free group $\{g^i : i \in \mathbb{Z}\}$ for $e > 1$) together with a solution $x = h$ such that $h^e = g$ in the multiplicative group $\mathbb{Z}_N^*$ of the invertible integers modulo the product $N = PQ$ of two large primes.

Free groups are widely used in computer science, and most modern cryptography relies on the hardness of computational problems over finite groups. So, as argued in [22], pseudo-free groups are a very interesting notion from a cryptographic perspective. Moreover, (non-Abelian) free groups are used in the so-called Dolev–Yao model [7] for the symbolic analysis of public-key cryptographic protocols. In the last few years, there have been several efforts to bridge the gap between the symbolic model of [7] (typically used in the area of formal methods for the analysis of security protocols) and the standard computational model used in cryptography (see, for example, [1,3,10,12,17–20]) with the goal of proving computational soundness results for symbolic analysis methods. An interesting question is whether pseudo-free groups can be used to extend (in a computationally sound way) the Dolev–Yao security model (in which encryption and decryption are viewed as black-box operations with no algebraic properties) with richer data structures and cryptographic functions (e.g., homomorphic encryption schemes) that make fundamental use of computational groups. Other motivations for studying pseudo-free groups mentioned in [22] are the following:

- Using a stronger assumption (that subsumes many other common cryptographic assumptions, like the hardness of computing discrete logarithms and the strong RSA assumption) may make proofs easier.
- As the strong RSA assumption has been very useful in the construction of many cryptographic functions [4,6,8] which are not known to be secure under the standard version of the RSA assumption, assuming that a group is pseudo-free may allow an even wider range of applications.
- Pseudo-freeness has been linked [11] to the construction of specific cryptographic primitives, like directed transitive signature schemes, for which no solution is currently known. (See [21] for a recent work in this area.)

The main question left open by Rivest in [22] is: do pseudo-free groups exist?

In [22] Rivest suggested the RSA group $\mathbb{Z}_N^*$ (where $N = PQ$ is the product of two large primes) as a possible candidate pseudo-free Abelian group and nicknamed the corresponding conjecture the *super-strong RSA assumption*. In this paper we resolve Rivest's conjecture and prove that $\mathbb{Z}_N^*$ is pseudo-free under the strong RSA assumption, at least when $N = PQ$ is the product of two "safe primes" (i.e., odd primes such that $p = (P-1)/2$ and $q = (Q-1)/2$ are also prime[1]), a special class of prime numbers widely used in cryptography. In other words, we prove that if the *strong* RSA assumption holds true, then the *super-strong* RSA assumption also holds. Our result is the first example of provably secure pseudo-free group based on a standard cryptographic assumption. In fact, we prove that the RSA group satisfies an even stronger version of

[1] Equivalently, using more standard mathematical terminology, $P = 2p + 1$ and $Q = 2q + 1$ where p and q are *Sophie Germain* primes.

the pseudo-freeness property than the one defined in [22]: we show that no adversary can efficiently compute an unsatisfiable *system* of equations (as opposed to a single equation) together with a solution in the given computational group.

Our proof is based on a rewriting process that, starting from an arbitrary equation (or system of equations), yields simpler and simpler equations with the following properties:

- Unsatisfiable equations over the free group are mapped to unsatisfiable equations over the free group, and
- Solutions to the original equations (over a computational group) can be efficiently mapped to solutions to the resulting equations (over the same computational group).

Some of our transformations work for arbitrary groups and might be of independent interest. For example, we show how to transform systems of equations into a single equation (Theorem 3) and how to map equations in several variables to univariate equations (Lemma 3).

Organization The rest of the paper is organized as follows. In Sect. 2 we introduce basic definitions and notation for equations and groups. In Sect. 3 we prove that the RSA group satisfies the basic definition of pseudo-free group (involving a single equation). In Sect. 4 we extend the result to systems of equations. Section 5 concludes with a discussion of open problems.

2. Preliminaries

In this section we give some background about the mathematical structures studied in this paper. A function f is negligible if it decreases faster than any inverse polynomial, i.e., for any $c > 0$, there is an n_0 such that $|f(n)| \leq 1/n^c$ for all $n > n_0$. For any two positive integers a and b, the greatest common divisor of a and b is denoted $\gcd(a, b)$.

2.1. *Computational Groups*

A group is an algebraic structure with a binary associative operation $\circ$, a unary operation $()^{-1}$ (inverse), and a constant 1 (identity) satisfying the equational axioms

$$(x \circ y) \circ z = x \circ (y \circ z),$$
$$x \circ 1 = 1 \circ x = x,$$
$$x \circ (x)^{-1} = (x)^{-1} \circ x = 1.$$

A group is Abelian if the operation $\circ$ is commutative, i.e., it also satisfies

$$x \circ y = y \circ x.$$

An Abelian group G is *free* if there is a subset of group elements $A \subset G$ such that any $g \in G$ can be uniquely expressed as a product $g = \prod_{a \in A} a^{d_a}$, where $d_a \in \mathbb{Z}$ for all $a \in A$. It easily follows that for any set of symbols A, the free Abelian group generated by

A (denoted $\mathcal{F}(A)$) is isomorphic to the additive group $\mathbb{Z}^{|A|}$ of $|A|$-dimensional integer vectors.

In this paper we are interested in computational groups, i.e., groups that admit an efficient algorithmic implementation. In order to properly formalize the notion of computational group in the asymptotic computational setting, one needs to consider either infinite groups or infinite families $\mathcal{G} = \{G_N\}_{N\in\mathcal{N}}$ of finite groups. In this paper we focus on families of finite groups, as these are the groups most commonly used in cryptography, and for which the definition of pseudo-free group is nontrivial. (The free group itself is a trivial example of infinite pseudo-free group.)

Definition 1. Let $\mathcal{G} = \{G_N\}_{N\in\mathcal{N}}$ be a family of finite groups indexed by $N \in \mathcal{N} \subseteq \{0,1\}^*$. A computational group family (associated to $\mathcal{G}$) is defined by a collection of representation functions $\langle\cdot\rangle_N : G_N \to \{0,1\}^*$ (for $N \in \mathcal{N}$) such that the following operations can be performed in (probabilistic) polynomial (in the bit-size of N) time:

- Test membership in a group: given $N \in \mathcal{N}$ and $x \in \{0,1\}^*$, determine if $x = \langle y\rangle_N$ is the representation of a group element $y \in G_N$.
- Compute the group operation: given $N \in \mathcal{N}$, $\langle x\rangle_N$ and $\langle y\rangle_N$ (for any $x, y \in G_N$) compute $\langle x \circ y\rangle_N$.
- Invert group elements: given $N \in \mathcal{N}$ and $\langle x\rangle_N$ (for some $x \in G_N$), compute $\langle x^{-1}\rangle_N$.
- Compute the representation of the group identity element: given $N \in \mathcal{N}$, output $\langle 1\rangle_N$, where 1 is the identity element of G_N.
- Sample group elements: on input $N \in \mathcal{N}$, output the representation $\langle x\rangle$ of a randomly chosen group element $x \in \langle G\rangle$ (with not necessarily uniform probability distribution).

In the definition above, we focused on computational groups in which each group element has a unique representation, as all the computational groups studied in this paper have this property. (The definition of computational group can be easily extended to cases where group elements may have multiple representations, by introducing an efficiently computable equivalence relation on group representations.) For the groups considered in this paper, all operations (except sampling) can be performed in *deterministic* polynomial time, but this is not required by the definition of computational group. The requirement that membership in $\langle G_N\rangle_N$ be efficiently decidable is also not strictly necessary, but convenient, and all computational groups studied in this paper have this property. Also, sometimes the definition of computational group requires the distribution output by the sampling algorithm $\langle x\rangle_N \in \langle G_N\rangle_N$ to be uniform over G_N, while other times no sampling algorithm is required at all. In this paper $\langle x\rangle_N \in \langle G_N\rangle_N$ is an arbitrary sampling procedure, which is used to generate nontrivial group elements.

Throughout the paper, whenever a computational group family is clear from the context, we use the expression "computational group" to refer to a specific group G_N and associated representation function $\langle\rangle_N$ in the family. Also, for brevity, we identify the computational group $\langle G_N\rangle_N$ with the underlying mathematical group, and write $x \circ y$, x^{-1}, etc., to denote the corresponding operation on the representations of the group elements. We use multiplicative notation xy for the group binary operation $x \circ y$ and use

exponential notation x^n to denote the n-fold composition of x with itself. Formally, x^n is defined inductively by the rules $x^0 = 1$, $x^{n+1} = x \circ x^n$. The notation is extended to negative exponents in the obvious way $x^{-n} = (x^n)^{-1}$.

2.2. *Pseudo-free Groups*

Let X and A be two disjoint finite sets of variable and constant symbols. We define $X^{-1} = \{x^{-1} : x \in X\}$ and $A^{-1} = \{a^{-1} : a \in A\}$. A *group equation* over variables X and constants A is a pair $E = (w_1, w_2)$, usually written as $E : w_1 = w_2$, where w_1 and w_2 are words over the alphabets $(X \cup X^{-1})^*$ and $(A \cup A^{-1})^*$, respectively. (Many other equivalent definitions are possible, e.g., one can allow w_1 and w_2 to be arbitrary words over the alphabet $X \cup X^{-1} \cup A \cup A^{-1}$ and also restrict the definition to equations of the form $w = 1$. Any equation can be easily rewritten in any of these forms using the commutativity and associativity properties of Abelian groups.) Unless otherwise specified, we interpret E as an equation over the free group $\mathcal{F}(A)$. A *solution* to $E : w_1 = w_2$ (over the free group $\mathcal{F}(A)$) is a function $\sigma : X \to \mathcal{F}(A)$ such that $\sigma(w_1) = w_2$ (in $\mathcal{F}(A)$), where σ is extended to words over $X \cup X^{-1}$ homomorphically in the obvious way. We say that an equation $E : w_1 = w_2$ is *satisfiable* (over the free group) if it admits a solution. We say that it is *unsatisfiable* otherwise.

Let G be a (computational) group. A group equation over G (denoted E_α) is defined by an equation E over variables X and constants A, and a function $\alpha : A \to G$. A solution to equation $E_\alpha : w_1 = w_2$ is a function $\xi : X \to G$ such that $\xi(w_1) = \alpha(w_2)$.

For computational purposes, we assume that equations $E : w_1 = w_2$ are given using compact notation for expressions of the form a^i with the exponent i represented in binary, so that exponentially large exponents can be stored in polynomial space. This is easily seen to be equivalent to many other formalisms to compactly represent terms w_1, w_2, like, for example, the straight-line programs used in [11].

Intuitively, a computational group is pseudo-free if no efficient algorithm can find a nontrivial relation among randomly chosen group elements, i.e., an equation (or system of equations) which is unsatisfiable over the free group, together with a solution over the computational group. Since for any finite group G, the equation $x^{|G|+1} = a$ is unsatisfiable over the free group $\mathcal{F}(\{a\})$ but has solution $x = a$ over G, in order to properly define pseudo-free groups, we need to consider families of groups $\{G_N\}$ where N is chosen at random. In particular, given a randomly chosen N, the order of the group $o(G_N) = |G_N|$ should be hard to compute. Technically, we assume that the set of indexes $\mathcal{N}$ is endowed with a sequence of probability distributions $(\mathcal{N}_k)_k$ such that $\mathcal{N}_k$ can be sampled in (expected) polynomial (in k) time. Typically, $\mathcal{N}_k$ is the uniform distribution over all strings in $\mathcal{N}$ of length k, but other distributions are possible. The set of indexes $\mathcal{N}$, together with the polynomial-time sampling algorithm and associated probability distributions $\mathcal{N}_k$, is called a *probability ensemble*.

Definition 2. A family of computational groups $\mathcal{G} = \{G_N\}_{N \in \mathcal{N}}$ is pseudo-free (with respect to a probability ensemble $\mathcal{N}$) if for any set A of polynomial size $|A| = p(k)$ (where k is a security parameter) and probabilistic polynomial (in k) time algorithm $\mathcal{A}$, the following holds. Let $N \in \mathcal{N}_k$ be a randomly chosen group index, and $\alpha : A \to G_N$ a function defining $|A|$ group elements chosen independently at random according to the

computational group sampling procedure. Then, the probability that $\mathcal{A}(N, \alpha) = (E, \xi)$ outputs an unsatisfiable equation E (over variables X and constants A) together with a solution $\xi: X \to G_N$ to E_α over G_N is a negligible function in k.

2.3. *The RSA group*

In this paper, we study the group $\mathbb{Z}_N^*$ of invertible integers modulo N. This is a computational group, with the usual representation of each group element as an integer in $\{0, \ldots, N-1\}$. Membership $g \in \mathbb{Z}_N^*$ can be easily tested by computing the $\gcd(g, N)$ and checking that $\gcd(g, N) = 1$. The group $\mathbb{Z}_N^*$ can be efficiently sampled uniformly at random by picking an integer $g \in \{0, \ldots, N-1\}$ with uniform distribution, and checking if $g \in \mathbb{Z}_N^*$. However, in this paper, it is more convenient to consider the computational group $\mathbb{Z}_N^*$ together with a different sampling procedure that chooses g at random from a subgroup of $\mathbb{Z}_N^*$. An element $g \in \mathbb{Z}_N^*$ is called a *quadratic residue* if $g = h^2 \bmod N$ for some $h \in \mathbb{Z}_N^*$. The set of quadratic residues modulo N is denoted QR_N, and it is a subgroup of $\mathbb{Z}_N^*$. The subgroup QR_N can be efficiently sampled by picking $h \in \mathbb{Z}_N^*$ uniformly at random and setting $g = h^2 \bmod N$. Unless otherwise specified, in this paper we always consider the computational group $\mathbb{Z}_N^*$ with this sampling procedure that selects g uniformly at random from QR_N.

When $N = P \cdot Q$ is the product of two prime numbers, $\mathbb{Z}_N^*$ is commonly called an RSA group, after the encryption function of Rivest, Shamir, and Adleman [23], which started a widespread use of these groups in cryptography. In this paper we are interested in RSA groups where P and Q are primes of special form. A prime number p is called a Sophie Germain prime if $2p + 1$ is also prime. In the cryptographic literature, the number $2p + 1$ (where p is a Sophie Germain prime) is usually called a *safe prime*. In other words, a safe prime $P = 2p + 1$ is an odd prime number such that $p = (P - 1)/2$ is also prime. Safe primes are relatively easy to find in practice (e.g., by choosing p at random and testing p and $2p + 1$ for primality), although there is no known mathematical proof showing that there are infinitely many of them. Safe primes are widely used in cryptography. For example, the RSA group $\mathbb{Z}_N^*$ where $N = P \cdot Q$ is the product of two safe primes has been used in [6,8,9].

We say that a computational problem (parameterized by an integer k) is *asymptotically hard* if for every probabilistic polynomial (in k) time algorithm, the probability that the algorithm solves the problem (computed over the random choice of the input and the internal randomness of the algorithm) is a negligible function in k. For any $k \geq 1$, let $\mathcal{N}_k$ be the set of all safe prime products of bit-size bounded by k. We assume some standard probability distribution on $\mathcal{N}_k$, as typically used in cryptographic applications. For example, one can choose $N \in \mathcal{N}_k$ as the product of two safe primes selected independently and uniformly at random among all $(k/2)$-bit safe primes. The following computational problems are conjectured to be asymptotically hard and have been used as the basis for many cryptographic applications:

- Factoring problem: given a random integer $N \in \mathcal{N}_k$, compute prime factors P, Q such that $N = P \cdot Q$.
- RSA problem [23]: given a random integer $N \in \mathcal{N}_k$, an integer e relatively prime with $\phi(N) = (P - 1)(Q - 1)$, and a randomly chosen group element $\gamma \in \mathbb{Z}_N^*$, compute a $\xi \in \mathbb{Z}_N^*$ such that $\xi^e = \gamma \bmod N$.

– Strong RSA problem [4]: given a random integer $N \in \mathcal{N}_k$ and a randomly chosen group element $\gamma \in \mathbb{Z}_N^*$, output an integer $e > 1$ and a group element $\xi \in \mathbb{Z}_N^*$ such that $\xi^e = \gamma \bmod N$.

In this paper we are primarily interested in the strong RSA problem and its relation to pseudo-freeness. It is convenient to consider the following variant of the strong RSA problem where the input γ is chosen as a random quadratic residue:

– Strong QR-RSA problem [6]: given a random integer $N \in \mathcal{N}_k$ and a randomly chosen quadratic residue $\gamma \in \mathrm{QR}_N$, output an integer $e > 1$ and a group element $\xi \in \mathbb{Z}_N^*$ such that $\xi^e = \gamma \bmod N$.

It can be easily shown [6] that this variant is not any easier than the standard strong RSA problem.

Theorem 1 (See [6], Sect. 4). *If the strong RSA problem modulo safe prime products $\mathcal{N}$ is asymptotically hard, then the strong QR-RSA problem is also asymptotically hard (with respect to the same distribution ensemble $\mathcal{N}_k$).*

For any prime product $N = P \cdot Q$, the group $\mathbb{Z}_N^*$ has cardinality $o(\mathbb{Z}_N^*) = \phi(N) = (P-1)(Q-1)$, and it is isomorphic to $\mathbb{Z}_P^* \times \mathbb{Z}_Q^*$, with isomorphism given by $\xi \mapsto (\xi \bmod P, \xi \bmod Q)$. If $P = 2p+1$ and $Q = 2q+1$ are safe primes, the group $\mathbb{Z}_N^*$ has order $4pq$, and the subgroup $\mathrm{QR}_N \subset \mathbb{Z}_N^*$ has order $o(\mathrm{QR}_N) = pq$. In particular, all elements in QR_N have order[2] 1, p, q, or pq.

2.4. *Statistical Distance*

Let X and Y be two discrete random variables over a (countable) set A. The statistical distance between X and Y is the quantity

$$\Delta(X, Y) = \frac{1}{2} \sum_{a \in A} \bigl|\Pr\{X = a\} - \Pr\{Y = a\}\bigr|.$$

In this paper we use the fact that for any two random variables X and Y over set A, and predicate $p\colon A \to \{0, 1\}$,

$$\bigl|\Pr\bigl[p(X) = 1\bigr] - \Pr\bigl[p(Y) = 1\bigr]\bigr| \le \Delta(X, Y).$$

In particular, if $p(X)$ happens with nonnegligible probability, and $\Delta(X, Y)$ is negligible, then also $p(Y)$ happens with nonnegligible probability.

3. The RSA Group is Pseudo-Free

In this section we prove, under the strong RSA assumption, that the RSA group $\mathbb{Z}_N^*$ (where N is the product of two safe primes, and elements are sampled uniformly at random from QR_N) is pseudo-free.

[2] The order of an element γ in a group G is the smallest positive integer $o(\gamma) \ge 1$ such that $\gamma^{o(\gamma)} = 1$.

Theorem 2. *Assume that the strong RSA problem is asymptotically hard with respect to a distribution ensemble $\mathcal{N}$ over safe prime products. Then the computational group family $\mathbb{Z}_N^*$ of invertible integers modulo $N \in \mathcal{N}$ (with the modular multiplication group operation and uniform sampling procedure over QR_N) is pseudo-free with respect to the same distribution ensemble $\mathcal{N}$.*

Proof. Assume that $\mathbb{Z}_N^*$ is not pseudo-free, i.e., there is a probabilistic polynomial-time algorithm $\mathcal{A}$ that on input a randomly chosen $N \in \mathcal{N}_k$ and random group elements $\alpha: A \to \mathrm{QR}_N$ (for some polynomial-sized set A), outputs an unsatisfiable equation $E: w_1 = w_2$ (over constants A and variables X) together with a solution $\xi: X \to \mathbb{Z}_N^*$ to E_α over the group $\mathbb{Z}_N^*$. We use $\mathcal{A}$ to solve the strong QR-RSA problem for the same distribution of the modulus N. Namely, given a randomly chosen $N \in \mathcal{N}_k$ and $\gamma \in \mathrm{QR}_N$, we compute an integer $e > 1$ and group element $\xi \in \mathbb{Z}_N^*$ such that $\xi^e = \gamma$. By Theorem 1 this also implies an algorithm to solve the standard strong RSA problem.

The reduction works as follows. Let (N, γ) be an instance of the strong QR-RSA problem. We begin by checking if γ is a generator for QR_N. This can be easily done using the following lemma. We remark that here is where we use the assumption that γ is a quadratic residue. (This assumption will be explicitly used again only towards the end of the proof.) □

Lemma 1. *Let $N = P \cdot Q$ be the product of two distinct safe primes, and $\gamma \in QR_N$ a quadratic residue. Then γ is a generator for QR_N if and only if* $\gcd(\gamma - 1, N) = 1$.

Proof. Let $P = 2p + 1$ and $Q = 2q + 1$, where p and q are distinct primes. By the Chinese remainder theorem QR_N is isomorphic to $\mathrm{QR}_P \times \mathrm{QR}_Q$ with isomorphism

$$\gamma \mapsto (\gamma_p, \gamma_q) = (\gamma \bmod P, \gamma \bmod Q).$$

Let $o(\gamma_p)$ and $o(\gamma_q)$ be the orders of γ_p and γ_q in QR_P and QR_Q, respectively. Since $\gamma_p \in \mathrm{QR}_P$, we have $o(\gamma_p) \mid o(\mathrm{QR}_P) = p$, i.e., $o(\gamma_p) \in \{1, p\}$. Similarly, $o(\gamma_q) \in \{1, q\}$ and $o(\gamma) = o(\gamma_p) \cdot o(\gamma_q) \in \{1, p, q, pq\}$. Notice that γ is a generator for QR_N if and only if $o(\gamma) = pq$, or, equivalently, $o(\gamma_p) = p$ and $o(\gamma_q) = q$. Let $g = \gcd(\gamma - 1, N)$. Since g divides N, it must be $g \in \{1, P, Q, PQ\}$. We want to prove that $o(\gamma) = pq$ if and only if $g = 1$.

First assume that $g \neq 1$, or, equivalently, g is divisible by either P or Q (or both). Assume without loss of generality that P divides g. Then P also divides $\gamma - 1$, and $\gamma_p = 1$. This proves that $o(\gamma_p) = 1$ and $o(\gamma) = o(\gamma_p) \cdot o(\gamma_q) \neq pq$.

Now assume $o(\gamma) \neq pq$, i.e., either $o(\gamma_p) = 1$ or $o(\gamma_q) = 1$. Assume without loss of generality that $o(\gamma_p) = 1$. Then $\gamma = 1 \pmod P$ and $P \mid \gcd(\gamma - 1, N)$. So, $g = \gcd(\gamma - 1, N) \neq 1$. □

If γ is not a generator for QR_N, then we can easily solve the strong QR-RSA problem instance (N, γ) as described below. Given N and $\gamma \in \mathrm{QR}_N$, we compute $g = \gcd(\gamma - 1, N)$. Since $N = PQ$, it must be $g \in \{1, P, Q, PQ\}$. We distinguish three cases.

- If $g = PQ = N$, then N divides $\gamma - 1$, and $\gamma = 1 \pmod N$. So, we can immediately output a solution to the strong QR-RSA input problem (N, γ), e.g., $(\xi, e) = (1, 3)$.

- If $g \notin \{1, N\}$, then it must be $g \in \{P, Q\}$, and we can easily compute $\phi(N) = (P-1)\cdot(Q-1) = (g-1)((N/g)-1)$. This also easily yields a solution $(\xi, e) = (\gamma, \phi(N)+1)$ to the strong QR-RSA problem (N, γ).
- If $g = 1$, then by Lemma 1 γ is a generator for QR_N, and we proceed as follows.

In the rest of the proof we assume that γ is a generator of QR_N. We use γ to sample the group elements $\alpha(a) \in \mathrm{QR}_N$ and generate an input instance (N, α) for algorithm $\mathcal{A}$. Since $\mathcal{A}$ works only with nonnegligible probability, we need the input values $\alpha(a)$ to be distributed (almost) uniformly at random over QR_N. The following lemma shows that γ can be used to sample QR_N almost uniformly at random.

Lemma 2. *For any cyclic group G and generator $\gamma \in G$, if v is chosen uniformly at random from $\{0, \ldots, B-1\}$, then the statistical distance between γ^v and the uniform distribution over G is at most $|G|/2B$.*

Proof. Let v be chosen in the interval $\{0, \ldots, B-1\}$. Notice that for any $\gamma^i \in G$ (with $i \in \{0, \ldots, |G|-1\}$), $\gamma^v = \gamma^i$ if and only if $v = i \pmod{|G|}$. Therefore, the probability that $\gamma^v = \gamma^i$ is

$$\Pr\{\gamma^v = \gamma^i\} = \Pr\big\{v = i \pmod{|G|}\big\} = \left\lceil (B-i)/|G| \right\rceil / B.$$

In particular, this probability deviates from the uniform distribution by at most

$$\begin{aligned}\left|\Pr\{\gamma^v = \gamma^i\} - \frac{1}{|G|}\right| &= \left|\frac{1}{B}\left\lceil\frac{B-i}{|G|}\right\rceil - \frac{1}{|G|}\right| \\ &= \max\left\{\frac{1}{B}\left\lceil\frac{B}{|G|}\right\rceil - \frac{1}{|G|}, \frac{1}{|G|} - \frac{1}{B}\left\lceil\frac{B-(|G|-1)}{|G|}\right\rceil\right\} < \frac{1}{B}.\end{aligned}$$

So, the statistical distance between γ^i and the uniform distribution is at most

$$\frac{1}{2}\sum_{i=0}^{|G|-1}\left|\Pr\{\gamma^v = \gamma^i\} - \frac{1}{|G|}\right| < \frac{|G|}{2B},$$

as claimed. □

For any $a \in A$, choose $v_a \in \{0, \ldots, N\cdot|A|\cdot K - 1\}$ uniformly at random for some super-polynomial function $K(k) = k^{\omega(1)}$, and set $\alpha(a) = \gamma^{v_a}$. By Lemma 2, the statistical distance between $\alpha(a)$ and the uniform distribution over QR_N is at most $|\mathrm{QR}_N|/2N|A|K \le 1/2|A|K$. Since the values $\alpha(a)$ are independently chosen, the statistical distance between α and a uniformly chosen assignment is at most $1/2K = k^{-\omega(1)}$.

Invoke algorithm $\mathcal{A}$ on input (N, α). We know that when α is distributed uniformly at random, algorithm $\mathcal{A}$ is successful with nonnegligible probability $\delta(k) = k^{-O(1)}$. Since α is within negligible statistical distance $1/K(k)$ from uniform, $\mathcal{A}$ succeeds on input α at least with nonnegligible probability $\delta(k) - 1/K(k)$. In the rest of the proof, we assume that $\mathcal{A}$ is successful, and we consider the conditional success probability of the reduction. We will show that the conditional success probability is at least $3/8$.

Fix the value of N, generator $\gamma \in \mathrm{QR}_N$, and input (N, α) passed to algorithm $\mathcal{A}$. Let $E : w_1 = w_2$ and ξ be the equation and solution to E_α returned by $\mathcal{A}$. Remember that, for every $a \in A$, $\alpha(a)$ equals γ^{v_a} for a randomly chosen $v_a \in \{0, \ldots, N \cdot |A| \cdot K - 1\}$. For any $a \in A$, let $w_a = v_a \bmod pq$ and $z_a = (v_a - w_a)/pq$. We remark that although the values v_a are known, and w_a, z_a are uniquely determined by v_a, the values w_a and z_a cannot be easily computed from v_a because the product pq is not known. Therefore, the values w_a and z_a cannot be used in the reduction process. We will use w_a and z_a only in the analysis of the reduction.

Notice that, given w_a, the conditional distribution of z_a is uniform over a set

$$S_a = \left\{0, \ldots, \left\lfloor \frac{N|A|K - 1 - w_a}{pq} \right\rfloor\right\} \tag{1}$$

of size at least

$$|S_a| \geq 1 + \left\lfloor \frac{N|A|K - 1 - w_a}{pq} \right\rfloor \geq \left\lfloor \frac{N|A|K}{pq} \right\rfloor \geq 4|A|K \geq 4,$$

where we have used $w_a \leq pq - 1$ and $N > 4pq$. Also, given w_a, the value of $\alpha(a) = \gamma^{v_a} = \gamma^{w_a}$ is uniquely determined, and z_a is uniformly distributed over the set S_a independently from α, E, and ξ. In particular, the integers $z_a \in S_a$ are distributed independently from the entire view (and success) of algorithm $\mathcal{A}$.

Assume that $\mathcal{A}$ is successful, i.e., E is unsatisfiable over $\mathcal{F}(A)$, and $\xi : X \to \mathbb{Z}_N^*$ is a valid solution to E_α. We use equation E and solution ξ to solve the original strong QR-RSA problem (N, γ). This is done in two steps. First, we transform equation E and solution ξ to E_α, into a new unsatisfiable equation E' and solution ξ' to E'_α containing only one variable symbol. Then, E' and ξ' are used to solve the strong QR-RSA problem (N, γ).

The equation and solution (E, ξ) are transformed into a univariate equation and solution (E', ξ') using the following lemma.

Lemma 3. *For any computational group family $\mathcal{G}$, there is a polynomial-time algorithm that on input an equation E over constants A and variables X, a group G from $\mathcal{G}$, and a variable assignment $\xi : X \to G$, outputs a univariate equation E' and value $\xi' \in G$ such that*

- *If E is unsatisfiable over the free group $\mathcal{F}(A)$, then E' is also unsatisfiable over $\mathcal{F}(A)$; and*
- *For any assignment $\alpha : A \to G$, if ξ is a solution to E_α, then ξ' is a solution to E'_α.*

Proof. Fix the input equation $E : \prod_{x \in X} x^{e_x} = \prod_{a \in A} a^{d_a}$ and assignment $\xi : X \to G$ from the variables X to a computational group G. Using the extended Euclidean algorithm, we compute $e = \gcd(e_x : x \in X)$ and integers e'_x $(x \in X)$ such that $\sum_x e_x e'_x = e$. The output equation is $E' : x^e = \prod_{a \in A} a^{d_a}$, with solution $\xi' = \prod_{x \in X} \xi(x)^{e_x/e}$. We need to prove that this output has the desired properties.

Assume that E' has a solution over the free group $\mathcal{F}(A)$. We want to prove that also E has a solution over $\mathcal{F}(A)$. Let $\xi' \in \mathcal{F}(A)$ be a solution to E', i.e., $(\xi')^e = \prod_{a \in A} a^{d_a}$. For

any $x \in X$, define $\xi(x) = (\xi')^{e'_x}$. The variable assignment ξ is a solution to E because

$$\prod_x \xi(x)^{e_x} = (\xi')^{\sum_x e_x e'_x} = (\xi')^e = \prod_a a^{d_a}.$$

This shows that E is satisfiable over the free group $\mathcal{F}(A)$ too and proves the first property.

Now, fix an assignment $\alpha : A \to G$, and assume that $\xi : X \to G$ is a solution to E_α, i.e., $\prod_x \xi(x)^{e_x} = \prod_a \alpha(a)^{d_a}$ in G. Then

$$(\xi')^e = \left(\prod_x \xi(x)^{e_x/e}\right)^e = \prod_x \xi(x)^{e_x} = \prod_a \alpha(a)^{d_a},$$

i.e., ξ' is a solution to E'_α over G. □

At this point we have an unsatisfiable equation of the form $E' : x^e = \prod_a a^{d_a}$ and a solution $\xi' \in \mathbb{Z}_N^*$ to E'_α. Notice that E' is satisfiable over the free group $\mathcal{F}(A)$ if and only if $e | \gcd(d_a : a \in A)$. So, it must be $e \nmid \gcd(d_a : a \in A)$. Also, from the definition of $\alpha(a)$ we know that

$$(\xi')^e = \prod_a \alpha(a)^{d_a} = \gamma^{\sum_a v_a d_a}. \tag{2}$$

In the rest of the proof we distinguish various cases, depending on the value of $\gcd(e, pq)$.

- If $\gcd(e, pq) = pq$ and $e \neq 0$, then we can immediately output the solution $(\gamma, |e| + 1)$ to the strong QR-RSA problem (N, γ) because $o(\gamma) = pq$ and $\gamma^{|e|+1} = \gamma \cdot \gamma^{\pm e} = \gamma \pmod N$. We remark that, although we cannot compute $\gcd(e, pq)$ (or even check if $\gcd(e, pq) = pq$) because pq is not known, we can guess that this is the case and simply check if $(\gamma, |e| + 1)$ is indeed a solution to the given strong QR-RSA problem. Similar remarks apply to the other cases below.
- If $\gcd(e, pq) \in \{p, q\}$, then $o(\gamma^e) = pq/\gcd(e, pq) \in \{p, q\}$. In particular, γ^e is not a generator of QR_N, and, by Lemma 1, $\gcd(\gamma^e - 1, N) \neq 1$. Since $\gamma^e \neq 1 \pmod N$, we also have $\gcd(\gamma^e - 1, N) \neq N$. Therefore, it must be $g = \gcd(\gamma^e - 1, N) \in \{P, Q\}$. So, we can compute $\phi(N) = (P - 1)(Q - 1) = (g - 1)((N/g) - 1)$ and output the solution $(\gamma, \phi(N) + 1)$ to the strong QR-RSA problem (N, γ).
- The remaining cases are where $e = 0$ or $\gcd(e, pq) = 1$ and are described below.

If $e = 0$, Lemma 4 below shows that $d = \sum_a v_a d_a = 0$ with probability at most $1/4$. It follows that with probability at least $3/4$, $(\gamma, |d| + 1)$ is a solution to the strong QR-RSA problem (N, γ) because $|d| + 1 > |d| \geq 1$ and

$$\gamma^{|d|+1} = \gamma \cdot \gamma^{\pm d} = \gamma \cdot \xi^{\pm 0} = \gamma.$$

So, the conditional success probability of the reduction is at least $3/4 > 3/8$.

Lemma 4. *The conditional probability (given α, $e = 0$, and $\{d_a : a \in A\}$ such that $e \nmid \gcd\{d_a : a \in A\}$) that $d = \sum_a v_a d_a \neq 0$ is at least $3/4$.*

Proof. We know that $v_a = w_a + pqz_a$, where each $z_a \in S_a$ is chosen independently and uniformly at random from a set of size $|S_a| \geq 4$. Since e does not divide $\gcd(d_a : a \in A)$, there exists an $\hat{a} \in A$ such that $d_{\hat{a}} \neq 0$. Fix the value of v_a for all $a \neq \hat{a}$. Since $d = 0$ for at most one value of $z_{\hat{a}} \in S_{\hat{a}}$, and $z_{\hat{a}}$ is independent of $\mathcal{A}$'s view $(\alpha, e, \{d_a\}_{a \in A})$, the conditional probability that $d = 0$ is at most $1/|S_{\hat{a}}| \leq 1/4$. □

The last case to consider is where $\gcd(e, pq) = 1$ and $e \neq 0$. This time, we first show that $e \nmid d$ with probability at least $3/8$.

Lemma 5. *The conditional probability (given* α, $\gcd(e, pq) = 1$, *and* $\{d_a : a \in A\}$ *such that* $e \nmid \gcd\{d_a : a \in A\}$*) that* e *does not divide* $d = \sum_a v_a d_a$ *is at least* $3/8$.

Proof. Since e does not divide $\gcd(d_a : a \in A)$, e does not divide $d_{\hat{a}}$ for some $\hat{a} \in A$. Remember that $v_a = w_a + pqz_a$, where the conditional distribution of z_a (given w_a) is uniform over the set S_a. Moreover, since $\gcd(e, pq) = 1$, pq is invertible modulo e. We want to bound the probability that e divides d, or equivalently, $d = 0 \pmod e$. Solving $d = 0 \pmod e$ for $z_{\hat{a}}$, we get $z_{\hat{a}} = -(\sum_{a \neq \hat{a}} v_a d_a + w_{\hat{a}})/pq \pmod e$. Since $z_{\hat{a}}$ is chosen uniformly at random in the interval $S_{\hat{a}}$, this happens with probability at most

$$\frac{\lceil |S_{\hat{a}}|/e \rceil}{|S_{\hat{a}}|} \leq \frac{|S_{\hat{a}}| + e - 1}{e \cdot |S_{\hat{a}}|} = \frac{1}{e} + \frac{1}{|S_{\hat{a}}|} - \frac{1}{e \cdot |S_{\hat{a}}|} \leq 5/8,$$

where we have used the fact that $|S_{\hat{a}}| \geq 4$ and $e \geq 2$ are integers. So, $e \nmid d$ with probability at least $3/8$. □

Let $e' = e/t$ and $d' = d/t$ where $t = \gcd(e, d)$. Assuming that $e \nmid d$ (which, by Lemma 5, happens with probability at least $3/8$), we have $t \neq e$, and consequently $e' = e/t > 1$. Notice that from $\gcd(e, pq) = 1$ and $t | e$ we get also $\gcd(t, o(\mathrm{QR}_N)) = \gcd(t, pq) = 1$. Therefore the congruence $\xi^{e't} = \gamma^{d't} \pmod N$ implies $\xi^{2e'} = \gamma^{2d'} \pmod N$. (Notice that $\xi^{e'} = \gamma^{d'} \pmod N$ does not necessarily follow from $\xi^{e't} = \gamma^{d't} \pmod N$ because ξ may not be a quadratic residue, and $\gcd(t, o(\mathbb{Z}_N^*))$ may equal 2. However, if we square both terms, we get that $\xi^{2e'}, \gamma^{2d'} \in \mathrm{QR}_N$, and $\gcd(t, o(\mathrm{QR}_N)) = 1$ is enough to conclude that $\xi^{2e'} = \xi^{2e/t} = \gamma^{2d/t} = \gamma^{2d'} \pmod N$.)

At this point, we have (γ, ξ, e', d') such that $\xi^{2e'} = \gamma^{2d'} \pmod N$, $e' > 1$, and $\gcd(e', d') = 1$. We know that N divides $\xi^{2e'} - \gamma^{2d'} = (\xi^{e'} - \gamma^{d'})(\xi^{e'} + \gamma^{d'})$. If $\xi^{e'} \neq \pm\gamma^{d'}$, then we can compute the factorization $\{P, Q\} = \{\gcd(N, \xi^{e'} - \gamma^{d'}), \gcd(N, \xi^{e'} + \gamma^{d'})\}$ of $N = PQ$ and $\phi(N) = (P - 1)(Q - 1)$, which immediately yields a solution $(\gamma, \phi(N) + 1)$ to the strong QR-RSA problem (N, γ).

So, assume that $\xi^{e'} = \pm\gamma^{d'}$. If $\xi^{e'} = \gamma^{d'}$, we can use the Euclidean algorithm to compute two integers e'' and d'' such that $e'e'' + d'd'' = \gcd(e', d') = 1$, and output $(\xi^{d''}\gamma^{e''}, e')$. This is a valid solution to the strong QR-RSA problem (N, γ) because $e' > 1$ (as a consequence of Lemma 5) and

$$(\xi^{d''}\gamma^{e''})^{e'} = \xi^{e'd''}\gamma^{e'e''} = \gamma^{d'd'' + e'e''} = \gamma.$$

Finally, we observe that if $\xi^{e'} = -\gamma^{d'}$, then e' must necessarily be odd (this is so because $(-\gamma^{d'})^{pq} = -1$, and therefore $\xi^{e'} = -\gamma^{d'}$ is not a quadratic residue), and consequently $(-\xi^{e'}) = -\xi^{e'} = \gamma^{d'}$. So, the last case $\xi^{e'} = -\gamma^{d'}$ immediately reduces to the previous one $\xi^{e'} = \gamma^{d'}$ by replacing ξ with $-\xi$.

4. Systems of Equations

The intuition behind the definition of pseudo-free group is that no polynomial-time adversary can "prove" that the given computational group is not free. The "proofs" implicit in Definition 2 consist of a single equation which is unsatisfiable over the free group but satisfiable over the computational group. This choice is motivated by the fact that unsatisfiability of equations over free groups and satisfiability over computational groups can be efficiently demonstrated. (Specifically, unsatisfiability over free Abelian groups is decidable in polynomial time, and satisfiability over arbitrary computational groups can be proved by giving a satisfying assignment.) An immediate extension that comes to mind is to consider systems of equations. Satisfiability for systems of equations is defined in the obvious way: a variable assignment satisfies a system of equations if it simultaneously satisfies all the equations in the system. As observed in [22], for the case of non-Abelian free groups, the results in [14] (see also [13, Lemma 3 and Corollaries 2 and 3]) allow one to combine systems of equations into a single equation. Specifically, the method is based on showing that the two equations $x = 1$ and $y = 1$ are equivalent to the single equation $x^2ax^2a^{-1} = (yby b^{-1})^2$, and it allows us to transform any finite system of equations into a single equation with exactly the same set of solutions. Unfortunately, the same is not true for Abelian groups, and the set of solutions to a system of equations cannot in general be represented by a single equation. Consider, for example, the equations $x = 1$ and $y = 1$. The solution to this system is clearly unique. However, no single equation in two variables can have a unique solution. (Any bivariate equation has always either zero or infinitely many solutions over the free group.)

In this section we show that in the case of Abelian groups, it is still possible to transform systems of equations into a single equation which is equivalent to the system, but in a weaker sense than having exactly the same set of solutions. The transformation maps any system of equations to a single equation whose solution set is a superset of the solutions to the system. However, if the system is unsatisfiable, then also the single equation is guaranteed to be unsatisfiable. This weaker notion of equivalence is enough to prove that Definition 2 is equivalent to the following seemingly stronger definition.

Definition 3. A family of computational groups $\mathcal{G} = \{G_N\}_{N\in\mathcal{N}}$ is pseudo-free if (with the notation of Definition 2) the probability that $\mathcal{A}(N, \alpha) = (\{E^i\}_{i\in I}, \xi)$ outputs an unsatisfiable *system* of equations $\{E^i\}_{i\in I}$ (over variables X and constants A) together with a solution $\xi: X \to G_N$ to $\{E^i_\alpha\}_{i\in I}$ over G_N is a negligible function in k.

The transformation from systems of equations to single equations is described in the following theorem.

Theorem 3. *There is a polynomial-time algorithm that on input a system of equations $\{E^i\}_{i\in I}$ over constants A and variables X, outputs a single equation E over the same sets of constants A and variables X such that the following holds*:

- *If* $\{E^i\}_{i\in I}$ *is unsatisfiable* (*over the free Abelian group generated by* A), *then* E *is also unsatisfiable*; *and*
- *For any computational group* G *and assignment* $\alpha: A \to G$, *any solution* $\xi: X \to G$ *to* $\{E^i_\alpha\}_{i\in I}$ *is also a solution to* E_α.

The proof of the theorem is based on elementary lattice techniques. For a detailed introduction to lattices and their computational complexity, the reader is referred to [16]. Here we briefly recall the basic definitions and simple facts about lattices used in the proof of Theorem 3. For any matrix $\mathbf{M}$ with rational entries, the lattice generated by a matrix $\mathbf{M} = [\vec{m}_1, \ldots, \vec{m}_n]$ is the set $\mathcal{L}(\mathbf{M}) = \{\sum_i x_i \vec{m}_i : x_i \in \mathbb{Z} \text{ for } i = 1, \ldots, n\}$ of all integer linear combinations of the columns of $\mathbf{M}$. There is a polynomial-time algorithm that on input two rational matrices $\mathbf{M}$ and $\mathbf{M}'$, determines if $\mathcal{L}(\mathbf{M}) \subseteq \mathcal{L}(\mathbf{M}')$, and if not, finds a vector $\vec{u} \in \mathcal{L}(\mathbf{M}) \setminus \mathcal{L}(\mathbf{M}')$. The dual of a lattice $\mathcal{L}(\mathbf{M})$ is the set of all vectors $\vec{u}$ in the linear span of the columns of $\mathbf{M}$ that have integer scalar product with all lattice vectors in $\mathcal{L}(\mathbf{M})$. The dual of a lattice is a lattice, and the dual of the dual of a lattice equals the original lattice. The dual of a lattice $\mathcal{L}(\mathbf{M})$ is denoted $\hat{\mathcal{L}}(\mathbf{M})$. Moreover, there is a polynomial-time algorithm that on input a rational matrix $\mathbf{M}$ outputs a rational matrix $\mathbf{M}'$ such that $\mathcal{L}(\mathbf{M}') = \hat{\mathcal{L}}(\mathbf{M})$. It immediately follows from the definition of dual lattice that $\mathcal{L}(\mathbf{M})$ is a sub-lattice of $\mathcal{L}(\mathbf{M}')$ (i.e., $\mathcal{L}(\mathbf{M}) \subseteq \mathcal{L}(\mathbf{M}')$) if and only if $\hat{\mathcal{L}}(\mathbf{M}')$ is a sub-lattice of $\hat{\mathcal{L}}(\mathbf{M})$ (i.e., $\hat{\mathcal{L}}(\mathbf{M}') \subseteq \hat{\mathcal{L}}(\mathbf{M})$). We are now ready to prove Theorem 3.

Proof. Let $\{E^i\}_{i\in I}$ be a system of equations over the set of constant symbols A and variables X, and let $\sigma: X \to \mathcal{F}(A)$ be a generic variable assignment. Write each equation E^i and the assignment $\sigma(x)$ as

$$E^i: \prod_{x\in X} x^{e_{i,x}} = \prod_{a\in A} a^{d_{i,a}},$$

$$\sigma(x) = \prod_{a\in A} a^{s_{x,a}},$$

where the $e_{i,x}$, $d_{i,a}$, and $s_{x,a}$ are integers for all $i \in I$, $x \in X$, and $a \in A$. We use notation $e_{*,*}$ to denote the matrix with $|I|$ rows and $|X|$ columns with integer entries $(e_{i,x})_{i\in I, x\in X}$, and $e_{i,*}$ and $e_{*,x}$ to denote the rows and columns of matrix $e_{*,*}$. The matrices $d_{*,*}$, $s_{*,*}$ and vectors $d_{i,*}$, $d_{*,a}$, $s_{x,*}$, $s_{*,a}$ are defined similarly. Notice that σ is a solution to the system of equations over the free group if and only if

$$\sum_{x\in X} e_{i,x} s_{x,a} = d_{i,a}$$

for all $i \in I$ and $a \in A$, or, equivalently, in matrix notation, $e_{*,*}s_{*,*} = d_{*,*}$. So, the system of equations is solvable over the free group if and only if the integer lattice $\mathcal{L}(e_{*,*})$ contains $\mathcal{L}(d_{*,*})$ as a sub-lattice. Moreover, the two lattices satisfy $\mathcal{L}(e_{*,*}) \supseteq \mathcal{L}(d_{*,*})$ if and only if their duals satisfy the reverse inclusion $\hat{\mathcal{L}}(e_{*,*}) \subseteq \hat{\mathcal{L}}(d_{*,*})$. The inclusion $\hat{\mathcal{L}}(e_{*,*}) \subseteq \hat{\mathcal{L}}(d_{*,*})$ can be checked using standard techniques, and if it is not satisfied, one can efficiently find a vector $(u_i)_{i\in I} \in \hat{\mathcal{L}}(e_{*,*})$ such that $(u_i)_{i\in I} \notin \hat{\mathcal{L}}(d_{*,*})$.

If $\hat{\mathcal{L}}(e_{*,*}) \subseteq \hat{\mathcal{L}}(d_{*,*})$, then the system of equations $\{E^i\}_{i\in I}$ is satisfiable over the free group, and the algorithm can simply output an arbitrary equation $E = E^i$ from the system. Clearly, any solution to the system is also a solution to E. Moreover, the other condition in the theorem is vacuously satisfied because $\{E^i\}_{i\in I}$ is satisfiable over the free group.

So, let us assume that $\hat{\mathcal{L}}(e_{*,*}) \not\subseteq \hat{\mathcal{L}}(d_{*,*})$, and let $u_* = (u_i)_{i\in I}$ be a vector such that $(u_i)_{i\in I} \in \hat{\mathcal{L}}(e_{*,*}) \setminus \hat{\mathcal{L}}(d_{*,*})$. We know that $\sum_i u_i e_{i,x}$ is an integer for all $x \in X$ because u_* belongs to the dual lattice $\hat{\mathcal{L}}(e_{*,*})$. Moreover, since $\mathcal{L}(e_{*,*})$ is an integer lattice, all entries u_i are rational numbers. It follows that for any $a \in A$, $\sum_i u_i d_{i,a}$ is a rational number, but $\sum_i u_i d_{i,a}$ is not an integer for some $a \in A$. Let c be the smallest integer such that $c \cdot \sum_i u_i d_{i,a}$ is an integer for all $a \in A$. In other words, let c be the least common multiple of the denominators of the fractions $\sum_i u_i d_{i,a}$ for all $a \in A$. The output of the algorithm is the equation

$$E\colon \prod_{x\in X} x^{c\cdot\sum_i u_i e_{i,x}} = \prod_{a\in A} a^{c\cdot\sum_i u_i d_{i,a}}.$$

We need to show that this equation satisfies the two properties in the theorem.

Let $\alpha\colon A \to G_N$ and $\xi\colon X \to G_N$ be two assignments such that ξ is a solution to the system $\{E^i_\alpha\}_{i\in I}$ over computational group G_N, i.e., $\prod_{x\in X} \xi(x)^{e_{i,x}} = \prod_{a\in A} \alpha(a)^{d_{i,a}}$ for all $i \in I$. It follows that

$$\begin{aligned}
\xi\left(\prod_{x\in X} x^{c\cdot\sum_i u_i e_{i,x}}\right) &= \prod_{i\in I}\left(\prod_{x\in X} \xi(x)^{e_{i,x}}\right)^{cu_i} \\
&= \prod_{i\in I}\left(\prod_{a\in A} \alpha(a)^{d_{i,a}}\right)^{cu_i} \\
&= \alpha\left(\prod_{a\in A} a^{c\cdot\sum_i u_i d_{i,a}}\right),
\end{aligned}$$

i.e., ξ is also a solution to equation E_α. This proves the second property. For the first property, since the system is unsatisfiable, we need to prove that E is also unsatisfiable over the free group $\mathcal{F}(A)$. Assume for contradiction that E is satisfiable over the free group and let $\sigma(x) = \prod_{a\in A} a^{s_{x,a}}$ be a solution, i.e.,

$$\prod_{x\in X}\left(\prod_{a\in A} a^{s_{x,a}}\right)^{c\cdot\sum_i u_i e_{i,x}} = \prod_{a\in A} a^{c\cdot\sum_i u_i d_{i,a}}.$$

Since the group $\mathcal{F}(A)$ is free, this is true if and only if

$$c\sum_{x\in X} s_{x,a}\sum_{i\in I} u_i e_{i,x} = c\cdot\sum_{i\in I} u_i d_{i,a}$$

for all $a \in A$. Since $\sum_{x\in X} s_{x,a}\sum_{i\in I} u_i e_{i,x}$ is an integer, the left-hand side of the last equation is a multiple of c. So, the right-hand side is also a multiple of c, and $\sum_i u_i d_{i,a}$

is an integer for all $a \in A$. But this is a contradiction because by construction (namely, by the choice of $(u_i)_{i\in I}$) there exists an $a \in A$ such that $\sum_i u_i d_{i,a}$ is not an integer. □

Corollary 1. *A family of computational groups $\{G_N\}_{N\in\mathcal{N}}$ satisfies Definition 2 if and only if it satisfies Definition 3.*

Proof. If a group family is pseudo-free in the sense of Definition 3, then it satisfies Definition 2 as well because single equations are a special case of systems containing only one equation. Conversely, assume that a group family does not satisfies Definition 3, i.e., there exists an adversary $\mathcal{A}$ that on input a group index $N \in \mathcal{N}$ and random assignment $\alpha: A \to G_N$, outputs an unsatisfiable system of equations $\{E_i\}_{i\in I}$ over constants A and variables X, together with a solution $\xi: X \to G_N$ to the system over the computational group G_N. Then, using Theorem 3, $\mathcal{A}$ can be easily converted into an adversary $\mathcal{A}'$, contradicting Definition 2. Namely, on input group index $N \in \mathcal{N}$ and random assignment $\alpha : A \to G_N$, adversary $\mathcal{A}'$ invokes $\mathcal{A}$ on input (N, α) to get an unsatisfiable system of equations $\{E_i\}_{i\in I}$ together with a solution ξ over the computational group G_N. Finally, $\mathcal{A}'$ transforms $\{E_i\}_{i\in I}$ into a single equation E using Theorem 3 and outputs E, ξ. By Theorem 3, equation E is unsatisfiable over the free group, and ξ is a solution to E_α over G_N, proving that the group family does not satisfies Definition 2. □

The following corollary immediately follows from Theorem 2 and Corollary 1.

Corollary 2. *Let $\mathcal{N}$ be a distribution ensemble over safe prime products such that the strong RSA problem modulo $N \in \mathcal{N}$ is hard. Then the family of computational groups $\mathbb{Z}_N^*$ of invertible integers modulo $N \in \mathcal{N}$ (with the modular multiplication group operation and uniform sampling procedure over QR_N) satisfies Definition 3, i.e., it is pseudo-free with respect to systems of equations.*

5. Conclusion

We have given the first example of provably secure pseudo-free group under standard cryptographic assumptions. In particular, we proved that the RSA group $\mathbb{Z}_N^*$ where N is the product of two safe primes is pseudo-free, assuming the hardness of the strong RSA problem. Many open problems remain. In this section we illustrate some of them.

Our proof uses the fact that N is the product of two safe primes, and elements are sampled uniformly at random from the subgroup QR_N of quadratic residues. A natural question is whether $\mathbb{Z}_N^*$ is pseudo-free even when N is the product of two arbitrary primes, and elements are sampled uniformly at random from the whole group $\mathbb{Z}_N^*$. Another open problem is to relax the hypothesis of Theorem 2 and prove that $\mathbb{Z}_N^*$ is pseudo-free assuming that factoring N is hard. Notice that this last problem is probably very hard, as it would imply that inverting the RSA function is at least as hard as factoring, a long standing open problem in cryptography. However, there are many other cryptographic problems that have been proved at least as hard as factoring, like the discrete logarithm problem [2], the Diffie-Hellman problem [15], and the generalized Diffie-Hellman problem [5] modulo Blum integers. We remark that while computing

discrete logarithms in pseudo-free groups is provably hard [22], no relation between pseudo-freeness and the Diffie-Hellman problem is currently known. An interesting open question, already posed in [22], is to show that the Diffie–Hellman problem in pseudo-free groups is computationally hard.

Another interesting problem is to find other examples of pseudo-free groups, beside $\mathbb{Z}_N^*$, and possibly proving their security based on standard cryptographic assumptions. Of particular interest would be to find a good candidate of non-Abelian pseudo-free group. Generalizing the notion of pseudo-free group even further, one can consider other standard algebraic structures (e.g., pseudo-free rings), or structures defined by an arbitrary set of equational axioms.

Finally, it would be nice to find applications of pseudo-free groups, as those mentioned in [22] and in the introduction, to demonstrate the usefulness of the notion of pseudo-free group. It might be the case that some applications require even stronger notions of pseudo-freeness than the one defined in [22]. In Sect. 4 we already considered extending the definition to systems of equations and proved that pseudo-freeness with respect to systems of equations (Definition 3) is equivalent to the basic definition of pseudo-free group. Another possible extension is to consider more general Boolean combinations of equations, e.g., one can consider systems of equations $w_1 = w_2$ and inequations $w_1 \neq w_2$. For example, $x^2 = 1$ and $x \neq 1$ cannot be simultaneously satisfied over the free group but admit a solution $x = N - 1$ in $\mathbb{Z}_N^*$ for any $N \neq 2$. We remark that the satisfiability problem over free Abelian groups for arbitrary Boolean combinations of equations is NP-hard. (For example, 3SAT can be immediately reduced to such a formula mapping each Boolean variable x to a corresponding equation $x = 1$.) So, some unsatisfiable formula do not have short (polynomial-size) proofs of unsatisfiability, unless NP = coNP. Extensions of the notion of pseudo-free group to general Boolean combinations of formulas should require the adversary to output not only an unsatisfiable formula over the free group (together with a solution over the computational group) but also a short and easily verifiable proof that the formula is indeed unsatisfiable.

Acknowledgements

The author would like to thank Ron Rivest for an interesting conversation during TCC'04 that led to this work and an anonymous referee for carefully reading a preliminary version of this paper and suggesting a substantial simplification to the proof of Theorem 2. This research was supported in part by NSF grants CNS-0430595 and CCF-0634909. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

[1] M. Abadi, P. Rogaway, Reconciling two views of cryptography (The computational soundness of formal encryption). *J. Cryptol.* **15**(2), 103–127 (2002)

[2] E. Bach, Discrete logarithms and factoring. Technical Report CSD-84-186, University of California at Berkeley (1984)
[3] M. Backes, B. Pfitzmann, M. Waidner, A composable cryptographic library with nested operations (extended abstract). In: *Computer and Communications Security—Proceedings of CCS'03*, pp. 220–230 (2003)
[4] N. Baric', B. Pfitzmann, Collision-free accumulators and fail-stop signature schemes without trees. In: *Advances in Cryptology—Proceedings of EUROCRYPT'97*. LNCS, vol. 1233, pp. 480–494 (1997)
[5] E. Biham, D. Boneh, O. Reingold, Breaking generalized Diffie–Hellman modulo a composite is no easier than factoring. *Inf. Process. Lett.* **70**(2), 83–87 (1999)
[6] R. Cramer, V. Shoup, Signature schemes based on the strong RSA assumption. *ACM Trans. Inf. Syst. Secur.* **3**(3), 161–185 (2000). Preliminary version in CCS'99
[7] D. Dolev, A. Yao, On the security of public key protocols. *IEEE Trans. Inf. Theory* **29**(2), 198–208 (1983)
[8] E. Fujisaki, T. Okamoto, Statistical zero knowledge protocols to prove modular polynomial relations. In: *Proceedings of CRYPTO '97*. LNCS, vol. 1294, pp. 16–30 (1997)
[9] R. Gennaro, T. Rabin, H. Krawczyk, RSA-based undeniable signatures. *J. Cryptol.* **13**(4), 397–416 (2000). Preliminary version in CRYPTO'97
[10] P. Gupta, V. Shmatikiv, Towards computationally sound symbolic analysis of key exchange protocols (extended abstract). In: *Formal Methods in Security Engineering—Proceedings of FMSE'05*, ed. by V. Atluri, P. Samarati, R. Küsters, J.C. Mitchell. Fairfax, VA, USA, pp. 23–32 (2005)
[11] S. Hohenberger, The cryptographic impact of groups with infeasible inversion. Master's thesis, Massachusetts Institute of Technology, EECS Dept., Cambridge, MA (2003)
[12] R. Impagliazzo, M.B. Kapron, Logics for reasoning about cryptographic constructions. *J. Comput. Syst. Sci.* **72**(2), 286–320 (2006). Preliminary version in FOCS'03
[13] O. Kharlampovich, A. Myasnikov, Implicit function theorem over free groups. *J. Algebra* **290**(1), 1–203 (2005)
[14] A.I. Mal'cev, On some correspondence between rings and groups. *Mat. Sb.* **50**, 257–266 (1960)
[15] K.S. McCurley, A key distribution system equivalent to factoring. *J. Cryptol.* **1**(2), 95–105 (1988)
[16] D. Micciancio, S. Goldwasser, *Complexity of Lattice Problems: a Cryptographic Perspective*, The Kluwer International Series in Engineering and Computer Science, vol. 671. (Kluwer Academic, Boston, 2002)
[17] D. Micciancio, S. Panjwani, Adaptive security of symbolic encryption. In: *Theory of Cryptography Conference—Proceedings of TCC*. LNCS, vol. 3378, pp. 169–187 (2005)
[18] D. Micciancio, B. Warinschi, Completeness theorems for the Abadi–Rogaway logic of encrypted expressions. *J. Comput. Secur.* **12**(1), 99–129 (2004). Preliminary version in WITS'02
[19] D. Micciancio, B. Warinschi, Soundness of formal encryption in the presence of active adversaries. In: *Theory of Cryptography Conference—Proceedings of TCC'04*. LNCS, vol. 2951, pp. 133–151 (2004)
[20] J.C. Mitchell, A. Ramanathan, A. Scedrov, V. Teague, A probabilistic polynomial-time calculus for the analysis of cryptographic protocols. *Theor. Comput. Sci.* **353**(1–3), 118–164 (2006). Preliminary version in MFPS'01
[21] G. Neven, A simple transitive signature scheme for directed trees. *Theor. Comput. Sci.* **396**(1–3), 277–282 (2008)
[22] R.L. Rivest, On the notion of pseudo-free groups. In: *Theory of Cryptography Conference—Proceedings of TCC'04*. LNCS, vol. 2951, pp. 505–521 (2004)
[23] R.L. Rivest, A. Shamir, L. Adleman, A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM* **21**, 120–126 (1978)