Revisiting the Uber Assumption in the Algebraic Group Model: Fine-Grained Bounds in Hidden-Order Groups and Improved Reductions in Bilinear Groups

Lior Rotem
School of Computer Science and Engineering, Hebrew University of Jerusalem, Israel

Abstract

We prove strong security guarantees for a wide array of computational and decisional problems, both in hidden-order groups and in bilinear groups, within the algebraic group model (AGM) of Fuchsbauer, Kiltz and Loss (CRYPTO '18). As our first contribution, we put forth a new fine-grained variant of the Uber family of assumptions in hidden-order groups. This family includes in particular the repeated squaring function of Rivest, Shamir and Wagner, which underlies their time-lock puzzle as well as the main known candidates for verifiable delay functions; and a computational variant of the generalized BBS problem, which underlies the timed commitments of Boneh and Naor (CRYPTO '00). We then provide two results within a variant of the AGM, which show that the hardness of solving problems in this family in a less-than-trivial number of steps is implied by well-studied assumptions. The first reduction may be applied in any group (and in particular, class groups), and is to the RSA assumption; and our second reduction is in RSA groups with a modulus which is the product of two safe primes, and is to the factoring assumption.

Additionally, we prove that the hardness of any computational problem in the Uber family of problems in bilinear groups is implied by the hardness of the q-discrete logarithm problem. The parameter q in our reduction is the maximal degree in which a variable appears in the polynomials which define the specific problem within the Uber family. This improves upon a recent result of Bauer, Fuchsbauer and Loss (CRYPTO '20), who obtained a similar implication but for a parameter q which is lower bounded by the maximal *total* degree of one of the above polynomials. We discuss the implications of this improvement to prominent group key-exchange protocols.

2012 ACM Subject Classification Security and privacy → Cryptography; Theory of computation → Cryptographic primitives

Keywords and phrases Algebraic group model, Uber assumption, Bilinear groups, Hidden-order groups

Digital Object Identifier 10.4230/LIPIcs.ITC.2022.13

Related Version *Full Version*: `https://eprint.iacr.org/2022/584`

Funding Supported by the Adams Fellowship Program of the Israel Academy of Sciences and Humanities and by the European Union's Horizon 2020 Framework Program (H2020) via an ERC Grant (Grant No. 714253).

1 Introduction

The algebraic group model (the AGM) was introduced by Fuchsbauer, Kiltz and Loss[1] [28] with the aim of striking a middle ground between the generic group model (the GGM) and the standard model. In contrast to the GGM, algorithms within the AGM (also known as algebraic algorithms) do receive the representation of group elements and may use it in any

[1] Following Abdalla, Benhamouda and Mackenzie [3] and Bernhard, Fischlin and Warinschi [10]. Additionally, the earlier works of Boneh and Venkatesan [18] and Paillier and Vergnaud [40] considered algebraic *reductions*, rather than algebraic adversaries.

way they see fit. The restriction, however, is that whenever an algebraic algorithm outputs a group element, it must provide alongside it a representation of it in the basis of its input group elements. This representation serves as an explanation as to how the output element was computed from the input elements in an algebraic manner. Given the less restrictive nature of the AGM when compared to the GGM,[2] a central line of research over the last couple of years has focused on establishing the security of cryptographic schemes and assumptions within the AGM; see for example [28, 39, 38, 1, 6, 16, 7, 24, 25, 29, 34, 37, 44, 2, 33, 35].

The sequentiality of repeated squaring

The "repeated squaring" function in hidden-order groups, first suggested by Rivest, Shamir and Wagner [43], serves as the basis for the main candidate constructions of both time-lock puzzles and of verifiable delay functions [13, 41, 47]. For years, however, the sequentiality of this function remained purely as an assumption, and there was no known reduction (in idealized models or otherwise) relating it to the hardness of a better-established assumptions. Recently, Katz, Loss and Xu [37] presented a strengthened version of the AGM (the strong AGM) and provided evidence for the sequentiality of repeated squaring within this model.[3] Concretely, they showed that any strongly-algebraic algorithm which manages to speed-up the repeated squaring function in the group QR_N of quadratic residues modulo N (where N is the product of two safe primes), can be used in order to factor the modulus N. Their result provides a novel and important corroboration for the sequentiality of repeated squaring in the group QR_N. However, it is limited in two respects: Firstly, it inherently relies on the algebraic structure of QR_N and does not apply to other hidden-order groups of interest, such as RSA groups or the class group of imaginary quadratic number fields [23, 11, 15, 47]. Secondly, it addresses only the repeated squaring function, and leaves out other possible fine-grained problems in hidden-order groups.

The Uber problem in bilinear groups

The Uber family of problems in bilinear groups was introduced by Boneh, Boyen and Goh [14, 19] as a unified framework for reasoning about computational problems in such groups. A problem in the family is parameterized by three tuples of multivariate polynomials $\vec{F}, \vec{H}, \vec{K}$ and three polynomials Q_1, Q_2, Q_T and is defined by the following task: Given the group elements $\{g_1^{F_i(\vec{x})}\}_i, \{g_2^{H_i(\vec{x})}\}_i, \{g_T^{K_i(\vec{x})}\}_i$ for a random vector $\vec{x}$, compute $g_1^{Q_1(\vec{x})}$, $g_2^{Q_2(\vec{x})}$ and $g_T^{Q_T(\vec{x})}$, where g_1, g_2 and g_T are the generators of the source groups and the target group, respectively. Bauer, Fuchsbauer and Loss [7] recently showed that in the AGM, the hardness of any problem in this family, as long as it does not admit a trivial solution, is implied by the hardness of the q-DLOG problem in one of the source groups.[4] Their result provides a clean and succinct characterization of the Uber family, reducing its hardness to that of a seemingly simpler and better-understood family of problems. However, the parameter q in their result is lower bounded by the *maximal total degree* of the polynomials in $\vec{F}, \vec{H}, \vec{K}$, which may not be optimal. To see why that is, consider the following toy problem for any integer n: Given a generator g and a tuple $(g^{x_1}, g^{x_2}, g^{x_3}, g^{x_1 x_3}, g^{x_2 x_3})$ of group elements for randomly-chosen

[2] The AGM may also be instantiated in the standard model from falsifiable assumptions, as demonstrated by the elegant work of Agrikola, Hofheinz and Kastner [5]. This is in contrast to the GGM [27].

[3] Another recent result [45], proving the equivalence of speeding-up repeated squaring and factoring within the generic ring model is discussed in Section 3.

[4] The q-DLOG problem in a cyclic group $\mathbb{G}$ is defined as: Given a generator g and the q group elements $g^x, \dots, g^{x^q}$ for a randomly chosen x, compute x.

x_1, x_2, x_3, compute $g^{x_1x_2x_3}$. On the one hand, the result of Bauer et al. can be used to conclude that the hardness of this problem in the AGM is implied by the hardness of the 2-DLOG problem. On the other hand, it is not hard to see that this problem is actually equivalent to the Computational Diffie-Hellman (CDH) problem. The CDH problem was proven equivalent to the DLOG problem (i.e., 1-DLOG) in the AGM [28], suggesting that the bound of Bauer et al. might not be optimal with respect to the parameter q.[5]

2 Our Results

In this work, we provide stronger hardness results within the AGM, both for a new fine-grained Uber family of problems in hidden-order groups that we put forth, and for the Uber family of problems in bilinear groups.

2.1 Our Results for Fine-Grained Computations in Hidden-Order Groups

A fine-grained Uber family

As our first contribution, we present a univariate and fine-grained variant of the Uber family of problems in hidden-order groups. Our family of problems generalizes the repeated squaring function [43], as well as well as a computational variant[6] of the generalized BBS problem underlying Boneh and Naor's timed commitments [12, 17] and refinements thereof [31] (see also [30, 32]). A problem in this family is parameterized by integers $u_1, \ldots, u_\ell$ and an integer w, and requires that the adversary computes x^w for a uniformly-chosen group element x, given $(x^{u_1}, \ldots, x^{u_\ell})$ as input. Of course, if one can efficiently express w as a linear combination of $u_1, \ldots, u_\ell$ with integer coefficients, then one can trivially compute x^w from $(x^{u_1}, \ldots, x^{u_\ell})$ using a polynomial number of group operations. Therefore, we carefully define what it means for a strongly-algebraic algorithm to non-trivially solve a problem in our new Uber family, also accounting for the possibility of parallel computations. Looking ahead, the repeated squaring function is obtained by setting $u_1 = 1$ and $w = 2^T$, whereas the generalized BBS problem is obtained by setting $u_1 = 0$, $u_i = 2^{2^{i-2}}$ for $i = 2, \ldots, k+2$ and $w = 2^{2^{k+1}}$. Moreover, the repeated squaring function cannot be trivially solved (according to our triviality notion) in less than T steps, whereas the generalized BBS problem cannot be trivially solved is less than 2^k steps.

The algebraic hardness of the fine-grained Uber problem

Within the strong AGM of Katz, Loss and Xu [37], we provide evidence for the hardness of our new family of problems. Firstly, we present a general hardness result which may be applied in any cryptographic group, and prove that the hardness of any problem in the fine-grained Uber family in a group $\mathbb{G}$ (i.e., the hardness of computing the target group element x^w in a less-than-trivial number of steps), is implied by the RSA assumption in the same group.

[5] Though in the specific case of the toy problem considered above, the result of Fuchsbauer, Kiltz and Loss [28] already shows it to be equivalent to the DLOG problem, this is not the case for general instances of the Uber family in bilinear groups, motivating Theorem 3 below.

[6] In practice, one can always achieve pseudorandomness from this computational variant heuristically by applying a cryptographic hash function (e.g., SHA) onto the output of the problem [8].

▶ **Theorem 1** (informal). *Let $\mathbb{G}$ be a group, let $\ell \in \mathbb{N}$ and let $u_1, \ldots, u_\ell, w \in \mathbb{Z}$. Let* A *be a strongly-algebraic algorithm for the $(u_1, \ldots, u_\ell, w)$-univariate fine-grained Uber problem in the group $\mathbb{G}$, which makes a less-than-trivial number of steps. Then, there exists an algorithm* B *for the RSA problem in $\mathbb{G}$ whose running time and success probability are polynomially-related to those of* A.

Theorem 1 immediately implies that any strongly-algebraic algorithm that computes the repeated squaring function in less than T steps in some group $\mathbb{G}$, or solves the generalized BBS problem in less than 2^k steps, can be used in order to solve the RSA problem in the group. Note that Theorem 1 assumes nothing about the group $\mathbb{G}$, and in particular can be applied in any group in which the RSA assumption is believed to hold, such as RSA groups, class groups of imaginary quadratic number fields, and the group QR_N with respect to arbitrary bi-prime moduli. Importantly, Theorem 1 provides evidence for the sequentiality of repeated squaring in class groups, as the RSA problem has been considered and studied in these groups for a while now (see for example [23, 11, 26] and the references therein), whereas the sequentiality of repeated squaring in these groups is a much newer assumption [15, 47]. As far as we are aware, this is the first result supporting the sequentiality of repeated squaring in class groups.

Our second hardness result for the fine-grained Uber problem considers RSA groups with a modulus N which is the product of two safe primes. Informally, within the strong AGM, we prove that in such groups, the hardness of any problem in the fine-grained Uber family is implied by the hardness of factoring N.

▶ **Theorem 2** (informal). *Let N be the product of two safe primes and let $\ell \in \mathbb{N}$ and let $u_1, \ldots, u_\ell, w \in \mathbb{Z}$. Let* A *be a strongly-algebraic algorithm for the $(u_1, \ldots, u_\ell, w)$-univariate fine-grained Uber problem in $\mathbb{Z}_N^*$ which makes a less-than-trivial number of steps. Then, there exists an algorithm* B *for factoring N whose running time and success probability are polynomially-related to those of* A.

Observe that Theorem 2 strictly strengthens the result of Katz, Loss and Xu [37] in two respects. Firstly, by considering our new fine-grained Uber family, which captures in particular the sequentiality of the repeated squaring function (considered by Katz, Loss and Xu), but also other problems, such as the generalized BBS problem [17]. Secondly, Theorem 2 considers RSA groups with respect to moduli which are the product of two safe primes, whereas Katz, Loss and Xu consider the group QR_N with respect to the same family of moduli. Since a uniformly-random element in $\mathbb{Z}_N^*$ is in QR_N with probability 1/4, the hardness of any problem within the fine-grained Uber family with respect to $\mathbb{Z}_N^*$ implies in particular its hardness with respect to QR_N.

Interestingly, to the best of our knowledge, Theorems 1 and 2 are the first applications of the algebraic group model (or a variant thereof) in non-cyclic groups. As far as we are aware, all previous reductions in the AGM were either in cyclic groups of prime order or in the cyclic group QR_N where N is the product of two safe primes. Hence, our work exemplifies for the first time the applicability of the AGM beyond cyclic groups.

2.2 Our Results for Bilinear Groups

The Algebraic Hardness of the Uber problem in bilinear groups

We strengthen the characterization of Bauer, Fuchsbauer and Loss [7] of the Uber family framework in bilinear groups. Concretely, let $\vec{F}, \vec{H}, \vec{K}$ be vectors of polynomials and let Q_1, Q_2, Q_T be polynomials. We prove that within the AGM, as long as these polynomials do not admit a trivial solution, the hardness of their respective problem in the Uber family is implied by the q-DLOG assumption, where q is lower bounded by the maximal degree in which *a variable appears* in $\vec{F}, \vec{H}, \vec{K}$.

▶ **Theorem 3** (informal). *Let $\mathcal{G}$ be a bilinear group, let $\vec{F}, \vec{H}, \vec{K}$ be vectors of m-variate polynomials and let Q_1, Q_2, Q_T be m-variate polynomials that do not admit a trivial solution to the $(\vec{F}, \vec{H}, \vec{K}, Q_1, Q_2, Q_T)$-Uber problem. Let q be the maximal degree in which a variable appears in $\vec{F}, \vec{H}, \vec{K}$. Then, for any algebraic algorithm* A *for the $(\vec{F}, \vec{H}, \vec{K}, Q_1, Q_2, Q_T)$-Uber problem in $\mathcal{G}$, there exists an algorithm* B *for the q-DLOG problem in one of the source groups, whose running time and success probability are polynomially-related to those of* A.

Theorem 3 strengthens the result by Bauer, Fuchsbauer and Loss, since the total degree of a polynomial is always at least the degree of each variable in it, and in many typical cases it is indeed strictly greater. For example, consider again our toy example from before: Given $g, g^{x_1}, g^{x_2}, g^{x_3}, g^{x_1x_3}$ and $g^{x_2x_3}$, compute $g^{x_1x_2x_3}$. Since the polynomials defining the input elements of this problem are all multilinear,[7] Theorem 3 implies that within the AGM, its hardness is implied by the hardness of the discrete logarithm problem. Recall that the result of Bauer et al. bases the hardness of the aforesaid problem only on the hardness of the seemingly easier 2-DLOG problem.

Application: Group Key Exchange

The toy-problem example might seem contrived at first sight, but it is actually a special case of the Group Computational Diffie-Hellman (G-CDH) problem, which underlies the highly-influential group key-exchange protocols of Bresson, Chevassut, Pointcheval and Quisquater [22, 20, 21]. This problem is parameterized by an integer n (which in group key-exchange applications represents the number of users in the group) and a collection Γ of subsets of $\{1, \ldots, n\}$. The adversary is given a generator g of the group, alongside the group elements $\left\{g^{\prod_{i \in \mathcal{S}} x_i}\right\}_{\mathcal{S} \in \Gamma}$ for uniformly-chosen $x_1, \ldots, x_n$, and is asked to compute $g^{x_1 \cdots x_n}$. Typically, in group key-exchange protocols Γ includes at least one subset of size $n-1$. In such cases, Theorem 3 reduces the hardness of the (n, Γ)-G-CDH problem to the hardness of the discrete logarithm problem, whereas the previous bound of Bauer et al. reduces it to the hardness of the $(n-1)$-DLOG problem, where n may be a very large integer and perhaps not even a-priori bounded.

The decisional Uber problem in bilinear groups

As our second contribution in bilinear groups, we extend Theorem 3 to the decisional setting. Within the decisional algebraic group model (DAGM) of Rotem and Segev [44] which we extend to accommodate analysis in asymmetric bilinear groups, we prove that the hardness of the *decisional* Uber problem in bilinear groups is implied by the hardness of the q-DLOG problem. Again, the parameter q in our results is the maximal degree in which some variable appears in polynomials defining the problem.

▶ **Theorem 4** (informal). *Let $\mathcal{G}$ be a bilinear group, let $\vec{F}, \vec{H}, \vec{K}$ be vectors of m-variate polynomials and let Q_T be an m-variate polynomial which does not admit a trivial solution to the $(\vec{F}, \vec{H}, \vec{K}, Q_T)$-Uber problem. Let q be the maximal degree in which a variable appears in $\vec{F}, \vec{H}, \vec{K}, Q_T$. Then, for any algorithm* A *for the decisional $(\vec{F}, \vec{H}, \vec{K}, Q_T)$-Uber problem in $\mathcal{G}$, there exists an algorithm* B *for the q-DLOG problem in one of the source groups, whose running time and success probability that are polynomially-related to those of* A.

[7] These are the polynomials X_1, X_2, X_3, X_1X_3 and X_2X_3.

The parameters $\vec{F}, \vec{H}, \vec{K}, Q_T$ play a similar role in the decisional setting to their role in the computational setting; for a formal definition of the decisional Uber problem, see the full version. Theorem 4 improves upon a result of Rotem and Segev [44], who showed a similar result, but in their work the parameter q is strictly greater than the maximal total degree of the polynomials defining the problem.

3 Additional Related Work

Bauer, Fuchsbauer and Loss [7] also considered additional variants of the (computational) Uber problem in bilinear groups. Concretely, they proved that their result extends to: The flexible Uber problem, in which the adversary can choose the target polynomials Q_1, Q_2, Q_T; the Uber problem for rational functions, in which the polynomials $\vec{F}, \vec{H}, \vec{K}, Q_1, Q_2$ and Q_T are replaced by rational functions; the Uber problem with decisional oracles, in which the adversary is given access to an oracle for checking whether tuples of group elements satisfy a polynomial relation in the exponent; and the flexible "GeGenUber" problem in which the adversary may choose the generators with respect to which the problem is defined. It seems that the techniques used by Bauer et al. to extend their results to all of the aforesaid variants can be used essentially as is in order to extend our results (Theorems 3 and 4) to accommodate these variants as well, but we leave this task to future work. We refer the reader to [7] for a formal definition of these variants of the Uber problem and for the techniques used by Bauer et al. in order to extend their result to these variants.

The concurrent work of van Baarsen and Stevens [46] also considered reductions supporting the sequentiality of repeated squaring in hidden-order groups. They propose a strengthening of the strong algebraic group model and within it, they reduce the task of speeding up repeated squaring to that of finding a multiple of the group's order. It seems that the results of van Baarsen and Stevens are incomparable to our results in hidden-order groups. On the one hand, we consider the more general fine-grained Uber problem, which we put forth, while they only consider the repeated squaring problem. Additionally, the model of van Baarsen and Stevens seems more restrictive for general hidden-order groups. It assumes that either every algorithm receives as input a set of generators or the ability to uniformly sample group elements (as discussed in their paper, the two assumptions are essentially equivalent). This is essentially the case for $\mathbb{Z}_N^*$, but in general groups, this poses an additional assumption. In contrast, our reduction from the RSA problem is algebraic and does not require the ability to publicly sample uniformly-random group elements. It also works if the group element x specifying the fine-grained Uber problem instance is sampled uniformly by the challenger in a private-coin manner, or if it comes from a non-uniform distribution. In the latter case, we solve that RSA problem for the same distribution. Finally, the reduction of van Baarsen and Stevens loses a factor of n in running time, where n is an upper bound on the number of generators of the group. Potentially, n can be as large as logarithmic in the order of the group. In comparison, our reductions are essentially tight. On the other hand, the reduction of van Baarsen and Steve is from the problem of finding the group's order, which is harder than the RSA problem we consider in general groups. In particular, their result establishes the equivalence, within the strong AGM (which is equivalent to their strengthened model in $\mathbb{Z}_N^*$, where sampling is easy), of speeding up repeated squaring in RSA groups and factoring the RSA modulus, for general bi-prime moduli (whereas we only establish this equivalence when the modulus is the product of two safe primes).

Rotem and Segev [45] put forth the notion of generic-ring delay functions within the generic ring model of Aggarwal and Maurer [4], and the notion of a *sequentiality depth* of such functions. Informally, they proved that in the ring $\mathbb{Z}_N$ where N is an RSA modulus, generically computing a generic-ring delay function in a number of steps which is less than its sequentiality depth is equivalent to factoring the modulus N. In particular, they showed that this implies that within the generic ring model, computing the repeated squaring function [43] in $\mathbb{Z}_N$ with respect to delay parameter T in less than T sequential steps is equivalent to factoring. Their results are incomparable to ours for several reasons. First, the generic ring model is incomparable to the strong AGM: On the one hand, the generic ring model withholds the elements' representation from the adversary (which the strong AGM does not do); but on the other hand, it allows the adversary to apply all of the ring operation onto pairs of ring elements, whereas the strong AGM requires that the adversary explains how its output was computed solely using the group operation. Secondly, Rotem and Segev only consider RSA groups (within the ring $\mathbb{Z}_N$), whereas Theorem 1 may be applied in any group. Finally, when considering the specific case of RSA groups, our result (Theorem 2) is restricted to RSA groups with respect to a modulus N which is the product of two safe primes, whereas the result of Rotem and Segev is not.

4 Overview of Our Contributions

In this section we provide an informal overview of the main technical ideas underlying our contributions. We start by presenting the techniques we use to derive Theorems 1 and 2 in hidden-order groups, and then move on to describe our techniques in bilinear groups for deriving Theorem 3. For brevity, we do not discuss here how we extend Theorem 3 to the decisional setting to obtain Theorem 4. Due to space limitations, the reader is referred to the full version for further details and formal theorem statements and proofs.

4.1 Our Reductions in Hidden-Order Groups

For simplicity of presentation in this informal overview, when presenting our reduction from the factoring problem to the fine-grained Uber problem in RSA group (Theorem 2), and our reduction from the RSA problem to the fine-grained Uber problem in general groups (Theorem 1), we restrict our attention to the problem of speeding-up the repeated squaring function of Rivest, Shamir and Wagner [43]. The reader is referred to the full version for a formal definition of the fine-grained Uber problem, and for our theorem statements and reductions in their full generality (applying to all problems within the fine-grained Uber family, and not just to speeding-up repeated squaring).

The strong AGM

We prove our results for the fine-grained Uber problem in hidden-order groups in the strong algebraic group model (the SAGM) put forth by Katz, Loss and Xu [37]. Informally speaking, the SAGM strengthens the AGM, by requiring that whenever a strongly-algebraic algorithm A outputs a group element y, it outputs alongside it not only a representation of it in the basis of the input group elements, but also the entire sequence of group operations used to derive this representation. An important feature of this model, is that the length of this sequence is dominated by the running time of the algorithm. Hence, if we view the input elements to a strongly-algebraic algorithm A as polynomials of degree at most d in some underlying indeterminates, then the output y can be viewed as a polynomial of degree at most d^{2^t} in these indeterminates, where t is the running time of A. This is the case since each group operation at most doubles the degree of the highest degree polynomial in the computation up to it. Note that this observation remains true even if A runs in time t on many processors that may perform group operations in parallel. See the full version for a formal definition of the model.

The reduction of Katz, Loss and Xu in QR_N

Recall the reduction of Katz et al. from the factoring problem to the problem of speeding-up the repeated squaring. They focused on the group QR_N of quadratic residues modulo N, for a modulus N which is the product of two safe primes; that is $N = (2p+1) \cdot (2q+1)$, where $p, q, 2p+1$ and $2q+1$ are all primes. Let $T \in \mathbb{N}$ and consider a strongly-algebraic algorithm A, that given a uniformly random group element $x \leftarrow QR_N$ computes x^{2^T} in time $t < T$. The reduction samples such an element x and invokes A on it. Since A is strongly-algebraic, it produces alongside its output y an integer $\alpha \leq 2^t$ such that $y = x^{\alpha}$. Whenever A succeeds in computing x^{2^T}, it means that $x^{\alpha} = x^{2^T}$, or equivalently, $x^{2^T - \alpha} = 1$ (all equalities are in the group QR_N). Since $\alpha \leq 2^t < 2^T$, this implies that $\omega = 2^T - \alpha$ is a non-zero multiple of the order of x in QR_N. The analysis of Katz et al. proceeds by observing that when N is the product of two safe primes, then almost all elements in the group are generators, and that the order of the group is $\varphi(N)/4 = p \cdot q$, where $\varphi(\cdot)$ is Euler's totient function. Thus, if A succeeds, then 4ω is almost surely a multiple of $\varphi(N)$, and the reduction is completed by invoking a well-known algorithm for factoring N given a multiple of $\varphi(N)$ (e.g., [36, Theorem 8.50]).

Our reduction in RSA groups (Theorem 2)

Unlike in the group QR_N, when moving to consider the RSA group $\mathbb{Z}_N^*$, the group is no longer cyclic and hence a random element in the group is never a generator. However, our generalization of the result of Katz et al. to the RSA group $\mathbb{Z}_N^*$ is based on the observation that their reduction actually does not rely on the fact that the sampled x is a generator of QR_N. Instead, it only uses the fact that with overwhelming probability over the choice of x, its order satisfies a certain relation with $\varphi(N)$. We use this observation to prove that the reduction of Katz et al. can be applied not only in QR_N when N is the product of two safe primes, but also in $\mathbb{Z}_N^*$ when N is of this form. Concretely, denoting $N = (2p+1) \cdot (2q+1)$, we use the isomorphism of $\mathbb{Z}_N^*$ to the product group $\mathbb{Z}_2^2 \times \mathbb{Z}_p \times \mathbb{Z}_q$ in order to argue that almost all elements in $\mathbb{Z}_N^*$ have order either $p \cdot q = \varphi(N)/4$ or order $2p \cdot q = \varphi(N)/2$. Therefore, it is still the case that that whenever A succeeds in computing x^{2^T}, it must be that $4\omega = 4(2^T - \alpha)$ is a multiple of $\varphi(N)$ and the correctness of the reduction follows.

Our reduction in general groups (Theorem 1)

Let $\mathbb{G}$ be an abelian group. The goal of our reduction is to solve the RSA problem in $\mathbb{G}$, where the problem is parameterized by an integer e which is coprime to the order of $\mathbb{G}$. That is, given a uniformly random $u \in \mathbb{G}$, we need to find a group element $w \in \mathbb{G}$ such that $w^e = u$ (meaning, w is the eth root of u). Let $T \in \mathbb{N}$ and consider a strongly-algebraic algorithm A, that given a uniformly random group element $x \leftarrow \mathbb{G}$ computes x^{2^T} in time $t < T$. Our reduction starts by invoking A on input u (the input to the RSA problem). As before, since A is strongly-algebraic, it produces alongside its output y an integer $\alpha \leq 2^t$ such that $y = u^{\alpha}$. Following a similar argument as in the previous reduction, if y is indeed equal to u^{2^T}, then $\omega = 2^T - \alpha$ is a non-zero multiple of the order of u in $\mathbb{G}$. The new idea underlying our reduction from the RSA problem, is that we can use this information about the order of u to find its eth root. Suppose that we could find an inverse d of e modulo ω; i.e., an integer d satisfying $ed = n \cdot \omega + 1$ for some integer n. Then, we would be done, since u^d would be the eth root of u:

$$\left(u^d\right)^e = u^{ed} = u^{n \cdot \omega + 1} = (u^{\omega})^n \cdot u = 1 \cdot u = u, \qquad (1)$$

where $u^{\omega} = 1$ because ω is a multiple of u's order. The problem is that e might not have an inverse modulo ω, as the two integers might not be relatively prime. To remedy this situation, we factor out from ω any common divisors that it shares with e, by computing $\omega' = \omega/\mathsf{gcd}(\omega, e)$. On the one hand, we are now guaranteed that ω' and e are coprime, and we can find an inverse d' of e modulo ω'. On the other hand, the key observation is that ω' must still be a multiple of u's order in $\mathbb{G}$. This is because e is coprime to the order of $\mathbb{G}$ and thus, by Lagrange's theorem, it is also coprime to the order of u. This means that if we write $\omega = \mathsf{order}(u) \cdot c$ for some integer c, then

$$\omega' = \frac{\omega}{\mathsf{gcd}(\omega, e)} = \frac{\mathsf{order}(u) \cdot c}{\mathsf{gcd}(\mathsf{order}(u) \cdot c, e)} = \mathsf{order}(u) \cdot \frac{c}{\mathsf{gcd}(c, e)}$$

and ω' is indeed a multiple of $\mathsf{order}(u)$. Hence, our reduction may simply output $u^{d'}$, and the same analysis from Eq. (1) still applies, replacing ω and d with ω' and d' respectively.

4.2 Our Reduction in Bilinear Groups

For simplicity of presentation, we focus here on the case of symmetric bilinear groups. In this setting, we consider a single source group $\mathbb{G}$ of prime order $p \in \mathbb{N}$ that is equipped with a bilinear map e, mapping pairs of elements in $\mathbb{G}^2$ to elements in a target group $\mathbb{G}_T$. We also restrict our attention to a simple problem within the Uber family, referred to below as the (f, h)-Uber problem, as the reduction in this case already captures the gist of the techniques used in our proof of Theorem 3. In the (f, h)-Uber problem the adversary is given g and $g^{f(\vec{x})}$ and is required to compute $g_T^{h(\vec{x})}$, where g is a generator of $\mathbb{G}$, $g_T = e(g, g)$ is a generator of $\mathbb{G}_T$, f and h are polynomials parameterizing the problem, and $\vec{x} = (x_1, \ldots, x_m)$ is a m-tuple of elements in $\mathbb{Z}_p$ chosen independently and uniformly at random. We assume that there are no integers $\alpha, \beta, \gamma \in \mathbb{Z}_p$ such that $h = \alpha + \beta \cdot f + \gamma \cdot f^2$ over $\mathbb{Z}_p$, as otherwise the problem is trivial to solve and we cannot hope to reduce the q-DLOG problem (or any other problem which we believe to be hard) to it.[8] We start by recalling the reduction of Bauer et al. [7] and then move to describe our reduction and how it improves upon it.

The reduction of Bauer, Fucshbauer and Loss

Let A be an algebraic algorithm for the (f, h)-Uber problem. The reduction receives as input $g, g^x, \ldots, g^{x^q}$ for a uniformly sampled $x \leftarrow \mathbb{Z}_p$, and its goal is to find x. The idea of Bauer et al. was to have the reduction "plant" m independent randomized versions of the secret exponent x as the m secret exponents in a random instance of the (f, h)-Uber problem and then invoke A on this instance. This is done by sampling $\alpha_i, \beta_i \leftarrow \mathbb{Z}_p$ for each $i \in [m]$ and invoking A on input $(g, g^{f(\vec{x'})})$ where $\vec{x'} = (\alpha_1 \cdot x + \beta_1, \ldots, \alpha_m \cdot x + \beta_m)$. Since A is an algebraic algorithm, it provides alongside its output $y \in \mathbb{G}_T$ three integers $\alpha, \beta, \gamma \in \mathbb{Z}_p$ such that $y = e(g, g)^{\alpha} \cdot e(g, g^{f(\vec{x'})})^{\beta} \cdot e(g^{f(\vec{x'})}, g^{f(\vec{x'})})^{\gamma}$. Whenever A succeeds, it means that $y = g_T^{h(\vec{x'})}$, and hence, since g_T is a generator of $\mathbb{G}_T$, we obtain that

$$h(\vec{x'}) = \alpha + \beta \cdot f(\vec{x'}) + \gamma \cdot \left(f(\vec{x'})\right)^2 . \tag{2}$$

[8] If such integers existed, the adversary could simply compute and output $e(g, g)^{\alpha} \cdot e(g, g^{f(\vec{x})})^{\beta} \cdot e(g^{f(\vec{x})}, g^{f(\vec{x})})^{\gamma}$.

Observe that Eq. (2) is a univariate equation over the finite field $\mathbb{F}_p$. Roughly speaking, the non-triviality of the (f,h)-Uber problem guarantees that with overwhelming probability, Eq. (2) is not the trivial equation. Therefore, the reduction simply uses any efficient polynomial factorization algorithm (e.g., Berlekamp's algorithm [9, 42]) to find all solutions to Eq. (2) and then checks each of them to see if it is the secret exponent x. Note that in order to compute $g^{f(\vec{x'})}$ given as input to A, the reduction needs access to $g, g^x, \dots, g^{x^q}$ for q that is the *total* degree of f.

Our improved reduction (Theorem 3)

In order to reduce the parameter q needed by the reduction, our idea is to plant the secret exponent x in place of just one of the exponents $x_1, \dots, x_m$ in a random instance of the (f,h)-Uber problem, and sample the rest of the exponents uniformly at random. Concretely, our reduction samples a random index $i^* \leftarrow [m]$ and plants x instead of x_{i^*}: It samples $m-1$ additional values $x_1, \dots, x_{i^*-1}, x_{i^*+1}, \dots, x_m \leftarrow \mathbb{Z}_p$ and invokes A on input $(g, g^{f(\vec{x''})})$ where $\vec{x''} = (x_1, \dots, x_{i^*-1}, x, x_{i^*+1}, \dots, x_m)$. Observe that indeed, in order to compute the group element $g^{f(\vec{x''})}$ given as input to A, all the reduction needs is access to $g, g^x, \dots, g^{x^q}$ for a parameter q which is at least the degree of x_{i^*} in f. In particular, the reduction can be efficiently implemented as long as q is at least the maximal degree in which some variable appears in f. As before, since A is algebraic, it outputs alongside its output $y \in \mathbb{G}_T$ three integers $\alpha, \beta, \gamma \in \mathbb{Z}_p$ such that $y = e(g,g)^\alpha \cdot e(g, g^{f(\vec{x''})})^\beta \cdot e(g^{f(\vec{x''})}, g^{f(\vec{x''})})^\gamma$, and whenever $y = g_T^{h(\vec{x''})}$, it holds that

$$h(\vec{x''}) = \alpha + \beta \cdot f(\vec{x''}) + \gamma \cdot \left(f(\vec{x''})\right)^2. \tag{3}$$

Alas, we can no longer simply solve Eq. (3) for x. The problem is that for our choice of $x_1, \dots, x_{i^*-1}, x_{i^*+1}, \dots, x_m$, Eq. (3) might be a trivial equation, yielding no information about x. Worse still, this situation may arise with high probability over the choice of i^* and of $x_1, \dots, x_{i^*-1}, x_{i^*+1}, \dots, x_m$. So instead, our reduction uses the following observation by Rotem and Segev [44], which in turn generalizes the previous work of Fuchsbauer, Kiltz and Loss [28]. For a non-zero polynomial $\ell(X_1, \dots, X_m)$ in the indeterminates $X_1, \dots, X_m$, we define a cascade of multivariate polynomials recursively: We set $\ell_1 = \ell$; and for every $i \in \{2, \dots, m\}$, we let $\ell_i(X_i, \dots, X_m)$ be the first non-zero coefficient of $\ell_{i-1}(X_{i-1}, \dots, X_m)$, when ℓ_{i-1} is written as a univariate polynomial in the indeterminate X_{i-1} with coefficients which are polynomials in $X_i, \dots, X_m$. Rotem and Segev proved that for every vector $\vec{z} = (z_1, \dots, z_m) \in \mathbb{Z}_p^m$ such that $\ell(\vec{z}) = 0$, there exists $t^* \in [m]$ such that: The univariate polynomial $v(X) = \ell_{t^*}(X, z_{t^*+1}, \dots, z_m)$ is not the zero polynomial and additionally $v(z_{t^*}) = 0$.[9] Using this observation, our reduction considers the m-variate polynomial $\ell(\vec{X}) = \alpha + \beta \cdot f(\vec{X}) + \gamma \cdot \left(f(\vec{X})\right)^2 - h(X)$, and computes the polynomial $\ell_{i^*}(X_{i^*+1}, \dots, X_m)$ from it as described by the above recursive procedure, where i^* is the index sampled by the reduction when preparing the (f,h)-Uber instance to A. It then factors the univariate polynomial $v(X) = \ell_{i^*}(X, x_{i^*+1}, \dots, x_m)$ to find all of its roots, where $x_{i^*+1}, \dots, x_m$ are the random $\mathbb{Z}_p$ values chosen by the reduction for generating the (f,h)-Uber instance. As for the success probability of the reduction, whenever A succeeds in computing the $g_T^{h(\vec{x''})}$, it holds that $\vec{x''}$

[9] Fuchsbauer, Kiltz and Loss [28] essentially observed that this holds for bi-linear polynomials to reduce the hardness of the Computational Diffie-Hellman (CDH) problem to that of the discrete log problem within the AGM.

is a root of the m-variate polynomial ℓ. By the observation of Rotem and Segev, this means that there is an index $t^* \in [m]$ such that if the index i^* guessed by the reduction matches it, then $v(X) = \ell_{i^*}(X, x_{i^*+1}, \ldots, x_m)$ is not the zero polynomial, and x is a root of it. Hence, if A succeeds and $i^* = t^*$, then the reduction will successfully retrieve the secret exponent x of the q-DLOG problem.

References

1 Michel Abdalla, Manuel Barbosa, Tatiana Bradley, Stanislaw Jarecki, Jonathan Katz, and Jiayu Xu. Universally composable relaxed password authenticated key exchange. In *Advances in Cryptology – CRYPTO '20*, pages 278–307, 2020.

2 Michel Abdalla, Manuel Barbosa, Jonathan Katz, Julian Loss, and Jiayu Xu. Algebraic adversaries in the universal composability framework. In *Advances in Cryptology – ASIACRYPT '21*, pages 311–341, 2021.

3 Michel Abdalla, Fabrice Benhamouda, and Philip MacKenzie. Security of the J-PAKE password-authenticated key exchange protocol. In *IEEE Symposium on Security and Privacy*, pages 571–587, 2015.

4 Divesh Aggarwal and Ueli Maurer. Breaking RSA generically is equivalent to factoring. In *Advances in Cryptology – EUROCRYPT '09*, pages 36–53, 2009.

5 Thomas Agrikola, Dennis Hofheinz, and Julia Kastner. On instantiating the algebraic group model from falsifiable assumptions. In *Advances in Cryptology – EUROCRYPT '20*, pages 96–126, 2020.

6 Benedikt Auerbach, Federico Giacon, and Eike Kiltz. Everybody's a target: Scalability in public-key encryption. In *Advances in Cryptology – EUROCRYPT '20*, pages 475–506, 2020.

7 Balthazar Bauer, Georg Fuchsbauer, and Julian Loss. A classification of computational assumptions in the algebraic group model. In *Advances in Cryptology – CRYPTO '20*, pages 121–151, 2020.

8 Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the 1st ACM Conference on Computer and Communications Security*, pages 62–73, 1993.

9 Elwyn Ralph Berlekamp. Factoring polynomials over large finite fields. *Mathematics of Computation*, 24(111):713–735, 1970.

10 David Bernhard, Marc Fischlin, and Bogdan Warinschi. On the hardness of proving CCA-security of signed ElGamal. In *Public-Key Cryptography – PKC ,16*, pages 47–69, 2016.

11 LenIngrid Biehl, Johannes Buchmann, Safuat Hamdy, and Andreas Meyer. A signature scheme based on the intractability of computing roots. *Designs, Codes and Cryptography*, 25(3):223–236, 2002.

12 Lenore Blum, Manuel Blum, and Michael Shub. A simple unpredictable pseudo-random number generator. *SIAM Journal on Computing*, 15(2):364–383, 1986.

13 Dan Boneh, Joseph Bonneau, Benedikt Bünz, and Ben Fisch. Verifiable delay functions. In *Advances in Cryptology – CRYPTO '18*, pages 757–788, 2018.

14 Dan Boneh, Xavier Boyen, and Eu-Jin Goh. Hierarchical identity based encryption with constant size ciphertext. In *Advances in Cryptology – EUROCRYPT '05*, pages 440–456, 2005.

15 Dan Boneh, Benedikt Bünz, and Ben Fisch. A survey of two verifiable delay functions. Cryptology ePrint Archive, Report 2018/712, 2018.

16 Dan Boneh, Justin Drake, Ben Fisch, and Ariel Gabizon. Efficient polynomial commitment schemes for multiple points and polynomials. Cryptology ePrint Archive, Report 2020/081, 2020.

17 Dan Boneh and Moni Naor. Timed commitments. In *Advances in Cryptology – CRYPTO '00*, pages 236–254, 2000.

18 Dan Boneh and Ramarathnam Venkatesan. Breaking RSA may not be equivalent to factoring. In *Advances in Cryptology – EUROCRYPT '98*, pages 59–71, 1998.

19 Xavier Boyen. The Uber-assumption family – A unified complexity framework for bilinear groups. In *Proceedings of the 2nd International Conference on Pairing-based Cryptography*, pages 39–56, 2008.

20 Emmanuel Bresson, Olivier Chevassut, and David Pointcheval. Provably authenticated group diffie-hellman key exchange – the dynamic case. In *Advances in Cryptology – ASIACRYPT '01*, pages 290–309, 2001.

21 Emmanuel Bresson, Olivier Chevassut, and David Pointcheval. Provably secure authenticated group diffie-hellman key exchange. *ACM Transactions on Information and System Security*, 10(3):10:1–10:45, 2007.

22 Emmanuel Bresson, Olivier Chevassut, David Pointcheval, and Jean-Jacques Quisquater. Provably authenticated group diffie-hellman key exchange. In *Proceedings of the 8th ACM Conference on Computer and Communications Security*, pages 255–264, 2001.

23 Johannes Buchmann and Safuat Hamdy. A survey on IQ cryptography. In *In Proceedings of Public Key Cryptography and Computational Number Theory*, pages 1–15, 2001.

24 Alessandro Chiesa, Yuncong Hu, Mary Maller, Pratyush Mishra, Noah Vesely, and Nicholas Ward. Marlin: Preprocessing zkSNARKs with universal and updatable SRS. In *Advances in Cryptology – EUROCRYPT '20*, pages 738–768, 2020.

25 Geoffroy Couteau and Dominik Hartmann. Shorter non-interactive zero-knowledge arguments and ZAPs for algebraic languages. In *Advances in Cryptology – CRYPTO '20*, pages 768–798, 2020.

26 Ivan Damgård and Maciej Koprowski. Generic lower bounds for root extraction and signature schemes in general groups. In *Advances in Cryptology – EUROCRYPT '02*, pages 256–271, 2002.

27 Alexander W. Dent. Adapting the weaknesses of the random oracle model to the generic group model. In *Advances in Cryptology – ASIACRYPT '02*, pages 100–109, 2002.

28 Georg Fuchsbauer, Eike Kiltz, and Julian Loss. The algebraic group model and its applications. In *Advances in Cryptology – CRYPTO '18*, pages 33–62, 2018.

29 Georg Fuchsbauer, Antoine Plouviez, and Yannick Seurin. Blind Schnorr signatures and signed ElGamal encryption in the algebraic group model. In *Advances in Cryptology – EUROCRYPT '20*, pages 63–95, 2020.

30 Juan A. Garay and Markus Jakobsson. Timed release of standard digital signatures. In *Financial Cryptography*, pages 190–207, 2002.

31 Juan A. Garay, Philip D. MacKenzie, Manoj Prabhakaran, and Ke Yang. Resource fairness and composability of cryptographic protocols. In *Proceedings of the 4th Theory of Cryptography Conference*, pages 404–428, 2006.

32 Juan A. Garay and Carl Pomerance. Timed fair exchange of standard signatures. In *Financial Cryptography*, pages 190–207, 2003.

33 Ashrujit Ghoshal and Stefano Tessaro. Tight state-restoration soundness in the algebraic group model. In *Advances in Cryptology – CRYPTO '21*, pages 64–93, 2021.

34 Sergey Gorbunov, Leonid Reyzin, Hoeteck Wee, and Zhenfei Zhang. Pointproofs: Aggregating proofs for multiple vector commitments. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, pages 2007–2023, 2020.

35 Julia Kastner, Julian Loss, and Jiayu Xu. On pairing-free blind signature schemes in the algebraic group model. To appear in *Public-Key Cryptography — PKC '22* (available at `https://eprint.iacr.org/2020/1071.pdf`), 2022.

36 Jonathan Katz and Yehuda Lindell. *Introduction to modern cryptography (2nd edition)*. Chapman & Hall/CRC press, 2014.

37 Jonathan Katz, Julian Loss, and Jiayu Xu. On the security of time-lock puzzles and timed commitments. In *Proceedings of the 18th Theory of Cryptography Conference*, pages 390–413, 2020.

38 Mary Maller, Sean Bowe, Markulf Kohlweiss, and Sarah Meiklejohn. Sonic: Zero-knowledge SNARKs from linear-size universal and updatable structured reference strings. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 2111–2128, 2019.

39 Taiga Mizuide, Atsushi Takayasu, and Tsuyoshi Takagi. Tight reductions for Diffie-Hellman variants in the algebraic group model. In *Topics in Cryptology – CT-RSA '19*, pages 169–188, 2019.

40 Pascal Paillier and Damien Vergnaud. Discrete-log-based signatures may not be equivalent to discrete log. In *Advances in Cryptology – ASIACRYPT '05*, pages 1–20, 2005.

41 Krzysztof Pietrzak. Simple verifiable delay functions. In *Proceedings of the 10th Conference on Innovations in Theoretical Computer Science*, pages 60:1–60:15, 2019.

42 Michael Oser Rabin. Probabilistic algorithms in finite fields. *SIAM Journal on Computing*, 9(2):273–280, 1980.

43 R. L. Rivest, A. Shamir, and D. A. Wagner. Time-lock puzzles and timed-release crypto, 1996.

44 Lior Rotem and Gil Segev. Algebraic distinguishers: From discrete logarithms to decisional uber assumptions. In *Proceedings of the 18th Theory of Cryptography Conference*, pages 366–389, 2020.

45 Lior Rotem and Gil Segev. Generically speeding-up repeated squaring is equivalent to factoring: Sharp thresholds for all generic-ring delay functions. In *Advances in Cryptology – CRYPTO '20*, pages 481–509, 2020.

46 Aron van Baarsen and Marc Stevens. On time-lock cryptographic assumptions in abelian hidden-order groups. In *Advances in Cryptology – ASIACRYPT '21*, pages 367–397, 2021.

47 Benjamin Wesolowski. Efficient verifiable delay functions. In *Advances in Cryptology – EUROCRYPT '19*, pages 379–407, 2019.