The Distinction Between Fixed and Random Generators in Group-Based Assumptions[*]

James Bartusek[1], Fermi Ma[1], and Mark Zhandry[2]

[1] Princeton University
{bartusek.james,fermima1}@gmail.com
[2] Princeton University & NTT Research
mzhandry@princeton.edu

Abstract. There is surprisingly little consensus on the precise role of the generator g in group-based assumptions such as DDH. Some works consider g to be a fixed part of the group description, while others take it to be random. We study this subtle distinction from a number of angles.

- In the generic group model, we demonstrate the plausibility of groups in which random-generator DDH (resp. CDH) is hard but fixed-generator DDH (resp. CDH) is easy. We observe that such groups have interesting cryptographic applications.
- We find that seemingly tight generic lower bounds for the Discrete-Log and CDH problems with preprocessing (Corrigan-Gibbs and Kogan, Eurocrypt 2018) are not tight in the sub-constant success probability regime if the generator is random. We resolve this by proving tight lower bounds for the random generator variants; our results formalize the intuition that using a random generator will reduce the effectiveness of preprocessing attacks.
- We observe that DDH-like assumptions in which exponents are drawn from low-entropy distributions are particularly sensitive to the fixed- vs. random-generator distinction. Most notably, we discover that the Strong Power DDH assumption of Komargodski and Yogev (Komargodski and Yogev, Eurocrypt 2018) used for non-malleable point obfuscation is in fact *false* precisely because it requires a fixed generator. In response, we formulate an alternative fixed-generator assumption that suffices for a new construction of non-malleable point obfuscation, and we prove the assumption holds in the generic group model. We also give a generic group proof for the security of fixed-generator, low-entropy DDH (Canetti, Crypto 1997).

1 Introduction

Starting with the seminal work of Diffie and Hellman [21], the *Computational Diffie-Hellman* (CDH) assumption in certain cyclic groups has become a core pillar of modern cryptography. For a finite cyclic group G and generator g, the assumption holds if it is hard to compute g^{ab} given (g, g^a, g^b) for random a, b.

[*] The full version of this paper is available at ia.cr/2019/202 [3].

The corresponding *Decisional Diffie-Hellman* (DDH) assumption, introduced by Brands [12], is that given (g, g^a, g^b) for random a, b, it is hard to distinguish g^{ab} from g^c for random c.

A somewhat subtle issue is the precise role of g in these assumptions: is it fixed in the group description, or is it randomly chosen along with a and b? For CDH in groups where the totient of the order is known, a folklore equivalence between the fixed and random generator variants exists (e.g. see Chapter 21 of Galbraith's textbook [25]). For DDH, Shoup [40] observed that the fixed generator assumption appears to be a *stronger* assumption than the random generator version, though a formal separation between the two is unknown. Despite this apparent distinction, the cryptographic literature commonly refers to both the fixed and random generator variants simply as "DDH".[3]

A likely explanation for this practice is that in most applications of cryptographic groups, it is straightforward to switch between fixed and random generators. For example, in ElGamal encryption [22], users who want the additional security of random-generator DDH can easily specify a random generator in their public key.

Sadeghi and Steiner [37] observed that this justification does not apply in settings where the choice of group generator is left to a potentially untrusted party.[4] They give the example of a bank that offers its customers an anonymous payment system, claiming provable security under group-based assumptions. If the bank is free to choose parameters such as the group generator, then for security it is crucial that any underlying assumptions hold in their (stronger) fixed generator form. While Sadeghi and Steiner did not point to specific assumptions that can be broken simply by fixing the group generator, they stressed that continuing to conflate these distinct assumptions could lead to serious ambiguities and mistakes in the future.

In the nearly two decades since Sadeghi and Steiner [37] first called attention to the above issue, dozens of new and increasingly sophisticated group-based assumptions have been introduced. Accordingly, researchers have devoted significant effort to evaluating the plausibility of these assumptions (e.g. [2, 20]), frequently in idealized models such as the generic group model [36, 39, 34]. We observe that these generic group justifications generally ignore the question of whether the generator is fixed or random, but that in most cases this distinction does not seem to affect real world security of these assumptions.

In this work, however, we will see that this is not *always* the case.

[3] For example, the Katz-Lindell textbook [29] defines DDH with a fixed generator, while Cramer-Shoup [19] defines DDH with a random generator.

[4] Sadeghi and Steiner [37] actually consider the more general possibility of the untrusted party choosing the group itself maliciously. This question is beyond the scope of our work, but in many cases it is an equally important consideration.

1.1 Our Results

We first examine how the fixed vs. random generator distinction affects the classical Discrete-Log, CDH, and DDH problems in a variety of different settings, obtaining the following results:

- **Generic Separations for CDH and DDH.** We prove that fixed- and random-generator DDH are *inequivalent* assumptions in the generic group model [36, 39, 34]. We show that for groups of *unknown order*, fixed- and random-generator CDH are also inequivalent assumptions in the generic group model. In addition, we give evidence (relying on a new assumption about arithmetic circuits) that they are inequivalent even if the group order is known but its factorization is not.[5]
- **Split-CDH and Split-DDH Groups.** We define Split-CDH (resp. Split-DDH) groups for which the fixed-generator variant of CDH (resp. DDH) is easy but the random-generator variant is hard, and we observe that such groups imply interesting cryptographic applications. A split-CDH group can be turned into a *self-bilinear map* [43, 30] where the random-generator variant of the Multilinear CDH assumption holds. This implies powerful primitives such as multiparty non-interactive key agreement (with trusted setup).[6] A split-DDH group can be used to instantiate a variant of the Boneh-Franklin identity-based encryption [8] scheme. We stress here that giving candidate constructions of these groups is outside of the scope of this work. On the negative side, we prove that a natural class of non-interactive key exchange protocols (without trusted setup) are *insecure* in certain split-CDH groups.
- **Asymptotic Bounds for Discrete-Log and CDH with Preprocessing.** We revisit the recent work of Corrigan-Gibbs and Kogan [18], which seemingly resolves the generic hardness of Discrete-Log and CDH with preprocessing. We observe that while their lower bounds are tight for the fixed-generator variants, they leave a gap in the random-generator setting for algorithms with sub-constant success probability. We close these gaps by proving tight lower bounds for the random-generator variants. Our bounds suggest that using a random generator can reduce the impact of preprocessing attacks, and in turn group parameters can be set more aggressively than previously thought in situations where random-generator Discrete-Log or CDH are sufficient.

Next, we turn our attention to the class of Diffie-Hellman-like assumptions involving *non-uniform random exponents*. An example of such an assumption is Canetti's "DDH-II" assumption [13], which states that DDH remains hard even if the exponent a in (g, g^a, g^b, g^{ab}) is drawn from a well-spread distribution (so that a has super-logarithmic min-entropy). While these assumptions are somewhat undesirable due to their non-standard nature [27], Wee [42] showed that these

[5] This inequivalence was also suggested by Saxena and Soh [38].

[6] A similar observation was also made in [38].

assumptions (ones that require hardness given only super-logarithmic entropy) are *necessary* for applications such as point-function obfuscation.

Before we rely on such assumptions, it is important to rule out idealized adversaries that attack the underlying structure of the assumption. The most common technique for achieving this is to prove the assumption holds in the generic group model [36, 39, 34]. Such proofs certainly do not imply the validity of the assumption; instead, these proofs are generally viewed as a *minimal level of guarantee* we need to gain confidence in an assumption [2].

Our central focus is on the recently proposed "Strong Power DDH" assumption of Komargodski and Yogev [31]. The assumption states that for x sampled from any arbitrary well-spread distribution $\mathcal{D}$, that $g^x, g^{x^2}, \ldots, g^{x^k}$ is indistinguishable from k uniformly random group elements. Our results are the following:

– **Strong Power DDH is False for a Fixed Generator.** We demonstrate the "Strong Power DDH" assumption underlying Komargodski and Yogev's non-malleable point obfuscator [31] as well as Fenteany and Fuller's non-malleable digital locker [23] is *false* in the fixed-generator setting. [7]This results from a subtle issue in the order of quantifiers; if g is fixed, an arbitrary well-spread distribution could depend on g. For example, x can come from the distribution that conditions on the bit-representation of g^x beginning with 0. Unfortunately, these constructions can only be instantiated with a fixed generator, so the original security proofs in [31] and [23] must rely on a false assumption.[8,9]

  In response to private communication from the authors of this work, Komargodski and Yogev have offered a simple fix [32] for their original construction through a new "Entropic Power DDH" assumption.[10] This new assumption suffices for non-malleable point obfuscation and is formulated precisely to address the vulnerability described above.

– **Fixing Non-Malleable Point Obfuscation and Justifying Assumptions in the Generic Group Model.** In this work, we offer an alternative resolution. We construct a new non-malleable point obfuscator that is qualitatively different from the one in [31]. Security of our construction relies on a newly formulated fixed-generator entropic assumption that we prove holds in the generic group model. Note that neither the Strong Power DDH Assumption [31] nor the revised Entropic Power DDH Assumption [32] come with generic group proofs of security.

[7] The authors privately communicated these issues to the authors of [31, 23].

[8] Relying on a random generator would require a common random string, which is not the model considered in [31] or in the version of [23] dated Jan 30, 2019 at eprint.iacr.org/2018/957/20190130:190441.

[9] This issue appears in the Eurocrypt 2018 version of [31], in an older ePrint version of [32] dated May 1, 2018 at eprint.iacr.org/2018/149/20180211:142746, and in the ePrint version of [23] dated Jan 30, 2019 at eprint.iacr.org/2018/957/20190130:190441.

[10] This refers to the newer ePrint version of [32] dated Feb 21, 2019 available at https://eprint.iacr.org/2018/149/20190221:133556.

Along the way, we develop general techniques (based heavily on [16]) for proving generic security of non-standard, entropic assumptions. As a final contribution, we demonstrate the applicability of these techniques by showing that the fixed- and random-generator versions of Canetti's DDH-II assumption [13] hold in the generic group model.[11] This assumption has been used in both its fixed-generator form (e.g. [28, 14, 20]) and random-generator form (e.g. [13, 6]).

1.2 Technical Overview

Part 1: Generic Separations and Split Groups.

Formalizing the Distinction. We will assume some process for generating a group description G of order N. This group description is assumed to include a generator g. The *fixed-generator* DDH assumption, or f-DDH, states that the tuples (g^x, g^y, g^{xy}) and (g^x, g^y, g^z) are computationally indistinguishable, given the description of G. Here, x, y, z are chosen randomly in $\mathbb{Z}_N$. On the other hand, the *random-generator* DDH assumption, or r-DDH, states that the tuples (h, h^x, h^y, h^{xy}) and (h, h^x, h^y, h^z) are computationally indistinguishable. Here, x, y, z are chosen randomly in $\mathbb{Z}_N$, and h is a random generator of G (chosen, say, by setting $h = g^r$ for a random r in $\mathbb{Z}_N^*$). We can also define fixed- and random-generator variants of Computational Diffie-Hellman (CDH) and Discrete-Log (DLog). For example, f-CDH states that given (g^x, g^y) for random x, y, it is computationally infeasible to find g^{xy}.

We consider the following three settings of groups: known prime group order, known composite group order of *unknown* factorization, and unknown group order. For each of the three assumptions and three settings (for 9 instances in total) we explore the relationship between the fixed- and random-generator variants. Trivially, the f- variants of the assumptions are at least as strong as the r- variants. In the other direction, some instances have known or folklore reductions showing equivalence [25]. For each of the cases that do not have a proof of equivalence, we provide a separation. This is formalized by augmenting the generic group model [39] with an oracle for the f- variant, and showing (potentially under reasonable computational assumptions) that the r- variant still holds. Table 1 summarizes our findings.

Applications of Split Groups. Looking at Table 1, we see that in the case of DDH, there is the potential for a group where f-DDH is easy but r-DDH is hard. We will call such groups split-DDH groups. Similarly, if the group order is unknown, potentially f-CDH is easy but r-CDH is hard; we call such groups split-CDH

[11] Previously, such proofs had been obtained by Bitanksy and Canetti [6] and Damgård, Hazay, and Zottarel [20], who considered the random- and fixed-generator versions, respectively. We observe that both of these proofs treat the well-spread distribution as independent of the generic group labeling. Our proof handles distributions with arbitrary dependence on the labels; for more discussion refer to Part 4 of Section 1.2.

	DLog	CDH	DDH
Known Order	✓ FL	✓ FL	$\times$
Unknown Factorization	✓ FL	$\times$?	$\times$
Unknown order	✓ FL	$\times$? [44]	$\times$

Table 1. Generic equivalences and separations. FL denotes a folklore result. ✓ means that the fixed and random generator versions are equivalent. $\times$ means that the random generator version is harder than the fixed generator version (in the generic model). $\times$? means the result holds under a plausible conjecture. These results are all given in the full version [3].

groups. In this section, we will see that such split Diffie-Hellman groups have useful cryptographic applications.

First, we observe that a split-CDH group is very close to a self-bilinear map [43]. A self-bilinear map is a group G together with a pairing $e : G^2 \to G$ such that $e(g^x, g^y) = e(g, g)^{xy}$. Let $g_1 = g$ and $g_n = e(g, g_{n-1})$. A typical computational assumption on self-bilinear maps would be the multilinear CDH assumption [9]: for any $n > 1$, given $g^{x_0}, \ldots, g^{x_n}$, it is hard to compute $g_n^{\prod_{i=0}^n x_i}$. Notice that by applying the mapping $e(\cdot, \cdot)$, it is only possible to compute $g_{n+1}^{\prod_{i=0}^n x_i}$.

An f-CDH oracle gives such an oracle where $e(g, g) = g$. Therefore, a split-CDH group gives all the functionality of a self-bilinear map. But notice that since $e(g, g) = g$, $g_n = g$ for any n. Therefore, the multilinear CDH assumption is false. However, we observe that if we choose a random element h, then $e(h, h) = h^r$ where $h = g^r$. As such, the f-CDH oracle would also give a self-bilinear map with respect to the random generator h. We then show that multilinear CDH is actually hard relative to h, assuming r-CDH is hard. Thus, we obtain a self-bilinear map from any split-CDH group. As a consequence, following [43] we would immediately obtain multiparty non-interactive key agreement, broadcast encryption satisfying a distributed setup notion [10], and attribute-based encryption for circuits.

In the full version [3] we show that Split-DDH groups allow for a simple identity-based encryption (IBE) scheme based on the Boneh-Franklin [8] construction.

Part 2: Trusted Setup Assumptions The previous sections demonstrated that the f- and r-DDH assumptions are distinct assumptions that may not both be true. But then which DDH assumption should be used? In practice, g is typically part of a standards library chosen by a trusted third party (e.g. NIST). As such, users have essentially three choices:

1. Believe that the trusted third party chose g at random, and use the r-DDH assumption.

2. Do not trust the third party, but instead assume that there are no bad g. In other words, rely on the f-DDH assumption for g.
3. Do not trust the third party, but instead have one of the users generate a random g and distribute it to everyone else. Then rely on r-DDH.

Option 1 means that users need to trust that no one could have subverted g and chosen a bad generator for which DDH is actually easy; history has shown such trust could very well be misplaced. Only Options 2 and 3 remove the need to trust a third party.

Remark 1. Note that to remove trusted setup assumptions entirely, we would need to ensure that G itself is guaranteed to satisfy f-DDH. One option is to assume that both G and g were generated by a deterministic process, so that all parties can calculate G, g for themselves without any setup. For groups based on finite fields, this requires deterministically generating large primes; while no polynomial-time provable algorithms are known, there are very simple heuristic algorithms. For elliptic curve-based groups, other options are available (e.g. using a field with small characteristic). For one approach to deterministic curve generation, see [11].

In most cases, it is straightforward to switch between Options 2 and 3. A scheme designed for f-DDH can often be converted into a scheme that relies only on r-DDH by having one of the parties choose a random generator. On the other hand, a scheme designed for r-DDH can often be converted into an f-DDH scheme by fixing a group element and not including it with the user's messages, saving slightly on transmission costs.

The above means slightly different parameter sizes for the two assumptions. For example, for public key schemes, the extra group element would naturally go in the public key. The result is that schemes secure under r-DDH naturally require one additional group element in the public key relative to the f-DDH analog. As authors often compare parameter sizes in terms of group elements (e.g. [24]), it is important that they clearly identify which assumption is used.

In some cases, however, switching between f-DDH and r-DDH will have a more profound impact. For example, in a protocol between mutually distrusting parties, which party will be entrusted to come up with the generator? While we are not aware of any instances of protocols in the literature that cannot be made to work with a random generator, it is straightforward to devise protocols where no single party can be trusted to choose the generator. As such, care must be taken when using the r-DDH assumption in these settings.

Diffie-Hellman Key Exchange. For the remainder of this section, we will focus on a concrete setting where it is not possible to trivially switch between f-DDH and r-DDH: Diffie-Hellman key exchange. In the protocol, Alice chooses a random $a \leftarrow \mathbb{Z}_N$ and computes $A = g^a$, and Bob chooses a random $b \leftarrow \mathbb{Z}_N$ and computes $B = g^b$. Then the two parties exchange A, B. In most treatments, Diffie-Hellman is a *non-interactive key exchange* (NIKE), which means that A and B are sent simultaneously. Alice then computes the secret key $K = g^{ab} = B^a$ and Bob computes $K = g^{ab} = A^b$. By the DDH assumption, an eavesdropper who learns A, B can learn nothing about K.

The key issue here is that Alice and Bob need to know g in order to generate their first message. So if we want one of them, say Alice, to come up with the generator, the result is an *interactive* protocol with Alice sending the first message, and only then can Bob send his. Therefore, in addition to requiring slightly more communication, Option 3 actually changes the nature of the protocol. What we see is that Diffie-Hellman can only remain a setupless NIKE under the f-DDH assumption.

Now, it is possible to alter Diffie-Hellman to work with CDH by extracting hardcore bits from the unpredictable key. By the equivalence of f-CDH and r-CDH in known prime-order groups, we can obtain a setupless NIKE protocol from r-CDH (and hence also r-DDH). In groups of unknown order, however, this does not apply. As our main technical result from this section, we give evidence that in groups where the totient of the order is *unknown*, r-CDH alone is *insufficient* for constructing setupless NIKE. This is formalized by assuming that f-CDH is easy and demonstrating an attack on a wide class of key agreement protocols that generalize the classical Diffie-Hellman protocol.

Part 3: Random-Generator Discrete-Log and CDH with Preprocessing. A recent line of works [35, 33, 5, 18, 16] have explored *non-uniform* attacks on various problems in cryptographic groups. Here, a computationally expensive offline pre-processing stage generates an advice string, which in a later online stage can be used to speed up computation in the group. We are interested in the relationship between the length S of the advice string, the running time T of the online stage, the group order N, and the success probability ϵ.

Very recently, Corrigan-Gibbs and Kogan [18] seemingly resolve the non-uniform hardness of the *discrete logarithm* problem. Namely, they show in the generic group model that $\epsilon = \widetilde{O}(ST^2/N)$, where the $\widetilde{O}$ hides logarithmic factors. This matches known upper bounds (attacks) up to logarithmic factors.

However, all the works in this line (both lower bounds and attacks) only consider the fixed generator version of discrete log. Corrigan-Gibbs and Kogan briefly mention this, concluding that "using a fixed generator is essentially without loss of generality" since a discrete log with respect to one generator can be solved by solving two discrete logs with respect to a different generator.

When considering just polynomial reductions between problems, the above is certainly true. However, when it comes to precisely quantifying hardness, the problem no longer remains identical for different generators. In particular, suppose we have an algorithm that solves discrete log with respect to generator g with probability ϵ and we want to solve a discrete log instance with respect to generator $h = g^r$. To do so, on input h^x, we apply the algorithm twice to find the discrete logs of h and h^x with respect to g. This gives r and rx, allowing us to solve for x. But since we needed to solve both instances correctly, our overall success probability is only ϵ^2. Of course if ϵ is a constant so is ϵ^2, but in the low success probability regime, squaring the advantage significantly changes the hardness of the problem.

We resolve the question of the hardness of random-generator discrete log in the pre-processing setting, showing that $\epsilon = \widetilde{\Theta}\left(\frac{T^2}{N} + \frac{S^2T^4}{N^2}\right)$. The attack side is simple: there are two natural ways to attack a random-generator discrete log instance h, h^x. One is to ignore the pre-processing, and apply the Baby-step Giant-step algorithm, with success $\Omega(\frac{T^2}{N})$. The other is to use the pre-processing to solve two discrete log instances relative to some fixed generator g, in the manner described above. This gives success $\Omega((\frac{ST^2}{N})^2)$, as shown in [18]. By choosing which algorithm to use based on the parameters S, T, N, one obtains $\epsilon = \Omega\left(\frac{T^2}{N} + \frac{S^2T^4}{N^2}\right)$.

On the other hand, to prove the lower bound we need to show, essentially, that the two algorithms above are the only possible algorithms. This does not follow from the analysis of [18]. Instead, we use the tools developed in subsequent works [17, 16] (based on the earlier pre-sampling techniques developed by Unruh [41] for the Random Oracle model) to switch to a "bit-fixing" model, where we then show the optimality of the algorithms. In addition, we show that the same relationship holds as well for r-CDH. Generically, auxiliary input r-CDH is as hard as either using the auxiliary information to solve two discrete logarithms, or ignoring the input and solving one discrete logarithm.

1.3 Part 4: Low-Entropy Fixed-Generator Assumptions

Background: Point Obfuscation from Low-Entropy Assumptions. Our discussion thus far has focused on Discrete Log/Diffie-Hellman-type assumptions where g^a, g^b are uniformly random group elements. However, the security of many important cryptographic applications often relies on a stronger version of these assumptions in which a and/or b might not be drawn uniformly at random.

Canetti's construction of point function obfuscation is perhaps the most well-known example. A point function $f_x(\cdot)$ is a boolean function that accepts on x and rejects on all other inputs. Roughly speaking, an obfuscated point function $\mathcal{O}(f_x(\cdot))$ implements the same input/output functionality as $f_x(\cdot)$, but leaks no information about x beyond what can be learned through black-box oracle queries to $f_x(\cdot)$. In other words, the obfuscated program acts as a *virtual black box* for evaluating the function.[12] Canetti's point function obfuscator is simple: to obfuscate $f_x(\cdot)$, draw a random group element g^b and output (g^b, g^{xb}). Evaluation on input y is done by computing $(g^b)^y$ and accepting if it matches g^{xb}.

The security of this construction follows from an assumption Canetti refers to as DHI-II (in subsequent works it has been renamed to "DDH-II"; we will adopt this name), which states that $(g, g^a, g^b, g^{ab}) \approx_C (g, g^a, g^b, g^c)$ where g is a random generator, b, c are chosen uniformly at random, and a has super-logarithmic min-entropy, i.e. it is sampled from a *well-spread* distribution $\mathcal{D}$.

[12] We defer a more detailed discussion on virtual-black-box obfuscation to [1] (see [42] for specifics on point function obfuscation).

We stress that DDH-II is technically an infinite family of assumptions, since it requires indistinguishability if $\mathcal{D}$ is *any* well-spread distribution (even ones that are not efficiently sampleable).

Under DDH-II, the obfuscated program (g^b, g^{xb}) hides all information about the point x as long as x is drawn from a well-spread distribution, since g^{xb} is indistinguishable from g^c. This immediately implies a notion of average-case virtual-black-box (VBB) security. Canetti proves that if a point function obfuscator is average-case VBB for *any* well-spread distribution, this implies full (worst-case) VBB security. It was later shown by Wee ([42], Section 4.2) that Canetti's approach is essentially inherent: VBB-secure point function obfuscation *requires* strong assumptions that are hard for any well-spread distribution.

Background: Non-Malleable Point Obfuscation. Canetti's original motivation for studying point obfuscation was to realize useful properties of random oracles [4] in the standard model. If $H(\cdot)$ is a random oracle, observe that $H(x)$ is a secure point obfuscation of $f_x(\cdot)$, where evaluation is a single random oracle call followed by a comparison. Komargodski and Yogev [31] observe that the random oracle obfuscator $H(x)$ satisfies a strong *non-malleability* property, in the sense that given $H(x)$ it is impossible to compute $H(f(x))$ for any (meaningfully) related point $f(x)$, without first recovering x. This property is missing from Canetti's point obfuscator [13], e.g. since given (g^b, g^{xb}), one can easily compute $(g^b, g^{(x+1)b})$, which is an obfuscation of the related point $f(x) = x + 1$.

Komargodski and Yogev [31] propose the following modification to Canetti's point obfuscator. To obfuscate the point x, sample a random b and output $(g^b, (g^b)^{g^{x^4+x^3+x^2+x}})$. Note that for this expression to make sense, $g^{x^4+x^3+x^2+x}$ must be mapped back into the exponent space under some fixed public mapping. Evaluation on input y is done by computing $g^{y^4+y^3+y^2+y}$, mapping this element back to the exponent space and raising g^b to that power, and finally comparing to $(g^b)^{g^{x^4+x^3+x^2+x}}$.

Komargodski and Yogev [31] argue their obfuscation resists bounded-degree polynomial *mauling* attacks, in which an adversary given an obfuscation of x attempts to produce an obfuscation of $P(x)$ for some bounded-degree polynomial $P(\cdot)$. Roughly, the intuition is that the adversary cannot replace g^b with any other $g^{b'}$, since generating $(g^{b'})^{g^{P(x)}}$ does not appear possible given only $(g^b)^{g^{x^4+x^3+x^2+x}}$. But if the adversary cannot change g^b, the argument is that the linear constraints imposed by the form of $x^4 + x^3 + x^2 + x$ make it impossible to replace x with $P(x)$.

Formally, security in [31] is proved under the newly introduced "Strong Power DDH" assumption, which states it is hard to distinguish $g^x, g^{x^2}, \ldots, g^{x^\ell}$ from ℓ random group elements, if x is drawn from any well-spread distribution.

Fixed-Generator Strong Power DDH is False. In stating the assumption, Komargodski and Yogev [31] do not specify how g is chosen or the relationship between g and the distribution over x. We observe that if g is a fixed generator, then their assumption is false. For a uniformly random group element, there must be some bit in its description with noticeable entropy. If it is bit i, we let $\mathcal{D}$ be the distribution over all points x such that the ith bit of the description of g^x is 0. Then $\mathcal{D}$ has high min-entropy, and moreover g^x for $x \leftarrow \mathcal{D}$ is distinguishable from a random group element by inspecting the ith bit.

If the assumption is taken in its random-generator formulation, the security proof in [31] breaks down, since an adversary can potentially replace g with a different generator g'. A natural idea to fix the construction would be to generate g using a public source of randomness.[13] However, this would move the construction into the CRS model, where strong non-malleability results were previously known [15].

Fixing Non-Malleable Point Obfuscation. We remedy this situation by giving an alternative low-entropy fixed-generator assumption, and proving that this assumption is sufficient to achieve their notion of non-malleable point obfuscation. We formulate our assumption in a way that allows us to prove it holds in the generic group model. Our assumption is the following:

> Let $p \in [2^{\lambda-1}, 2^{\lambda}]$ and let n be at most $\mathsf{poly}(\lambda)$. Fix a group G of order p along with a generator g and any well-spread distribution $\mathcal{D}$ over $\mathbb{Z}_p$ (which can depend on G). Next sample $k_2, \ldots, k_n$ uniformly at random from $\mathbb{Z}_p$. Then no efficient adversary can distinguish $\{g^{k_i x + x^i}\}_{i \in \{2,\ldots,n\}}$ for $x \leftarrow \mathcal{D}$ from $n-1$ uniformly random group elements, even given $k_2, \ldots, k_n$.[14]

The intuition for the design of this assumption is the following. We want to modify the group elements $g^x, g^{x^2}, \ldots$ in Strong Power DDH to block distributions $\mathcal{D}$ which "condition" on the fixed g, as we have already seen how such distributions falsify the assumption. However, we are restricted to modifications that preserve our ability to perform a security reduction for the proof of non-malleability, as in [31].

Without delving into the non-malleability security proof itself, the key requirement is that the reduction must be able to construct specific polynomials (in x) in the exponent. We tweak the construction so that the reduction can construct a polynomial of the form $ax + x^2 + x^3 + x^4 + x^5$, where a is an arbitrary but known scalar. Then by using terms of the form $g^{k_i x + x^i}$, we enable the reduction to construct this polynomial by simply multiplying the $i = 2, \ldots, 5$ terms; it will know a since the k_i's are given in the clear. Intuitively, the k_i scalars contribute

[13] As noted in Section 1.1, Komargodski and Yogev have offered a fix through a new Entropic Power DDH Assumption in a revised ePrint posting [32], which does not come with a generic group proof. The goal of this section is to build non-malleable point obfuscation from an assumption that holds against generic adversaries.

[14] The assumption we actually use is slightly different: instead of stating indistinguishability from uniform, we require indistinguishability from $\{g^{k_i y + y^i}\}_{i \in \{2,\ldots,n\}}$ for the same $\{k_i\}_i$ but uniformly random y. We can prove both forms of this assumption hold in the GGM, but this second form yields a simpler proof of VBB security. For the purposes of this technical overview this distinction can be ignored.

enough randomness to prevent distributions $\mathcal{D}$ which make the $g^{k_i x + x^i}$ terms distinguishable from random.

Our resulting construction of non-malleable point obfuscation is (essentially) $a, g^{ax+x^2+x^3+x^4+x^5}$. We note that our construction does not require the "double exponentiation" of [31]. The full construction comes with two additional scalars and group elements that ensure that x is the only accepting input.

Discussion: Low-Entropy Fixed Generator Assumptions in the Generic Group Model. In order to gain confidence in our assumption, we prove it secure in the generic group model. As discussed in Section 1.1, this is usually viewed as a minimum requirement in order to gain confidence in a new group-based assumption. Recall that in the generic group model, group elements g^x are replaced with random "labels" $\sigma(x)$, where σ is a uniformly random injection from the space of exponents to some space of labels. An oracle stores the entire description of σ, and allows the generic adversary oracle access to honest group operations. For example, an adversary with labels $\sigma(x), \sigma(y)$ can request the label for $\sigma(x+y)$.

We find that in the setting of fixed generator lower entropy assumptions, the standard intuition for designing generic group model proofs falls short. Our goal is to prove no generic adversary can distinguish between $\{k_i, \sigma(k_i x + x^i)\}_{i \in \{2,\ldots,n\}}$ and $\{k_i, \sigma(r_i)\}_{i \in \{2,\ldots,n\}}$ for uniformly random k_i, r_i, and $x \leftarrow \mathcal{D}$. Since the group and generator are fixed in this assumption, we *must* consider distributions which depend on the group description itself. So in the generic model, any distribution $\mathcal{D}$ should be viewed as the output distribution of a potentially *inefficient* sampling algorithm $\mathcal{S}$ that is free to scan the entire labeling function σ. The only requirement we enforce is that given σ, the point $x \leftarrow \mathcal{S}(\sigma)$ has super-logarithmic entropy.

To illustrate the difference in this setting, suppose for a moment that the sampler $\mathcal{S}$ had to output x without seeing σ (as is the case when x is drawn uniformly at random from $\mathbb{Z}_p$). The standard generic group argument for indistinguishability would use the following structure:

> Imagine treating x as a formal variable instead of as a randomly drawn value. This replaces the group exponent space $\mathbb{Z}_p$ with formal polynomials $\mathbb{Z}_p[x]$, so the oracle now returns labels by sampling a uniformly random label from the image of σ each time it encounters a distinct formal polynomial. Observe that there are no (non-trivial) linear combinations of the $\{k_i x + x^i\}_i$ polynomials (taken as formal polynomials in x) that evaluate to identically zero polynomials over x. This implies that the adversary will never encounter non-trivial collisions in the labels it sees, and we can use the Schwartz-Zippel Lemma to argue that the adversary's view is identical in the world where x is random instead of a formal variable.

This type of argument breaks down if $\mathcal{S}$ can choose x *after* seeing the labeling function σ. Now $\mathcal{S}$ can try to pick x so that $\sigma(k_i x + x^i)$ conveys non-trivial distinguishing information to the adversary. In particular, it is no longer accurate to argue that we can produce an identical view for the adversary by replacing x with a formal variable.

We could intuitively hope that $\mathcal{S}$ is powerless to pick x that can bias the distribution of $\sigma(k_i x + x^i)$ away from uniform, as it does not know the random k_i. However, this intuition proves tricky to formalize, especially since $\mathcal{S}$ is given unlimited computational power and access to the entire function σ.

Connection to Preprocessing Attacks. To solve this problem, we apply the "bit-fixing" technique from Coretti, Dodis, and Guo [16]. They consider generic algorithms which are given an additional advice string, computed beforehand using a computationally unbounded algorithm with access to σ. Conditioned on the advice string, it is no longer accurate to argue σ is a random labeling function. However, they show (roughly) that if we obtain at most P bits of advice about σ, this only leaks useful information about σ on $O(P)$ points. So for generic security proofs, this allows us to switch to a setting in which σ is a random labeling function on all but $O(P)$ inputs.

We apply these techniques to our setting by re-casting the sampler $\mathcal{S}$ outputting x as a computationally unbounded algorithm outputting x as "advice". However in our setting, the challenger is the one receiving the advice instead of the adversary. It turns out that the [16] techniques still apply here, allowing us to argue that σ can be re-sampled on all but polynomially many points. Once we perform this re-sampling, we show that the adversary will not be able to apply group operations to its set of initial group elements and produce a point that was not re-sampled, except with negligible probability. Once this is established, standard generic group techniques suffice to complete the proof.

Generic Hardness of DDH-II. As a final contribution, we also prove the generic hardness of Canetti's DDH-II assumption. We remark that previous proofs of DDH-II [6, 20] operate in a highly idealized model that assumes the sampler is independent of the labeling function σ. Preventing the sampler from seeing the labels implicitly relies on the group itself being drawn at random, which in particular leads to counterexamples when dealing with fixed generator assumptions. For example, the Strong Power DDH assumption with fixed generator can be proven in this model even though it is false in the real world.

In the case of DDH-II, one of the elements the adversary receives is $\sigma(a)$ for low entropy a. We must show at a minimum that this does not allow the adversary to recover a (i.e. compute the discrete log), as distinguishing would then be trivial. Such a claim might not be immediately obvious, especially considering that we can *distinguish* $\sigma(a)$ from $\sigma(r)$ for uniform r for certain distributions on a. We observe that any adversary which succeeds in solving discrete log of $\sigma(a)$ with noticeable advantage for a well-spread distribution is also an adversary that solves discrete log (with much smaller advantage) for the uniform distribution. However, the resulting advantage exceeds the known generic bounds for discrete log algorithms [39]. The remainder of our proof makes use of bit-fixing techniques to reduce the problem of distinguishing the DDH-II instance to the problem of recovering a given just $\sigma(a)$.

2 Preliminaries

For $n \in \mathbb{N}$, let $[n]$ denote the set $\{1, \ldots, n\}$. We specify formal variables by bold letters $\mathbf{x}$. For a function f, let $im(f)$ denote the image of f.

Throughout, we let $\lambda \in \mathbb{N}$ be the security parameter. We use the usual Landau notations. A function $f(\lambda)$ is said to be negligible if it is $\lambda^{-\omega(1)}$ and we denote it by $f(\lambda) \coloneqq \mathsf{negl}(\lambda)$. A function $f(\lambda)$ is said to have polynomial growth rate if it is $\lambda^{O(1)}$ and we denote it by $f(\lambda) \coloneqq \mathsf{poly}(\lambda)$. A probability $p(\lambda)$ is said to be overwhelming if it is $1-\lambda^{-\omega(1)}$. We refer to $\mathcal{A}$ as PPT if it is a probabilistic polynomial time algorithm. If $\mathcal{A}$ has access to an oracle $\mathcal{O}$, we write $\mathcal{A}^{\mathcal{O}}$.

The statistical distance between two distributions D_1 and D_2 over a countable support S is defined to be $\Delta(D_1, D_2) \coloneqq \frac{1}{2}\sum_{x \in S} |D_1(x) - D_2(x)|$. Let $\gamma > 0$. We say that two distributions D_1 and D_2 are γ-close if $\Delta(D_1, D_2) \leq \gamma$. We let $x \leftarrow \mathcal{D}$ denote drawing x from the distribution $\mathcal{D}$. When X is a set, then $x \leftarrow X$ denotes drawing x *uniformly at random* from the set X. The following definition regarding infinite families of distributions will be used throughout.

Definition 1 (Well-Spread Distribution Ensemble). *An ensemble of distributions $\{\mathcal{D}_\lambda\}_\lambda$ over domains $\{\mathcal{X}_\lambda\}_\lambda$ is well-spread if for all large enough $\lambda \in \mathbb{N}$,*

$$H_\infty(\mathcal{D}_\lambda) = -\min_{x \in \mathcal{X}_\lambda} \log_2 \Pr[x \leftarrow \mathcal{D}_\lambda] = \omega(\log(\lambda)).$$

2.1 Generic Group Model

Definition 2 (Generic Group Model (GGM) [36, 39]). *An application in the generic group model is defined as an interaction between a T-attacker $\mathcal{A}$ and a challenger $\mathcal{C}$. For a cyclic group of order N with fixed generator g, a random injective function $\sigma : [N] \to [M]$ is sampled, mapping group exponents in $\mathbb{Z}_N$ to a set of labels $\mathcal{L}$. Label $\sigma(x)$ for $x \in \mathbb{Z}_N$ corresponds to the group element g^x.*

$\mathcal{C}$ initializes $\mathcal{A}$ with some set of labels $\{\sigma(x_i)\}_i$. It then implements the group operation oracle $\mathcal{O}_G(\cdot,\cdot)$, which on inputs $\sigma_1, \sigma_2 \in [M]$ does the following:

- *If either of σ_1 or σ_2 is not in $\mathcal{L}$, return $\perp$.*
- *Otherwise, set $x = \sigma^{-1}(\sigma_1)$ and $y = \sigma^{-1}(\sigma_2)$, compute $x + y \in \mathbb{Z}_N$, and return $\sigma(x+y)$.*

$\mathcal{A}$ is allowed at most T queries to the oracle, after which $\mathcal{C}$ outputs a bit indicating whether $\mathcal{A}$ was successful. We refer to the probability that this bit is 1 as $\mathsf{Succ}_{\mathcal{C}}(\mathcal{A})$.

Remark 2. It will often be convenient to represent each query to $\mathcal{O}_G$ as a linear polynomial over the initial set of elements $\{x_i\}_i$ given to $\mathcal{A}$.

For an *indistinguishability* application, we define the *advantage* of attacker $\mathcal{A}$ as $\mathsf{Adv}_{\mathcal{C}}(\mathcal{A}) = 2|\mathsf{Succ}_{\mathcal{C}}(\mathcal{A}) - 1/2|$. For an *unpredictability* application, the advantage is defined as $\mathsf{Adv}_{\mathcal{C}}(\mathcal{A}) = \mathsf{Succ}_{\mathcal{C}}(\mathcal{A})$. An application with associated challenger $\mathcal{C}$ is (T, ϵ)-secure in the GGM is for every T-attacker $\mathcal{A}$, $\mathsf{Adv}_{\mathcal{C}}(\mathcal{A}) \leq \epsilon$.

Definition 3 (Auxiliary-Input Generic Group Model (AI-GGM)). *We now consider (S, T)-attackers $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$. First $\sigma : [N] \to [M]$ is sampled. $\mathcal{A}_1$ receives σ as input and outputs an S-bit string* aux*. Then the challenger $\mathcal{C}$ operates as before, modeling interaction between $\mathcal{A}_2$ and $\mathcal{O}_G(\cdot, \cdot)$. Now $\mathcal{A}_2$ receives* aux *as input and is allowed T queries to the oracle. Success, advantage, and security are defined analogously.*

Definition 4 (Bit-Fixing Generic Group Model (BF-GGM)). *We now consider (S, T, P)-attackers $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$. First $\sigma : [N] \to [M]$ is sampled. $\mathcal{A}_1$ receives σ as input and outputs an S-bit string* aux *along with a set $\mathcal{P} \subseteq \mathbb{Z}_N$ of size P. Then σ is uniformly re-sampled on all but the points $\mathcal{P}$ (conditioned on maintaining the same image), producing the injection σ'. We let $im(\mathcal{P})$ refer to the images under σ and σ' of the points in $\mathcal{P}$. Then the challenger $\mathcal{C}$ operates as before, modeling interaction between $\mathcal{A}_2$ and $\mathcal{O}_G(\cdot, \cdot)$, where $\mathcal{O}_G(\cdot, \cdot)$ uses σ' to answer queries. $\mathcal{A}_2$ receives* aux *as input and is allowed T queries to the oracle. Success, advantage, and security are defined analogously.*

Theorem 1 ([16]). *Let $N, M, P \in \mathbb{N}$, $N \geq 16$, and $\gamma > 0$. If an unpredictability application with challenger $\mathcal{C}$ that initializes $\mathcal{A}$ with T' group elements is $((S, T, P), \epsilon')$-secure in the BF-GGM for*

$$P \geq 18(S + \log(\gamma^{-1}))(T + T'),$$

then it is $((S, T, P), \epsilon)$-secure in the AI-GGM for $\epsilon \leq 2\epsilon' + \gamma$.

3 Lower Bounds for Random Generator Discrete Log and CDH

We proceed to give tight lower bounds (up to logarithmic factors) for r-DLog and r-CDH in the AI-GGM, making use of the following special case of a lemma due to Yun [45].

Lemma 1 (Search-by-Hyperplane-Queries [45] (SHQ)). *Consider drawing z_1, z_2 uniformly at random from $\mathbb{Z}_N$, and allowing an adversary $\mathcal{A}$* hyperplane queries *of the form (a_1, a_2, b) where 1 is returned if $a_1 z_1 + a_2 z_2 = b$ and 0 otherwise. Then the probability that $\mathcal{A}$ outputs (z_1, z_2) after q hyperplane queries is at most q^2/N^2.*

Theorem 2. *The r-Dlog problem is $((S, T), \epsilon)$-secure in the AI-GGM for any prime $N \geq 16$ and*

$$\epsilon = \widetilde{O}\left(\frac{T^2}{N} + \left(\frac{ST^2}{N}\right)^2\right).$$

Proof. In the r-Dlog game, the challenger $\mathcal{C}$ draws $x \leftarrow \mathbb{Z}_N^*, y \leftarrow \mathbb{Z}_N$ and initializes $\mathcal{A}$ with $(\sigma(1), \sigma(x), \sigma(xy))$. $\mathcal{A}$ is successful if it outputs y after at most T generic group queries. We show that r-Dlog is

$\left((S,T), O\left(\frac{T^2}{N} + \frac{T^2P^2+T^3P}{N^2}\right)\right)$-secure in the BF-GGM. Then we can apply Theorem 1 with $\gamma = 1/N$ to get the result, noting that $T' = 3$ and $\log(1/\gamma) = \log(N)$, so $P = \widetilde{O}(ST)$.

$\mathcal{A} := \mathcal{A}_2$ takes as input the advice string aux generated by $\mathcal{A}_1$, makes T adaptive queries $\{c_1^{(t)}\sigma(x) + c_2^{(t)}\sigma(xy) + c_3^{(t)}\sigma(1)\}_{t\in[T]}$ to the generic group oracle and receives $\{\sigma(c_1^{(t)}x + c_2^{(t)}xy + c_3^{(t)})\}_{t\in[T]}$ in return. Let E be the event that there exists an $a \in \mathcal{P}$ and $t \in [T]$ such that $c_1^{(t)}x + c_2^{(t)}xy + c_3^{(t)} = a$ and $c_3^{(t)} \neq a$. Then

$$\Pr_{\sigma,x,y}[y \leftarrow \mathcal{A}^{\mathcal{O}_G}(\mathsf{aux})] \leq \Pr_{\sigma,x,y}[y \leftarrow \mathcal{A}^{\mathcal{O}_G}(\mathsf{aux}) \mid E] + \Pr_{\sigma,x,y}[y \leftarrow \mathcal{A}^{\mathcal{O}_G}(\mathsf{aux}) \mid \neg E].$$

We begin by analyzing the first probability in the sum. Condition on a particular image $\mathcal{L}$ of σ and a particular set of fixed points $\mathcal{P}$. The following holds for any such choice. We set up a reduction $\mathcal{B}$ which plays the SHQ game defined above and perfectly simulates the generic group game for $\mathcal{A}$. $\mathcal{B}$ has access to $\mathcal{L}, \mathcal{P}, im(\mathcal{P})$, and hyperplane query access to uniform values z_1, z_2 in $\mathbb{Z}_N$ which we implicitly set to be x, xy. We assume that $z_1 \neq 0$, which happens except with probability $1/N$. $\mathcal{B}$ operates as follows.

- Maintain a table mapping linear polynomials in $\mathbb{Z}_N[\mathbf{z_1}, \mathbf{z_2}]$ to $\mathcal{L}$. For each $a \in \mathcal{P}$, record the pair $(a, \sigma(a))$.
- Query the SHQ oracle on hyperplane $(1, 0, a)$ for each $a \in \mathcal{P}$. If any query returns 1, record the pair $(\mathbf{z_1}, \sigma(a))$, otherwise choose a uniform value r from all unused values in $\mathcal{L} \setminus im(\mathcal{P})$ and record $(\mathbf{z_1}, r)$. Do the same for $\mathbf{z_2}$. Next, store 1 along with its image. If $1 \in \mathcal{P}$ this is already done. If not, query $(1, 0, 1)$ to determine if $z_1 = 1$ and if so store 1 along with the image of $\mathbf{z_1}$. Do the same for $\mathbf{z}_2$. Otherwise, draw a uniform value r from all unused values in $\mathcal{L} \setminus im(\mathcal{P})$ and record $(1, r)$. Initialize $\mathcal{A}$ with the images of 1, $\mathbf{z_1}$, and $\mathbf{z_2}$.
- When $\mathcal{A}$ submits a query $c_1\mathbf{z_1} + c_2\mathbf{z_2} + c_3$, subtract each previously stored polynomial $Q(\mathbf{z_1}, \mathbf{z_2})$, resulting in some polynomial $k_1\mathbf{z_1} + k_2\mathbf{z_2} + k_3$. Query the SHQ oracle on $(k_1, k_2, -k_3)$. If 1 is returned, let s be the element stored along with $Q(\mathbf{z_1}, \mathbf{z_2})$, record $(c_1\mathbf{z_1} + c_2\mathbf{z_2} + c_3, s)$, and return s to $\mathcal{A}$. Otherwise, choose a uniform value r from all unused values in $\mathcal{L} \setminus im(\mathcal{P})$, record $(c_1\mathbf{z_1} + c_2\mathbf{z_2} + c_3, r)$ and return r.
- If E occurs, $\mathcal{B}$ will see a 1 returned by the SHQ oracle on a hyperplane query (k_1, k_2, k_3) for $k_3 \neq 0$, meaning at least one of $k_1, k_2 \neq 0$. Record this tuple. At the end of the interaction, $\mathcal{A}$ will return a $y \in \mathbb{Z}_N$. Now $\mathcal{B}$ outputs $(k_3(k_1 + k_2y)^{-1}, y)$.

Setting $z_1 = x$ and $z_2 = xy$, it is clear that $\mathcal{B}$ perfectly simulates the r-Dlog game for $\mathcal{A}$. If E occurs, we know that $k_3 = k_1x + k_2xy = x(k_1 + k_2y)$, and $k_3 \neq 0$, so $k_1 + k_2y \neq 0$. Thus if $\mathcal{A}$ is successful and returns y, $\mathcal{B}$ successfully computes $x = k_3(k_1 + k_2y)^{-1}$. Applying Lemma 1, and noting that $\mathcal{B}$ makes less than $2(P+1) + T(P+T) = O(TP + T^2)$ queries, we get that

$$\Pr_{\sigma,x,y}[y \leftarrow \mathcal{A}^{\mathcal{O}_G}(\mathsf{aux}) \mid E] = O\left(\frac{T^2P^2 + T^3P + T^4}{N^2}\right).$$

To analyze the second probability, we move to a hybrid game in the BF-GGM where x and y are set to be formal variables $\mathbf{x}$ and $\mathbf{y}$ at the beginning of the game. The challenger implements group operations over $\mathbb{Z}_N[\mathbf{x}, \mathbf{y}]$, initializing its table with the points in $(a, \sigma(a))$ for all $a \in \mathcal{P}$. Every time $\mathcal{A}$ queries for a new polynomial, $\mathcal{C}$ chooses a uniform element in $\mathcal{L} \setminus im(\mathcal{P})$ among those unused so far. When $\mathcal{A}$ outputs a guess for y at the end of the game, the true value is chosen uniformly at random, so $\mathcal{A}$ wins with probability $1/N$. Given that E does not occur, $\mathcal{A}$'s probability of distinguishing these two games is bounded by the probability that in the original game, two of its T queries are different polynomials over $\mathbf{x}$ and $\mathbf{y}$ but evaluate to the same element, or there exists some query $c_1^{(j)}\mathbf{x} + c_2^{(j)}\mathbf{xy} + c_3^{(j)}$ such that $c_1^{(j)}x + c_2^{(j)}xy = 0$ and at least one of $c_1^{(j)}, c_2^{(j)} \neq 0$. So there are $O(T^2)$ possible equations that could be satisfied and by Schwartz-Zippel, each occurs with probability $O(1/N)$ over the random choice of x and y. Thus by a union bound, $\mathcal{A}$'s probability of distinguishing is $O(T^2/N)$.

Combining, we have that $\mathcal{A}$'s probability of success is

$$O\left(\frac{T^2P^2 + T^3P + T^4}{N^2}\right) + O\left(\frac{T^2}{N}\right) + O\left(\frac{1}{N}\right) = O\left(\frac{T^2}{N} + \frac{T^2P^2 + T^3P}{N^2}\right).$$

□

In the full version [3], we use similar techniques to show the same bound for r-CDH, which again is tight.

Theorem 3. *The r-CDH problem is $((S,T),\epsilon)$-secure in the AI-GGM for any prime $N \geq 16$ and*

$$\epsilon = \widetilde{O}\left(\frac{T^2}{N} + \left(\frac{ST^2}{N}\right)^2\right).$$

4 Non-Malleable Point Obfuscation

In this section, we construct a non-malleable point obfuscator secure against *polynomial mauling attacks*, which were first considered by Komargodski and Yogev [31]. We first briefly review relevant definitions.

4.1 Definitions

Denote by $\mathcal{I}_x$ the function that returns 1 on input x and 0 otherwise.

Definition 5. *(Point Obfuscation) A point obfuscator for a domain $\{\mathcal{X}_\lambda\}_\lambda$ of inputs is a PPT Obf that takes as input a point $x \in \mathcal{X}_\lambda$ and outputs a circuit such that the following hold.*

- ***Functionality Preservation:*** *For all $\lambda \in \mathbb{N}$, there exists a negligible function μ such that for all $x \in \mathcal{X}_\lambda$,*

$$\Pr[\mathsf{Obf}(x) \equiv \mathcal{I}_x] = 1 - \mu(\lambda).$$

- ***Virtual Black Box (VBB) Security:*** *For all PPT $\mathcal{A}$ and any polynomial function p, there exists a PPT $\mathcal{S}$ such that for all $x \in \mathcal{X}_\lambda$ and any predicate $P : \mathcal{X}_\lambda \to \{0,1\}$, and all large enough λ,*

$$\left|\Pr[\mathcal{A}(\mathsf{Obf}(x)) = P(x)] - \Pr[\mathcal{S}^{\mathcal{I}_x}(1^\lambda) = P(x))]\right| \le \frac{1}{p(\lambda)}.$$

We give another property of point obfuscators first considered in [13] and re-defined in [7].

Definition 6 (Distributional Indistinguishability). *Let $\{\mathcal{X}_\lambda\}_\lambda$ be a family of domains. Then a point obfuscator* Obf *for $\{\mathcal{X}_\lambda\}_\lambda$ satisfies Distributional Indistinguishability if for all PPT $\mathcal{A}$ and well-spread ensembles of distributions $\{\mathcal{D}_\lambda\}_\lambda$ over $\{\mathcal{X}_\lambda\}_\lambda$, there exists a negligible function $\mu(\lambda)$ such that*

$$|\Pr[\mathcal{A}(\mathsf{Obf}(x)) = 1] - \Pr[\mathcal{A}(\mathsf{Obf}(u)) = 1]| = \mu(\lambda),$$

where $x \leftarrow \mathcal{D}_\lambda$ and u is drawn from the uniform distribution over $\mathcal{X}_\lambda$.

[13, 7] show that Distributional Indistiguishability is equivalent to VBB security for point obfuscators. Now we give the [31] definition of non-malleability. This definition involves the notion of a Verifier algorithm, which simply checks that the potentially mauled obfuscation is valid.

Definition 7. *(Verifier) A PPT $\mathcal{V}$ for a point obfuscator* Obf *for an ensemble of domains $\{\mathcal{X}_\lambda\}_\lambda$ is called a Verifier if for all $\lambda \in \mathbb{N}$ and $x \in \mathcal{X}_\lambda$, it holds that* $\Pr[\mathcal{V}(\mathsf{Obf}(x)) = 1] = 1$, *where the probability is taken over the randomness of $\mathcal{V}$ and* Obf.

Definition 8. *(Non-malleable Point Function Obfuscation) Let* Obf *be a point function obfuscator for an ensemble of domains $\{\mathcal{X}_\lambda\}_\lambda$ with an associated verifier $\mathcal{V}$. Let $\{\mathcal{F}_\lambda\}_\lambda = \{f : \mathcal{X}_\lambda \to \mathcal{X}_\lambda\}_\lambda$ be an ensemble of families of functions, and let $\{\mathcal{D}_\lambda\}_\lambda$ be an ensemble of distributions over $\mathcal{X}_\lambda$. Then* Obf *is a non-malleable point obfuscator for $\mathcal{F}$ and $\mathcal{D}$ if for any PPT $\mathcal{A}$, there exists a negligible function μ such that for any $\lambda \in \mathbb{N}$,*

$$\Pr[\mathcal{V}(C) = 1, f \in \mathcal{F}_\lambda, C \equiv \mathcal{I}_{f(x)} \mid x \leftarrow \mathcal{D}_\lambda, (C, f) \leftarrow \mathcal{A}(\mathsf{Obf}(x))] \le \mu(\lambda).$$

In the following, we rely on the existence of a *pseudo-deterministic* $\mathsf{GroupGen}$ algorithm that may use randomness, but on input the security parameter 1^λ outputs a *unique* description of a group $\mathbb{G}_\lambda$ with a unique generator g and prime order $p(\lambda) \in [2^{\lambda-1}, 2^\lambda]$. As discussed in the introduction, this would involve psuedo-deterministic generation of large primes. This is not provably efficient, but we can rely for example on Cramer's conjecture to argue efficiency. See [26] for further discussion on psuedo-deterministic algorithms, including group generator generation.

4.2 Assumptions

Assumption 1 *Let* $\mathsf{GroupGen}(1^\lambda) = (\mathbb{G}_\lambda, g, p(\lambda))$*, where* $2^{\lambda-1} < p(\lambda) < 2^\lambda$*. Let* $\{\mathcal{D}_\lambda\}$ *be a family of well-spread distributions where the domain of* $\mathcal{D}_\lambda$ *is* $\mathbb{Z}_{p(\lambda)}$*. Then for any* $n = poly(\lambda)$*, for any PPT* $\mathcal{A}$*,*

$$\left|\Pr[\mathcal{A}(\{k_i, g^{k_i x + x^i}\}_{i\in[2,\ldots,n]}) = 1] - \Pr[\mathcal{A}(\{k_i, g^{k_i r + r^i}\}_{i\in[2,\ldots,n]}) = 1]\right| = \mathsf{negl}(\lambda),$$

where $x \leftarrow \mathcal{D}_\lambda$*,* $r \leftarrow \mathbb{Z}_{p(\lambda)}$*, and* $k_i \leftarrow \mathbb{Z}_{p(\lambda)}$*.*

Assumption 2 *Let* $\mathsf{GroupGen}(1^\lambda) = (\mathbb{G}_\lambda, g, p(\lambda))$*, where* $2^{\lambda-1} < p(\lambda) < 2^\lambda$*. Let* $\{\mathcal{D}_\lambda\}$ *be a family of well-spread distributions where the domain of* $\mathcal{D}_\lambda$ *is* $\mathbb{Z}_{p(\lambda)}$*. Then for any* $n = poly(\lambda)$*, for any PPT* $\mathcal{A}$ *(which outputs an element of* $\mathbb{G}_\lambda$*),*

$$\Pr[g^x = \mathcal{A}(\{k_i, g^{k_i x + x^i}\}_{i\in[2,\ldots,n]})] = \mathsf{negl}(\lambda),$$

where $x \leftarrow \mathcal{D}_\lambda$ *and* $k_i \leftarrow \mathbb{Z}_{p(\lambda)}$*.*

We prove the following in the full version [3].

Lemma 2. *Assumption 1 implies Assumption 2.*

4.3 The Obfuscator

Our obfuscation consists of three scalars and three group elements. We remark that the first group element is sufficient for our proof on non-malleability, but that we include the next two to obtain functionality preservation.

- $\mathsf{Obf}(1^\lambda, x)$: Compute $\mathsf{GroupGen}(1^\lambda) = (\mathbb{G}_\lambda, g, p(\lambda))$. Draw $a, b, c \leftarrow \mathbb{Z}_{p(\lambda)}$ and output
$$a, b, c, g^{ax+x^2+x^3+x^4+x^5}, g^{bx+x^6}, g^{cx+x^7}.$$
- $\mathsf{Eval}(1^\lambda, (a, b, c, h_a, h_b, h_c), x)$: Compute $\mathsf{GroupGen}(1^\lambda) = (\mathbb{G}_\lambda, g, p(\lambda))$. Accept if and only if
$$h_a = g^{ax+x^2+x^3+x^4+x^5}, h_b = g^{bx+x^6}, h_c = g^{cx+x^7}.$$

Theorem 4. *The above point obfuscator satisfies functionality preservation.*

Proof. Fix a point $x \in \mathbb{Z}_{p(\lambda)}$. We show the probability that there exists a $y \neq x$ such that $\mathsf{Eval}(1^\lambda, \mathsf{Obf}(1^\lambda, x), y)$ accepts is at most $4/p(\lambda)^2$. Union bounding over all x completes the proof.

The randomness in Obf consists of the elements a, b, c. Fix just a for now and let $t = ax + x^2 + x^3 + x^4 + x^5$. Then any y which causes Eval to accept satisfies $ay + y^2 + y^3 + y^4 + y^5 = t$. This leaves four possible $y \neq x$. For each such y, we write $P(\mathbf{b}) = (x^6 - y^6) + (x - y)\mathbf{b}$ and $Q(\mathbf{c}) = (x^7 - y^7) + (x - y)\mathbf{c}$ which are linear polynomials over $\mathbf{b}$ and $\mathbf{c}$ respectively with non-zero linear coefficient. Then y only causes Eval to accept if $P(b) = 0$ and $P(c) = 0$. But these occur simultaneously with probability $1/p(\lambda)^2$ over the uniform randomness of b, c. So by a union bound, there exists a $y \neq x$ such that $\mathsf{Eval}(1^\lambda, \mathsf{Obf}(1^\lambda, x), y)$ accepts with probability at most $4/p(\lambda)^2$. □

Theorem 5. *Under Assumption 1, the above point obfuscator satisfies Virtual Black Box Security.*

Proof. The obfuscator satisfies distributional indistinguishability, which follows directly from Assumption 1 with $n = 7$. A reduction simply receives $\{k_i, h_i\}_{i \in [2,\dots,7]}$ and forms the obfuscation $(\sum_{i=2}^{5} k_i, k_6, k_7, \prod_{i=2}^{5} h_i, h_6, h_7)$. As mentioned earlier, this is equivalent to VBB security. □

Theorem 6. *Let $\{\mathcal{D}_\lambda\}$ be a well-spread distribution ensemble with domain $\{\mathbb{Z}_{p(\lambda)}\}_\lambda$. Let $\mathcal{F}_{poly} = \{f_\lambda : \mathbb{Z}_{p(\lambda)} \to \mathbb{Z}_{p(\lambda)}\}_\lambda$ be the ensemble of functions where f_λ is the set of non-constant, non-identity polynomials [15] in $\mathbb{Z}_{p(\lambda)}[x]$ with $\mathsf{poly}(\lambda)$ degree. Then under Assumption 1, the above obfuscator is non-malleable for $\mathcal{F}_{poly}$ and distribution ensemble $\{\mathcal{D}_\lambda\}$.*

Proof. First, we fix the verifier to check that the Eval circuit is using the g output by $\mathsf{GroupGen}(1^\lambda)$. Now we show that any mauling adversary $\mathcal{A}$ can be used to break Assumption 2, which as seen above follows from Assumption 1.

We first handle the case where $\mathcal{A}$ outputs an f of degree at least 2. Let $m \geq 2$ be the degree of $\mathcal{A}$'s polynomial. We define the following reduction $\mathcal{B}$.

- Receive $\{k_i, h_i\}_{i \in [2,\dots,7m]} := \{k_i, g^{k_i x + x^i}\}_{i \in [2,\dots,7m]}$ from the Assumption 2 challenger, where $x \leftarrow \mathcal{D}_\lambda$.
- Send $(\sum_{i=2}^{5} k_i, k_6, k_7, \prod_{i=2}^{5} h_i, h_6, h_7)$ to $\mathcal{A}$, which returns $(f, a, b, c, j_a, j_b, j_c)$ where $a, b, c \in \mathbb{Z}_{p(\lambda)}$ and j_a, j_b, j_c are group elements.
- Compute $cf(x) + f(x)^7 = \ell_0 + \ell_1 x + \cdots + \ell_{7m} x^{7m}$.
- Return $(j_c/(g^{\ell_0} \prod_{i=2}^{7m} (h_i^{\ell_i})))^{1/(\ell_1 - \sum_{i=2}^{7m} k_i \ell_i)}$.

$\mathcal{B}$ perfectly simulates the obfuscation for $x \leftarrow \mathcal{D}_\lambda$ for $\mathcal{A}$, which is guaranteed to return a valid obfuscation of $f(x)$ with $1/\mathsf{poly}(\lambda)$ probability. In this case, $f_c = g^{\ell_0 + \ell_1 x + \cdots + \ell_{7m} x^{7m}}$. Then $\mathcal{B}$ successfully computes g^x unless $\ell_1 - \sum_{i=2}^{7m} k_i \ell_i = 0$. We know that $\ell_{7m} \neq 0$ and that k_{7m} is uniformly random and independent of $\mathcal{A}$'s view, so this occurs with probability at most $1/p(\lambda) = \mathsf{negl}(\lambda)$. Thus, $\mathcal{B}$ breaks Assumption 2 with $1/\mathsf{poly}(\lambda)$ probability.

In the case that f is linear, we set up the same reduction $\mathcal{B}$, except for the last two steps.

- Compute $af(x) + f(x)^2 + f(x)^3 + f(x)^4 + f(x)^5 = \ell_0 + \ell_1 x + \cdots + \ell_5 x^5$.
- Return $(j_a/(g^{\ell_0} \prod_{i=2}^{5} (h_i^{\ell_i})))^{1/(\ell_1 - \sum_{i=2}^{5} k_i \ell_i)}$.

Like before, it suffices to argue that $\ell_1 - \sum_{i=2}^{5} k_i \ell_i \neq 0$ except with negligible probability. In this case, the adversary receives $z := k_2 + k_3 + k_4 + k_5$. Thus letting $k_5 = z - k_2 - k_3 - k_4$, there are 3 free variables k_2, k_3, k_4 in $\mathcal{A}$'s view. We can then re-write $\ell_1 - \sum_{i=2}^{5} k_i \ell_i \neq 0$ as

$$\ell_1 - \ell_5 z + (\ell_5 - \ell_2) k_2 + (\ell_5 - \ell_3) k_3 + (\ell_5 - \ell_4) k_4.$$

[15] Note that constant and identity polynomials correspond to "trivial" mauling attacks that cannot be prevented. A constant polynomial corresponds to picking an unrelated y and obfuscating y, while the identity polynomial corresponds to doing nothing.

So in order for this to evaluate to 0 with non-negligible probability, each of the coefficients on k_2, k_3, k_4 must be 0. Let $f(x) = rx + s$. Then writing out what the ℓ_i are, we see that the following must hold.

$$r^5 = 5r^4 s + r^4 = 10r^3 s^2 + 4r^3 s + r^3 = 10r^2 s^3 + 6r^2 s^2 + 3r^2 s + r^2$$

It is easily verified that the only solutions to the above system are when $r = 0$ or $(r = 1, s = 0)$. These correspond to when f is constant or the identity, so we can conclude that if $\mathcal{A}$ succeeds in breaking non-malleability, $\mathcal{B}$ breaks Assumption 2 with $1/\mathsf{poly}(\lambda)$ probability. $\square$

5 Justifying Assumptions in the Generic Group Model

We will need some additional background from [16], plus a couple of new simple lemmas. Note that while we make use of techniques from [16] that establish theorems relating the AI-GGM and BF-GGM, we never technically operate in the BF-GGM. We need a more fine-grained approach, starting in the plain GGM and modifying the labeling function and challenger's game incrementally.

5.1 Background

Definition 9 ([16]). *An (N, M)-injection source Σ is a random variable that takes on as value function tables corresponding to injections $\sigma : [N] \to [M]$. An (N, M)-injection source Σ is called $(P, \mathcal{L}, 1 - \delta)$-dense for $\mathcal{L} \subseteq [M]$ if it is fixed on at most P coordinates and if for every subset I of non-fixed coordinates,*

$$H_\infty(\Sigma_I) \geq (1 - \delta) \log \left(\frac{(N - P)!}{(N - P - |I|)!} \right),$$

where Σ_I is the random variable Σ restricted to the coordinates in I. When $\delta = 0$, the source is called $(P, \mathcal{L})$-fixed.

Remark 3. We denote by $\mathcal{A}^\Sigma$ an algorithm that has oracle access to an injection σ drawn from the source Σ. This means that $\mathcal{A}$ can perform forward queries where on input x the oracle returns $\sigma(x)$ or backward queries where on input x the oracle returns $\sigma^{-1}(x)$.

Lemma 3 ([16]). *Let Σ be a uniform (N, M)-injection source and $f : [M]^{[N]} \to \{0, 1\}^S$ a potentially randomized function. Let $\Sigma_{f,x,\mathcal{L}}$ be the random variable corresponding to the distribution of Σ conditioned on $f(\Sigma) = x$ and $im(\Sigma) = \mathcal{L}$. Then for any $\gamma > 0, P \in \mathbb{N}$, there exists a family $\{Y_{x,\mathcal{L}}\}_{x,\mathcal{L}}$, indexed by values $x \in \{0, 1\}^S$ and size-N subsets $\mathcal{L}$ of $[M]$, of convex combinations $Y_{x,\mathcal{L}}$ of $(P, \mathcal{L}, 1 - \frac{S + \log(1/\gamma)}{P \log(N/e)})$-dense sources, such that $\Sigma_{f,x,\mathcal{L}}$ is γ-close to $Y_{x,\mathcal{L}}$. Furthermore, replacing each $Y_{x,\mathcal{L}}$ with its corresponding convex combination $Z_{x,\mathcal{L}}$ of $(P, \mathcal{L})$-fixed sources, we have that for any distinguisher $\mathcal{D}$ taking an S-bit input and making at most T queries to its injection oracle,*

$$|\Pr[\mathcal{D}^{\Sigma}(f(\Sigma)) = 1] - \Pr[\mathcal{D}^{Z_{f(\Sigma), im(\Sigma)}}(f(\Sigma)) = 1]| \leq \frac{2(S + \log 1/\gamma) \cdot T}{P} + \gamma.$$

The above is actually slightly modified from the statement in [16], with the only difference being that we allow f to be randomized. The only place in their proof that makes use of f being deterministic is Claim 19, essentially that (where everything is conditioned on some range $\mathcal{L}$), $E_x[H_\infty(\Sigma|f(\Sigma)=x)] \geq \log(N!) - S$. Their proof of this claim can easily be adapted to allow randomized f. Say that f uses k uniformly random bits. Then define the deterministic function $f' : \{0,1\}^k \times [N]^{[N]} \to \{0,1\}^S$ that runs f using its first input as the randomness. Let K be the random variable corresponding to drawing a uniformly random string in $\{0,1\}^k$. Now by averaging, we have that for any x, $H_\infty(\Sigma|X=x) \geq H_\infty((K,\Sigma)|X=x) - k$. Then, following the proof in [16],

$$E_x[H_\infty(\Sigma|f(\Sigma)=x)] \geq E_x[H_\infty((K,\Sigma)|f'(K,\Sigma)=x)] - k$$
$$= E_x[H((K,\Sigma)|f'(K,\Sigma)=x)] - k \geq \log(N!) + k - S - k = \log(N!) - S,$$

where H is Shannon entropy, and the equality is due to the fact that conditioned on x, (K,Σ) is uniform over all values (r,σ) such that $f'(r,\sigma)=x$.

Lemma 4 ([16]). *For any $(P,N,1-\delta)$-dense (N,N)-injection (bijection) source Y and its corresponding (P,N)-fixed source Z, it holds that for any (adaptive) distinguisher $\mathcal{D}$ that makes at most T queries to its oracle,*

$$|\Pr[\mathcal{D}^Y = 1] - \Pr[\mathcal{D}^Z = 1]| \leq T\delta \log N.$$

Now we give two additional lemmas, useful for proving Theorem 7.

Lemma 5. *Let Σ be a uniform (N,N)-injection (bijection) source with $\log(N) = \Theta(\lambda)$ and $f : [N]^{[N]} \to \{0,1\}^S$ a potentially randomized function. Let Σ' be the random variable on σ' that results from drawing $\sigma \leftarrow \Sigma$, $x \leftarrow f(\sigma)$, and then $\sigma' \leftarrow \Sigma_{f,x,[N]}$ defined in Lemma 3. Say that for all σ, $H_\infty(X|\Sigma=\sigma) = \omega(\log(\lambda))$. Then*

$$E_{\Sigma'}[\max_x\{\Pr[X=x|\Sigma'=\sigma]\}] = \mathsf{negl}(\lambda).$$

Proof. With two applications of Bayes' Theorem, we see that for any $x \in \{0,1\}^S$

$$\Pr[X=x|\Sigma'=\sigma] = \frac{\Pr[\Sigma'=\sigma|X=x]\Pr[X=x]}{\Pr[\Sigma'=\sigma]} = \frac{\Pr[\Sigma=\sigma|X=x]\Pr[X=x]}{Pr[\Sigma'=\sigma]}$$
$$= \frac{\left(\frac{\Pr[X=x|\Sigma=\sigma]\Pr[\Sigma=\sigma]}{\Pr[X=x]}\right)\Pr[X=x]}{\Pr[\Sigma'=\sigma]} = \Pr[X=x|\Sigma=\sigma]\left(\frac{\Pr[\Sigma=\sigma]}{\Pr[\Sigma'=\sigma]}\right).$$

So plugging in,

$$E_{\Sigma'}[\max_x\{\Pr[X=x|\Sigma'=\sigma]\}] = \sum_\sigma \max_x\{\Pr[X=x|\Sigma'=\sigma]\}\Pr[\Sigma'=\sigma]$$
$$= \sum_\sigma \max_x\{\Pr[X=x|\Sigma=\sigma]\Pr[\Sigma=\sigma]\} \leq \max_{x,\sigma}\{\Pr[X=x|\Sigma=\sigma]\} = \mathsf{negl}(\lambda). \square$$

Lemma 6. *Consider n events $X_1, \ldots, X_n$ such that each event occurs with probability at least α, where $\alpha > 2/n$. Then for a uniformly random $i, j \leftarrow [n]$, $\Pr[X_i \wedge X_j] \geq \frac{\alpha^2}{4}$.*

The proof can be found in the full version [3].

5.2 Proofs

Theorem 7. *Assumption 1 (Section 4) holds in the Generic Group Model.*

Proof. We define the following hybrid games.

- **Hybrid 0.** The Assumption 1 distinguishing game for generic adversary $\mathcal{A}$.
  Let $\mathsf{GroupGen}(1^\lambda) = (\mathbb{G}_\lambda, g, p(\lambda))$, where $2^{\lambda-1} < p(\lambda) < 2^\lambda$. Let $p := p(\lambda)$. Sample a uniformly random injection $\sigma : [p] \to [p']$ for an arbitrary $p' > p$. Let $S : [p']^{[p]} \to \mathbb{Z}_p$ be a possibly inefficient randomized algorithm such that $H_\infty(S(\sigma)|\sigma) = \omega(\log(\lambda))$. Sample $x \leftarrow S(\sigma)$.
  The challenger $\mathcal{C}$ receives as input $(\mathbb{G}_\lambda, g, p, \sigma, x)$, chooses $b \leftarrow \{0,1\}, r, k_i \leftarrow \mathbb{Z}_p$ for $i \in [2, ..., n]$, and initializes the adversary $\mathcal{A}$ with $\{k_i, \sigma(b(k_i x + x^i) + (1-b)(k_i r + r^i))\}_{i \in [2,...n]}$. The challenger $\mathcal{C}$ proceeds to implement the generic group oracle for $\mathcal{A}$, after which $\mathcal{A}$ outputs a guess $b' \in \{0,1\}$. $\mathcal{A}$ wins if $b' = b$.
- **Hybrid 1.** In this hybrid, we switch to a "bit-fixing" labeling σ'.
  Let $\mathsf{GroupGen}(1^\lambda) = (\mathbb{G}_\lambda, g, p(\lambda))$, where $2^{\lambda-1} < p(\lambda) < 2^\lambda$. Let $p := p(\lambda)$. Sample a uniformly random injection $\sigma : [p] \to [p']$ for an arbitrary $p' > p$. Let $S : [p']^{[p]} \to \mathbb{Z}_p$ be a possibly inefficient randomized algorithm such that $H_\infty(S(\sigma)|\sigma) = \omega(\log(\lambda))$. Sample $x \leftarrow S(\sigma)$.
  Let $Z_{x,im(\sigma)}$ be the family defined as in Lemma 3 (parameterized by some $P \in \mathbb{N}$ and $\gamma := 1/2^\lambda$). Sample $\sigma' \leftarrow Z_{x,im(\sigma)}$
  The challenger $\mathcal{C}$ receives as input $(\mathbb{G}_\lambda, g, p, \sigma', x)$, chooses $b \leftarrow \{0,1\}, r, k_i \leftarrow \mathbb{Z}_p$ for $i \in [2, ..., n]$, and initializes the adversary $\mathcal{A}$ with $\{k_i, \sigma'(b(k_i x + x^i) + (1-b)(k_i r + r^i))\}_{i \in [2,...n]}$. The challenger $\mathcal{C}$ proceeds to implement the generic group oracle for $\mathcal{A}$, after which $\mathcal{A}$ outputs a guess $b' \in \{0,1\}$. $\mathcal{A}$ wins if $b' = b$.

Now we assume the existence of an adversary $\mathcal{A}$ that makes $T(\lambda) = \mathsf{poly}(\lambda)$ queries and attains non-negligible advantage $\epsilon(\lambda)$ in **Hybrid 0**. Let $q(\lambda) = \mathsf{poly}(\lambda)$ be such that $q(\lambda) > 1/\epsilon(\lambda)$ for infinitely many λ. Let $T := T(\lambda)$ and $q := q(\lambda)$. Set $P = 30\lambda T^4 q = \mathsf{poly}(\lambda)$.

Claim. $\mathcal{A}$ attains advantage at least $1/2q$ in **Hybrid 1**.

Consider the following distinguisher $\mathcal{D}(x)$, which interacts with an oracle injection source mapping $[p] \to [p']$, and receives as input $x \leftarrow S(\sigma)$. $\mathcal{D}$ simulates the interaction between $\mathcal{C}$ and $\mathcal{A}$ described in **Hybrid 0** and outputs a bit indicating whether $\mathcal{A}$ was successful or not. If the injection source that $\mathcal{D}$ is interacting with is σ, then the simulation is exactly **Hybrid 0**. If it is $Z_{x,im(\sigma)}$, then the simulation is exactly **Hybrid 1**.

Applying Lemma 3 with the sampler $x \leftarrow S(\sigma)$ as the function f, we have that the success probability of $\mathcal{A}$ in **Hybrid 1** must be at least

$$\epsilon(\lambda) - \frac{2T(\log p + \log(1/\gamma))}{P} - \gamma \geq \frac{1}{q} - \frac{4\lambda T}{30\lambda T^4 q} - \frac{1}{2^\lambda} \geq \frac{1}{2q}.$$

We show that $\mathcal{A}$ obtaining this advantage leads to a contradiction. Condition on $im(\sigma) = \mathcal{L}$ for some $\mathcal{L}$ where $\mathcal{A}$ obtains at least advantage $1/2q$. Here Σ is defined as in Lemma 3, except $[M]$ is fixed to be $\mathcal{L}$, resulting in a bijection source. We drop subscripts from the associated distributions, so $Y_x := Y_{x,\mathcal{L}}$, $Z_x := Z_{x,\mathcal{L}}$, and $\Sigma_x := \Sigma_{S,x,\mathcal{L}}$. The distribution Z_x is a convex combination of bit-fixing distributions $\mathcal{B}_x^{(j)}$ with associated fixed points $\mathcal{P}_x^{(j)}$. Let this convex combination be $\mathcal{J}_x$. So to draw σ' from Z_x, we draw $j \leftarrow \mathcal{J}_x$, then $\sigma' \leftarrow \mathcal{B}_x^{(j)}$.

Now we analyze the adversary's generic group oracle queries. Any query $\mathcal{A}$ makes can be viewed as a linear polynomial over its challenge elements

$$\ell_1 + \sum_{i=2}^{n} \ell_i(b(k_i x + x^i) + (1-b)(k_i r + r^i)),$$

specified by coefficients $[\ell_1, ..., \ell_n]$. We split these queries into two parts based on whether the linear polynomial is constant or non-constant over the challenge elements (whether there is some $i \in [2, ..., n]$ such that $\ell_i \neq 0$). We will consider each initial handle that $\mathcal{A}$ receives as a non-constant query where $\ell_i = 1$ for some i and $\ell_j = 0$ for $j \neq i$. Assume without loss of generality that all of $\mathcal{A}$'s queries are distinct linear combinations.

Note that constant queries are identically distributed in the $b = 0$ and $b = 1$ cases. Let $\mathcal{T}_c$ denote the set of constants that are queried by $\mathcal{A}$ throughout its interaction. Then observe that if, for both settings of b, all of $\mathcal{A}$'s non-constant queries result in distinct group elements that each lie outside of the set $\mathcal{P}_x^{(j)} \cup \mathcal{T}_c$, the oracle responses are identically distributed in both cases. Now, for any T-query adversary that at some point queries two distinct non-constant linear polynomials that evaluate to the same point, we can define a T^2-query adversary that at some point queries a non-constant linear polynomial that evaluates to zero. Redefine $\mathcal{A}$ to be this latter adversary. Thus if $\mathcal{A}$ distinguishes, it must at some point form a non-constant query that evaluates to a value in $\mathcal{P}_x^{(j)} \cup \mathcal{T}_c \cup \{0\}$.

For a given query t, let $\mathcal{T}_c^{(t)}$ denote the set of constants among the first t queries made by $\mathcal{A}$. There must exist some query t such that both of the following hold with probability $1/(2qT^2)$.

- t is non-constant and evaluates to an element in $\mathcal{P}_x^{(j)} \cup \mathcal{T}_c^{(t)} \cup \{0\}$ OR t is a constant c and there exists an earlier non-constant query t' such that query t' evaluates to c
- all previous non-constant queries (except perhaps t') evaluate to an element outside of $\mathcal{P}_x^{(j)} \cup \mathcal{T}_c^{(t)} \cup \{0\}$

Otherwise, by a union bound, $\mathcal{A}$ could not obtain distinguishing success $1/(2q)$. Note that every non-constant query prior to t except perhaps t' is answered with a uniformly random value in $\mathcal{L} \setminus im(\mathcal{P}_x^{(j)} \cup \mathcal{T}_c^{(t)} \cup \{0\})$. Since $|\mathcal{P}_x^{(j)} \cup \mathcal{T}_c^{(t)} \cup \{0\}| = \mathsf{poly}(\lambda)$, we can imagine instead drawing each response uniformly from $\mathcal{L}$, which by a union bound will change $\mathcal{A}$'s view with negligible probability. Then $\mathcal{A}$ can simulate these answers itself with uniform randomness, with a negligible difference in success probability.

Now we are left with an adversary $\mathcal{A}$ that takes as input $\{k_i\} := \{k_i\}_{i\in[2,\ldots,n]}$, makes at most T^2 queries to σ', and outputs a set of coefficients $[\ell_1, \ldots, \ell_n]$ (representing the non-constant query t or t'). Define $\mathcal{P}'^{(j)}_x := \mathcal{P}_x^{(j)} \cup \mathcal{T}_c^{(t)} \cup \{0\}$.

Now we break up the analysis into whether $b = 0$ or $b = 1$. If $b = 0$, we are guaranteed that with probability $1/(2qT^2) - \mathsf{negl}(\lambda) = 1/\mathsf{poly}(\lambda)$ over all randomness in the game setup, $\{k_i\}$, and $\mathcal{A}$, the following holds.

$$\ell_1 + \sum_{i=2}^{n} \ell_i(k_i r + r^i) \in \mathcal{P}'^{(j)}_x$$

But note that r is drawn uniformly at random from a set of size p, *independently* of $\mathcal{A}$'s view. Thus by Schwartz-Zippel and a union bound, the above holds with probability at most $(T^2 + P + 1)n/p = \mathsf{negl}(\lambda)$.

Now let $b = 1$. We are guaranteed that with probability $1/(2qT^2) - \mathsf{negl}(\lambda)$ over all randomness in the game setup, $\{k_i\}$, and $\mathcal{A}$, the following holds:

$$\ell_1 + \sum_{i=2}^{n} \ell_i(k_i x + x^i) \in \mathcal{P}'^{(j)}_x.$$

Redefine $\mathcal{A}$ to output the above polynomial $Q(x) \in \mathbb{Z}_p[x]$ on input $\{k_i\}$. Now accounting for all randomness during the course of the game, we have that

$$\Pr_{\substack{\sigma\leftarrow\Sigma, x\leftarrow S(\sigma), j\leftarrow\mathcal{J}_x,\\ \sigma'\leftarrow\mathcal{B}_x^{(j)}, \{k_i\}\leftarrow\mathbb{Z}_p^{n-1}, \mathcal{A}}} [Q(x) \in \mathcal{P}'^{(j)}_x : Q \leftarrow \mathcal{A}^{\sigma'}(\{k_i\})] = \frac{1}{2qT^2} - \mathsf{negl}(\lambda).$$

Now we switch the distribution on σ' from $Z = \{Z_x\}_x$ to $Y = \{Y_x\}_x$. We can still represent Y_x in the same way as Z_x except the $\mathcal{B}_x^{(j)}$'s are replaced by $(P, 1-\delta)$-dense sources $\mathcal{D}_x^{(j)}$. Referring to the Lemma 3 statement, we have that

$$\delta \leq \frac{2\lambda + \log 1/\gamma}{P \log(p/e)} \leq \frac{1}{10T^4 q \log(p/e)}.$$

Now assume towards contradiction that this switch in distribution causes the adversary's success to become at most $1/(4qT^2)$. Then there must exist some fixed choice of σ, x and j such that $\mathcal{A}$'s difference in success over σ' and its input is at least $1/(4qT^2) - \mathsf{negl}(\lambda)$. So we have

$$\Pr_{\sigma'\leftarrow\mathcal{B}_x^{(j)},\{k_i\}\leftarrow\mathbb{Z}_p^{n-1},\mathcal{A}}[Q(x)\in{\mathcal{P}'}_x^{(j)} : Q\leftarrow\mathcal{A}^{\sigma'}(\{k_i\})] -$$
$$\Pr_{\sigma'\leftarrow\mathcal{D}_x^{(j)},\{k_i\}\leftarrow\mathbb{Z}_p^{n-1},\mathcal{A}}[Q(x)\in{\mathcal{P}'}_x^{(j)} : Q\leftarrow\mathcal{A}^{\sigma'}(\{k_i\})] \geq \frac{1}{4qT^2} - \mathsf{negl}(\lambda).$$

But now we can define a distinguisher that contradicts Lemma 4. The distinguisher knows the fixed x and the set of fixed points $\mathcal{P}_x^{(j)}$, and interacts with either $\mathcal{B}_x^{(j)}$ or $\mathcal{D}_x^{(j)}$, simulating $\mathcal{A}$ making T^2 queries. It can tell whether $\mathcal{A}$ succeeds by plugging x into the polynomial produced and comparing the result to the set of fixed points and the set of queries made by $\mathcal{A}$. Yet it can only distinguish with probability at most $T^2\delta\log p \leq 1/(5qT^2)$ which is a contradiction.

Now we imagine picking c uniformly at random from ${\mathcal{P}'}_x^{(j)}$. Since $1/(4qT^2) = 1/\mathsf{poly}(\lambda)$ and $|{\mathcal{P}'}_x^{(j)}| \leq T^2 + P + 1 = \mathsf{poly}(\lambda)$, we can say that

$$\Pr_{\substack{\sigma\leftarrow\Sigma,x\leftarrow S(\sigma),j\leftarrow\mathcal{J}_x,\sigma'\leftarrow\mathcal{D}_x^{(j)},\\ c\leftarrow{\mathcal{P}'}_x^{(j)},\{k_i\}\leftarrow\mathbb{Z}_p^{n-1},\mathcal{A}}}[Q(x)=c : Q\leftarrow\mathcal{A}^{\sigma'}(\{k_i\})] = \frac{1}{\mathsf{poly}(\lambda)}.$$

Now there must exist a $1/\mathsf{poly}(\lambda)$ fraction of $\{k_i\}$ such that the above holds with probability $1/\mathsf{poly}(\lambda)$ on each of those inputs. Denote this set $\mathcal{K}$, where $\mathcal{K}_i$ denotes the ith element of the set. We also now give σ' as an input to $\mathcal{A}$ rather than just giving it oracle access. So we have

$$\Pr_{\substack{\sigma\leftarrow\Sigma,x\leftarrow S(\sigma),j\leftarrow\mathcal{J}_x,\\ \sigma'\leftarrow\mathcal{D}_x^{(j)},c\leftarrow{\mathcal{P}'}_x^{(j)},\mathcal{A}}}[Q(x)=c : Q\leftarrow\mathcal{A}(\sigma',\mathcal{K}_i)] = \frac{1}{\mathsf{poly}(\lambda)} \quad \forall i\in[|\mathcal{K}|].$$

Then by Lemma 6, noting that $|\mathcal{K}| = \omega(\mathsf{poly}(\lambda))$,

$$\Pr_{\substack{\sigma\leftarrow\Sigma,x\leftarrow S(\sigma),j\leftarrow\mathcal{J}_x,\sigma'\leftarrow\mathcal{D}_x^{(j)},\\ c\leftarrow{\mathcal{P}'}_x^{(j)},\mathcal{A},i_1,i_2\leftarrow[|\mathcal{K}|]}}\left[Q_1(x)=c=Q_2(x) : \begin{array}{c} Q_1\leftarrow\mathcal{A}(\sigma',\mathcal{K}_{i_1}) \\ Q_2\leftarrow\mathcal{A}(\sigma',\mathcal{K}_{i_2}) \end{array}\right] = \frac{1}{\mathsf{poly}(\lambda)}.$$

Thus we can get rid of c, and are guaranteed that

$$\Pr_{\substack{\sigma\leftarrow\Sigma,x\leftarrow S(\sigma),j\leftarrow\mathcal{J}_x,\\ \sigma'\leftarrow\mathcal{D}_x^{(j)},\mathcal{A},i_1,i_2\leftarrow[|\mathcal{K}|]}}\left[Q_1(x)-Q_2(x)=0 : \begin{array}{c} Q_1\leftarrow\mathcal{A}(\sigma',\mathcal{K}_{i_1}) \\ Q_2\leftarrow\mathcal{A}(\sigma',\mathcal{K}_{i_2}) \end{array}\right] = \frac{1}{\mathsf{poly}(\lambda)}.$$

Now since $\mathcal{K}$ is a $1/\mathsf{poly}(\lambda)$ fraction of the entire domain of $\{k_i\}$, we can instead pick these sets from the entire domain, and with $1/\mathsf{poly}(\lambda)$ probability they will both lie in $\mathcal{K}$. This gives

$$\Pr_{\substack{\sigma\leftarrow\Sigma,x\leftarrow S(\sigma),j\leftarrow\mathcal{J}_x,\sigma'\leftarrow\mathcal{D}_x^{(j)},\\ \mathcal{A},\{k_i^{(1)}\},\{k_i^{(2)}\}\leftarrow\mathbb{Z}_p^{n-1}}}\left[Q_1(x)-Q_2(x)=0 : \begin{matrix} Q_1\leftarrow\mathcal{A}(\sigma',\{k_i^{(1)}\}) \\ Q_2\leftarrow\mathcal{A}(\sigma',\{k_i^{(2)}\})\end{matrix}\right] = \frac{1}{\mathsf{poly}(\lambda)}.$$

Now we look at the probability that Q_1 and Q_2 are distinct polynomials. For any fixed Q, there are at most a $1/p$ fraction of sets $\{k_i\}$ such that $\mathcal{A}(\{k_i\})$ could possibly output Q. This follows since given some $\{k_i\}$, the coefficients on $x^2,...,x^n$ in Q determine the $\ell_2,...,\ell_n$ in $\mathcal{A}$'s linear combination. Then there remains a $1/p$ chance that the $\{\ell_i\}$ and $\{k_i\}$ dot product to the correct linear coefficient in Q. So for uniformly random choice of the $\{k_i^{(1)}\}$ and $\{k_i^{(2)}\}$ sets, there is a $\mathsf{negl}(\lambda)$ chance that the resulting Q_1 and Q_2 output by $\mathcal{A}$ could possibly be equal.

Let E_1 be the event that $Q_1(x)-Q_2(x)=0$ and E_2 be the event that $Q_1\neq Q_2$. We want to say that $\Pr[E_1\wedge E_2]=1/\mathsf{poly}(\lambda)$. This follows from a simple union bound: $\Pr[E_1\wedge E_2]=1-\Pr[\neg E_1\vee\neg E_2]\geq 1-\Pr[\neg E_1]-\Pr[\neg E_2]=1-(1-1/\mathsf{poly}(\lambda))-\mathsf{negl}(\lambda)=1/\mathsf{poly}(\lambda)$.

So we redefine $\mathcal{A}$ to generate two random sets $\{k_i^{(1)}\}$ and $\{k_i^{(2)}\}$ for itself, determine the polynomials Q_1 and Q_2, solve for the roots of Q_1-Q_2, and output a uniformly random root. Note that the degree of Q_1-Q_2 will be at most $n=\mathsf{poly}(\lambda)$. Thus the following holds:

$$\Pr_{\sigma\leftarrow\Sigma,x\leftarrow S(\sigma),\sigma'\leftarrow Y_x,\mathcal{A}}[x\leftarrow\mathcal{A}(\sigma')]=\frac{1}{\mathsf{poly}(\lambda)}.$$

Now we can switch Y_x to Σ_x, and claim that

$$\Pr_{\sigma\leftarrow\Sigma,x\leftarrow S(\sigma),\sigma'\leftarrow\Sigma_x,\mathcal{A}}[x\leftarrow\mathcal{A}(\sigma')]=\frac{1}{\mathsf{poly}(\lambda)}.$$

If instead $\mathcal{A}$'s success was negligible after this switch, then there exists a fixed x for which the difference in success is $1/\mathsf{poly}(\lambda)$. But Y_x and Σ_x are γ-close with $\gamma=1/2^\lambda=\mathsf{negl}(\lambda)$ so this is impossible. Then, we can write

$$\Pr_{\sigma'\leftarrow\Sigma',\mathcal{A}}[x\leftarrow\mathcal{A}(\sigma')]=\frac{1}{\mathsf{poly}(\lambda)},$$

where Σ' is defined as in Lemma 5. This contradicts Lemma 5.

□

Assumption 3 *(f-DDH-II) Let* $\mathsf{GroupGen}(1^\lambda)=(\mathbb{G}_\lambda,g,p(\lambda))$, *where* $2^{\lambda-1}<p(\lambda)<2^\lambda$. *Let* $\{\mathcal{D}_\lambda\}_\lambda$ *be a family of well-spread distributions where the domain of* $\mathcal{D}_\lambda$ *is* $\mathbb{Z}_{p(\lambda)}$. *Then for any PPT* $\mathcal{A}$,

$$|\Pr[\mathcal{A}(g^x,g^r,g^{xr})=1]-\Pr[\mathcal{A}(g^x,g^r,g^s)=1]|=\mathsf{negl}(\lambda),$$

where $x\leftarrow\mathcal{D}_\lambda$, *and* $r,s\leftarrow\mathbb{Z}_{p(\lambda)}$.

Theorem 8. *Assumption 3 holds in the Generic Group Model.*

We give the proof in the full version [3]. Note that this trivially implies generic security of r-DDH-II.

6 Acknowledgements

We thank Justin Holmgren for collaboration in the early stages of this work and for contributing a number of extremely valuable insights. We also thank Alon Rosen for helpful feedback regarding exposition and presentation.

This material is based upon work supported by the ARO and DARPA under Contract No. W911NF-15-C-0227. 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 ARO and DARPA.

References

1. Barak, B., Goldreich, O., Impagliazzo, R., Rudich, S., Sahai, A., Vadhan, S.P., Yang, K.: On the (im)possibility of obfuscating programs. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 1–18. Springer, Heidelberg (Aug 2001)
2. Barthe, G., Fagerholm, E., Fiore, D., Mitchell, J.C., Scedrov, A., Schmidt, B.: Automated analysis of cryptographic assumptions in generic group models. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014, Part I. LNCS, vol. 8616, pp. 95–112. Springer, Heidelberg (Aug 2014)
3. Bartusek, J., Ma, F., Zhandry, M.: The distinction between fixed and random generators in group-based assumptions. Cryptology ePrint Archive, Report 2019/202 (2019), https://eprint.iacr.org/2019/202
4. Bellare, M., Rogaway, P.: Random oracles are practical: A paradigm for designing efficient protocols. In: Ashby, V. (ed.) ACM CCS 93. pp. 62–73. ACM Press (Nov 1993)
5. Bernstein, D.J., Lange, T.: Non-uniform cracks in the concrete: The power of free precomputation. In: Sako, K., Sarkar, P. (eds.) ASIACRYPT 2013, Part II. LNCS, vol. 8270, pp. 321–340. Springer, Heidelberg (Dec 2013)
6. Bitansky, N., Canetti, R.: On strong simulation and composable point obfuscation. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 520–537. Springer, Heidelberg (Aug 2010)
7. Bitansky, N., Canetti, R.: On strong simulation and composable point obfuscation. Journal of Cryptology 27(2), 317–357 (Apr 2014)
8. Boneh, D., Franklin, M.K.: Identity-based encryption from the Weil pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (Aug 2001)
9. Boneh, D., Silverberg, A.: Applications of multilinear forms to cryptography. Contemporary Mathematics 324, 71–90 (2002)
10. Boneh, D., Zhandry, M.: Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014, Part I. LNCS, vol. 8616, pp. 480–499. Springer, Heidelberg (Aug 2014)

11. Bos, J.W., Costello, C., Longa, P., Naehrig, M.: Selecting elliptic curves for cryptography: an efficiency and security analysis. Journal of Cryptographic Engineering 6(4), 259–286 (Nov 2016), https://doi.org/10.1007/s13389-015-0097-y
12. Brands, S.: Untraceable off-line cash in wallets with observers (extended abstract). In: Stinson, D.R. (ed.) CRYPTO'93. LNCS, vol. 773, pp. 302–318. Springer, Heidelberg (Aug 1994)
13. Canetti, R.: Towards realizing random oracles: Hash functions that hide all partial information. In: Kaliski Jr., B.S. (ed.) CRYPTO'97. LNCS, vol. 1294, pp. 455–469. Springer, Heidelberg (Aug 1997)
14. Canetti, R., Dakdouk, R.R.: Extractable perfectly one-way functions. In: Aceto, L., Damgård, I., Goldberg, L.A., Halldórsson, M.M., Ingólfsdóttir, A., Walukiewicz, I. (eds.) ICALP 2008, Part II. LNCS, vol. 5126, pp. 449–460. Springer, Heidelberg (Jul 2008)
15. Canetti, R., Varia, M.: Non-malleable obfuscation. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 73–90. Springer, Heidelberg (Mar 2009)
16. Coretti, S., Dodis, Y., Guo, S.: Non-uniform bounds in the random-permutation, ideal-cipher, and generic-group models. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018, Part I. LNCS, vol. 10991, pp. 693–721. Springer, Heidelberg (Aug 2018)
17. Coretti, S., Dodis, Y., Guo, S., Steinberger, J.P.: Random oracles and non-uniformity. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018, Part I. LNCS, vol. 10820, pp. 227–258. Springer, Heidelberg (Apr / May 2018)
18. Corrigan-Gibbs, H., Kogan, D.: The discrete-logarithm problem with preprocessing. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018, Part II. LNCS, vol. 10821, pp. 415–447. Springer, Heidelberg (Apr / May 2018)
19. Cramer, R., Shoup, V.: A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In: Krawczyk, H. (ed.) CRYPTO'98. LNCS, vol. 1462, pp. 13–25. Springer, Heidelberg (Aug 1998)
20. Damgård, I., Hazay, C., Zottarel, A.: Short paper on the generic hardness of ddh-ii (2014)
21. Diffie, W., Hellman, M.E.: New directions in cryptography. IEEE Transactions on Information Theory 22(6), 644–654 (1976)
22. ElGamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. In: Blakley, G.R., Chaum, D. (eds.) CRYPTO'84. LNCS, vol. 196, pp. 10–18. Springer, Heidelberg (Aug 1984)
23. Fenteany, P., Fuller, B.: Non-malleable digital lockers for efficiently sampleable distributions. Cryptology ePrint Archive, Report 2018/957 (2018), https://eprint.iacr.org/2018/957
24. Fujisaki, E.: Improving practical UC-secure commitments based on the DDH assumption. In: Zikas, V., De Prisco, R. (eds.) SCN 16. LNCS, vol. 9841, pp. 257–272. Springer, Heidelberg (Aug / Sep 2016)
25. Galbraith, S.D.: Mathematics of public key cryptography. Cambridge University Press (2012)
26. Gat, E., Goldwasser, S.: Probabilistic search algorithms with unique answers and their cryptographic applications. Electronic Colloquium on Computational Complexity (ECCC) 18, 136 (2011)
27. Goldwasser, S., Kalai, Y.T.: Cryptographic assumptions: A position paper. In: Kushilevitz, E., Malkin, T. (eds.) TCC 2016-A, Part I. LNCS, vol. 9562, pp. 505–522. Springer, Heidelberg (Jan 2016)
28. Kalai, Y.T., Li, X., Rao, A., Zuckerman, D.: Network extractor protocols. In: 49th FOCS. pp. 654–663. IEEE Computer Society Press (Oct 2008)

29. Katz, J., Lindell, Y.: odern Cryptography, Second Edition (2014)
30. Kim, J., Kim, S., Seo, J.H.: Multilinear map via scale-invariant FHE: Enhancing security and efficiency. Cryptology ePrint Archive, Report 2015/992 (2015), https://ia.cr/2015/992
31. Komargodski, I., Yogev, E.: Another step towards realizing random oracles: Non-malleable point obfuscation. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018, Part I. LNCS, vol. 10820, pp. 259–279. Springer, Heidelberg (Apr / May 2018)
32. Komargodski, I., Yogev, E.: Another step towards realizing random oracles: Non-malleable point obfuscation. Cryptology ePrint Archive, Report 2018/149 (2018), https://ia.cr/2018/149
33. Lee, H.T., Cheon, J.H., Hong, J.: Accelerating ID-based encryption based on trapdoor DL using pre-computation. Cryptology ePrint Archive, Report 2011/187 (2011), https://ia.cr/2011/187
34. Maurer, U.M.: Abstract models of computation in cryptography (invited paper). In: Smart, N.P. (ed.) 10th IMA International Conference on Cryptography and Coding. LNCS, vol. 3796, pp. 1–12. Springer, Heidelberg (Dec 2005)
35. Mihalcik, J.: An analysis of algorithms for solving discrete logarithms in fixed groups (2010), master's thesis, Naval Postgraduate School
36. Nechaev, V.I.: Complexity of a determinate algorithm for the discrete logarithm. Mathematical Notes 55(2), 165–172 (1994)
37. Sadeghi, A.R., Steiner, M.: Assumptions related to discrete logarithms: Why subtleties make a real difference. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 244–261. Springer, Heidelberg (May 2001)
38. Saxena, A., Soh, B.: A new cryptosystem based on hidden order groups. Cryptology ePrint Archive, Report 2006/178 (2006), https://ia.cr/2006/178
39. Shoup, V.: Lower bounds for discrete logarithms and related problems. In: Fumy, W. (ed.) EUROCRYPT'97. LNCS, vol. 1233, pp. 256–266. Springer, Heidelberg (May 1997)
40. Shoup, V.: On formal models for secure key exchange. Tech. Rep. RZ 3120, IBM (1999)
41. Unruh, D.: Random oracles and auxiliary input. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 205–223. Springer, Heidelberg (Aug 2007)
42. Wee, H.: On obfuscating point functions. In: Gabow, H.N., Fagin, R. (eds.) 37th ACM STOC. pp. 523–532. ACM Press (May 2005)
43. Yamakawa, T., Yamada, S., Hanaoka, G., Kunihiro, N.: Self-bilinear map on unknown order groups from indistinguishability obfuscation and its applications. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014, Part II. LNCS, vol. 8617, pp. 90–107. Springer, Heidelberg (Aug 2014)
44. Yamakawa, T., Yamada, S., Hanaoka, G., Kunihiro, N.: Generic hardness of inversion on ring and its relation to self-bilinear map. Cryptology ePrint Archive, Report 2018/463 (2018), https://ia.cr/2018/463
45. Yun, A.: Generic hardness of the multiple discrete logarithm problem. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015, Part II. LNCS, vol. 9057, pp. 817–836. Springer, Heidelberg (Apr 2015)