Nonmalleable Digital Lockers and Robust Fuzzy Extractors in the Plain Model

Daniel Apon* Chloe Cachet† Benjamin Fuller‡ Peter Hall§ Feng-Hao Liu¶

September 19, 2022

Abstract

We give the first constructions in the plain model of 1) nonmalleable digital lockers (Canetti and Varia, TCC 2009) and 2) robust fuzzy extractors (Boyen et al., Eurocrypt 2005) that secure sources with entropy below 1/2 of their length. Constructions were previously only known for both primitives assuming random oracles or a common reference string (CRS).

Along the way, we define a new primitive called a nonmalleable point function obfuscation with associated data. The associated data is public but protected from all tampering. We use the same paradigm to then extend this to digital lockers. Our constructions achieve nonmalleability over the output point by placing a CRS into the associated data and using an appropriate non-interactive zero-knowledge proof. Tampering is protected against the input point over low-degree polynomials and over any tampering to the output point and associated data. Our constructions achieve virtual black box security.

These constructions are then used to create robust fuzzy extractors that can support low-entropy sources in the plain model. By using the geometric structure of a syndrome secure sketch (Dodis et al., SIAM Journal on Computing 2008), the adversary's tampering function can always be expressed as a low-degree polynomial; thus, the protection provided by the constructed nonmalleable objects suffices.
Keywords: Point obfuscation, digital lockers, nonmalleability, virtual black box obfuscation, fuzzy extractors.

1 Introduction

The random oracle (RO) paradigm [9] allows one to analyze cryptographic primitives/protocols with an idealized random function, significantly simplifying the designs and analyses. Since instantiating RO with a real-life object is impossible for the general case [23], it is important to identify useful RO properties that are achievable under specific hard problems.

Initial Efforts – Point Obfuscation. Canetti [20] initiated a study on an important property of random oracles called *oracle hashing* — or *point obfuscation* — and realized it in the plain model. More

*MITRE. Email dapon.crypto@gmail.com
†University of Connecticut. Email chloe.cachet@uconn.edu.
‡University of Connecticut. Email benjamin.fuller@uconn.edu
§New York University. Email pf2184@nyu.edu
¶Florida Atlantic University. Email liuf@fau.edu

specifically, a point function I_{val} is indexed by a string val and acts as follows:

$$I_{\mathsf{val}}(\mathsf{val}') = \begin{cases} 1 & \mathsf{val} = \mathsf{val}' \\ 0 & \text{otherwise} \end{cases}.$$

An obfuscated point function should reveal nothing beyond the input/output behavior of the function $I_{\mathsf{val}}(\cdot)$. This security notion is called virtual black-box (VBB) security. Constructions are known from multiple assumptions [20, 48, 3, 28].

VBB secure obfuscation of point functions captures the idea that the output of the RO is independent of its input, and that one can verify whether the output (for now, up to one bit) of an RO is correctly generated from a specific input. While VBB security is impossible for general functions [5], VBB secure obfuscation appears possible for point functions. (Similar techniques are used to obfuscate wildcards, conjunctions, and hyperplanes [24, 18, 12, 41, 6, 32].)

Next Step – Nonmalleability. However, there are many other properties of the RO that make it a desirable object. For example, given an RO output value on input x, it should be infeasible to obtain another output of RO on any related input point (e.g., $x + 1$). Applied to our setting, this is known as *nonmalleable point obfuscation*. The nonmalleability of random oracles enables many other objects that resist active attack. For example, this work considers robust fuzzy extractors [16] as an application, which were first constructed from random oracles.

Canetti and Varia [25] defined a nonmalleable point function and realized it in the common reference string (CRS) model. However, as one of the most valuable properties of the RO is that no trusted setup is required, an ideal instantiation would not require a CRS.

To tackle this, Komargodski and Yogev [46] proposed a construction of a nonmalleable point obfuscation in the plain model.[1] Prior work in plain model point obfuscation considers a limiting tampering class of low-degree polynomials where the degree relates to the hardness of the underlying number-theoretic assumption.

Another Step Forward – Digital Lockers. An obfuscated point function only outputs one bit. However, we are generally interested in the RO outputting a random string for a given input. To emulate this functionality, a natural extension is the *multi-bit* point function, where each function $I_{\mathsf{val},\mathsf{key}}$ is indexed by a pair of strings $(\mathsf{val}, \mathsf{key})$ and works as follows:

$$I_{\mathsf{val},\mathsf{key}}(\mathsf{val}') = \begin{cases} \mathsf{key} & \mathsf{val} = \mathsf{val}' \\ \perp & \text{otherwise} \end{cases}.$$

An obfuscation of a function of this class is called a *digital locker*, which is useful in password [20] and biometric authentication [1, 22].

Though we know how to build digital lockers in the plain model [21], the only existing nonmalleable constructions require a CRS. Fenteany and Fuller [39] achieved half of the goal, constructing a digital locker that nonmalleabile against tampering only on val in the standard model. However, while the work [39] pointed out a technique to additionally protect key, it required a CRS, similarly to the original

[1] Unfortunately, their underlying cryptographic assumption was broken by Bartusek, Ma, and Zhandry [7]. An alternative assumption was posed in [47], but this did not suffice to show security. Fortunately, Bartusek, Ma, and Zhandry introduced their own assumption and accompanying construction, showing their assumption holds in a strong variant of the generic group model [7].

work [25]. As an ideal instantiation of RO does not require a trusted setup, this naturally motivates our main question:

Can one build a nonmalleable digital locker in the plain model without setup?

Our Technical Contributions We answer the main question in the affirmative, constructing a nonmalleable digital locker in the plain model. We present the following contributions:

1. **Point Obfuscation with Associated Data** We define a new primitive called a *nonmalleable point obfuscator with associated data*. We then instantiate this object using group assumptions introduced by Bartusek, Ma, and Zhandry [7].
2. **Creating a Multibit Output** We then integrate this construction with the real or random construction [21], yielding a nonmalleable digital locker that prevents tampering on the input and associated data only. This step is not black box in the point obfuscations. Instead, it is created from scratch using similar techniques from the same group assumptions as the constructed point obfuscation.
3. **Protecting the Multibit Output** By putting the CRS of a true simulation extractable noninteractive zero-knowledge proof (NIZK) [33] into the associated data, we can protect the output of the digital locker. Conceptually, our new tool protects the NIZK `crs`, which (if intact) can be used to derive nonmalleability for the other parts of the construction. This step is black box from an appropriate variant of a digital locker.

In all of the above steps, the prevented tampering class for the input point, val is low-degree polynomials, rather than the desired complete tamper resistance. However, this class is still meaningful in many applications where a RO was previously used.

1.1 Low Entropy Robust Fuzzy Extractors in the Plain Model

Despite a limited tampering class, our nonmalleable objects suffice to construct the first plain model robust fuzzy extractors [16] that support sources whose entropy is less than half their length $1/2$, a known barrier for information-theoretically secure constructions [36]. We notice that all prior computationally secure constructions relied on some form of a CRS, and our work shows that this component is not required.

A fuzzy extractor is a pair of algorithms $(\mathsf{Gen}, \mathsf{Rep})$ with two properties:

Correctness. Let w, w' be values that are close in some distance metric, and define $(\mathsf{key}, \mathsf{pub}) \leftarrow \mathsf{Gen}(w)$. Then it is true that $\mathsf{Rep}(w', \mathsf{pub}) = \mathsf{key}$.

Security. The value key is computationally indistinguishable from a uniform value given pub.

Digital lockers have been used to construct reusable fuzzy extractors, as in [52] [22, 1], i.e., one can derive multiple keys from the same entropy source. An additional desirable property is robustness [34], which prevents an adversary from modifying pub in an attempt to force Rep to produce a different key.

Robust fuzzy extractors are notoriously difficult to construct – we show various limitations of the prior constructions in Table 1. Dodis and Wichs [36] showed that it is only possible information-theoretically if the entropy of w is at least half its length. Feng and Tang [38] showed this barrier exists in the CRS

Scheme	Model	Security	SS errors	$\mathrm{H}_\infty(W) < \lvert W\rvert/2$?
[14, 15]	RO	IT	t	✓
[34]	Plain	IT	t	X
[30]	CRS	IT	t	X
[57, 55, 56]	CRS	Comp.	$2t$	✓
[38]	CRS*	Comp.	$2t$	✓
Syn. + NM Point Obf w/ Assoc. Data	Plain	Comp.	$2t$	✓
Syn. + NM Digital Locker	Plain	Comp.	$2t$	✓

Table 1: Comparison of Robust Fuzzy Extractors. The CRS* model means that the distribution of W can depend on the CRS, however, the CRS is still assumed not to be modified. For a distribution W, $\mathrm{H}_\infty(W)$ represents min-entropy (see Section 2) and $|W|$ represents its length. IT corresponds to information-theoretic security and Comp. represents security against computationally bounded adversaries. Syn. is the syndrome or null space of an appropriate error correcting code. The column of SS errors indicates the error tolerance of the underlying secure sketch. This parameter is related to the information leakage of the secure sketch. This work and prior computational works require a secure sketch that corrects $2t$ errors, which leads to more leakage.

model, as well. Feng and Tang construct a robust fuzzy extractor with computational security for entropy sources that can depend on the CRS.

We construct the first robust fuzzy extractor in the plain model that supports entropy for w that is less than half its length. We combine our nonmalleable digital locker with a specific error-correction component, the syndrome construction [11, 31, 35]. The syndrome construction allows the reduction to extract a low-degree polynomial that is consistent with the adversary's tampering. Similar techniques were used to construct CRS model robust fuzzy extractors from algebraic-manipulation detection codes [30]. We present a second construction directly from the nonmalleable point function from associated data which is able to extract a limited length key.[2]

To the best of our knowledge, our work and that of Cramer et al. [30] are the only two approaches to building a robust fuzzy extractor that do not build a robust extractor first. This is because our nonmalleable tools only prevent limited tampering classes; both works use the secure sketch component to guarantee the adversary's tampering is in this low complexity class.

1.2 Technical Overview

In this section, we present an overview of our techniques. In the CRS model, nonmalleability of point functions can be achieved as [25], by using a nonmalleable NIZK system – in addition to generating a regular $C \leftarrow \mathsf{DL}(I_{\mathsf{val},\mathsf{key}})$, one also appends a zero-knowledge proof π to the output showing knowledge of the pair $(\mathsf{val}, \mathsf{key})$ inside C. However, any non-trivial nonmalleable NIZK system would require a trusted (nontamperable) CRS for security of the proof system, so the overall obfuscation would be $(\mathtt{crs}, C, \pi)$. Without trusted setup, an adversary may simply replace the $\mathtt{crs}$, rendering the NIZK ineffective and breaking nonmalleability. So, this trusted setup required immediately fails at achieving our goal.

1.2.1 Point Obfuscation with Associated Data

To achieve our goal, we formalize a notion that blends any public string with point obfuscation in a meaningful way, called *point obfuscation with associated data*. More specifically, the obfuscator $\mathsf{Obf}(I_{\mathsf{val}}, \mathtt{ad})$

[2] We also show that a nonmalleable point function (without associated data) suffices to construct a robust secure sketch [35].

takes as input the point function I_{val} and an additional public string $\mathtt{ad}$ (e.g., $\mathtt{crs}$) and then outputs an obfuscated program C along with $\mathtt{ad}$. The output program C should be VBB secure, and $\mathtt{ad}$ is treated as public information.

We formulate nonmalleability properties that treat the two inputs quite differently. The adversary outputs $(C', \mathtt{ad}', f)$ and wins if C' is consistent with the values $f(\mathsf{val})$ and $\mathtt{ad}'$, and one of the following hold:

1. The function belongs to some targeted function class, i.e., $f \in \mathcal{F}$, or
2. The function f is the identity and $\mathtt{ad}' \neq \mathtt{ad}$.

Nonmalleability requires that the adversary has only a negligible winning probability, meaning that they cannot replace $\mathtt{ad}$ by any other string, nor tamper val consistently by any function in the class $\mathcal{F}$.

Remark 1. *It is undesirable that in the definition the adversary output their tampering function. The desired notion is that the adversary cannot output $(C', \mathtt{ad}')$ that is consistent with any f. This notion is impossible to achieve in the plain model if f contains linear shifts. Essentially, given an obfuscation of point x, an adversary not required to output their mauling function may simply create an obfuscation of independent point y. It is clear that if the function $f(z) = z - x + y$ is in $\mathcal{F}$, this would be a valid tampering, but it is impossible to prevent without requiring the adversary shows some awareness of its specific tampering. The definition where the adversary chooses and outputs f after seeing C does imply that all fixed functions f are prevented [47]. See Appendix A for more details.*

How to Construct this Object Before instantiating such an object, we recall some notations and related constructions of nonmalleable point obfuscations in prior work [46, 7, 39]. We note that all of these constructions rely on groups that only efficiently admit linear operations.

Suppose that g is a generator of a prime order group whose order is p. Throughout this paper, $[x]_g$ will be used to represent g^x (called implicit notation in [37]) so as to highlight the behavior in the exponent. We treat val as an element in $\mathbb{Z}_p$. Let the class of tampering functions $\mathcal{F}$ correspond to low degree polynomials over $\mathbb{Z}_p$. Previous constructions [7] use a set of polynomial encodings, denoted as $\mathcal{P}$, and compute the following for $\mathsf{Obf}(I_{\mathsf{val}})$:

1. Sample some $P \leftarrow \mathcal{P}$,
2. Output $P, [P(\mathsf{val})]_g$.

The intuition for security[3] is twofold: 1) that $\mathcal{P}$ is sufficiently randomized to argue virtual black box security [5], and that 2) for all instances of $P \in \mathcal{P}$ no fixed affine functions of P — i.e., $\alpha P(\mathsf{val}) + \beta$ — correspond to any $P'(f(\mathsf{val}))$ for $P' \in \mathcal{P}$ and low degree polynomial f. Prior work achieves these two properties jointly by randomizing the low degree coefficients of P and fixing some higher powers to have a coefficient of 1. For example, Bartusek et al. [7] consider $P_a(x) = ax + x^2 + x^3 + x^4 + x^5$.

Our construction builds such a function class $\mathcal{P}$, parameterizing $P \in \mathcal{P}$ by both a random a and $\mathtt{ad}$, so that $\mathtt{ad}$ and val can be blended in a secure way. Let $\rho := |\mathtt{ad}|$. Then, we have:

$$P_{a,\mathtt{ad}}(x) \stackrel{\text{def}}{=} ax + \sum_{i=2}^{\rho+1} \mathtt{ad}_i x^i + \sum_{i=\rho+2}^{\rho+6} x^i.$$

[3]The actual constructions are more complicated to ensure correctness holds, using other points of randomness and group elements to check correctness. These are not used in arguing nonmalleability. For simplicity, we do not discuss correctness in this section.

Algorithm 1: Augmented real-or-random construction that provides nonmalleability over input point and associated data. Obf is an obfuscator and NMObf is a nonmalleable obfuscator.

Input $(\mathsf{val}, \mathsf{key}, \mathtt{ad})$.
Sample random z.
for *each bit* i *of* key **do**
  if $\mathsf{key}_i = 1$ **then**
    $C_i \leftarrow \mathsf{Obf}(I_{\mathsf{val}})$
  else
    $C_i \leftarrow \mathsf{Obf}(I_z)$
  end
end
Let $C_0 \leftarrow \mathsf{NMObf}(I_{\mathsf{val},\mathtt{ad}})$. `// To distinguish an all-zero key and provide nonmalleability`
Output $C = (C_0, C_1, \ldots, C_{|\mathsf{key}|}, \mathtt{ad})$.

In the above, the random a corresponds to the lowest degree coefficient of P and the bits of $\mathtt{ad}$ set intermediate coefficients of the polynomial P. We can prove security using the same group assumption used in prior nonmalleable point obfuscation works [7, 39].

While the construction has a similar structure to prior work, analysis of nonmalleability is significantly more complicated by the fact that the adversary 1) knows $\mathtt{ad}$, 2) can output any value for $\mathtt{ad}'$, and 3) doesn't have to explain how $\mathtt{ad}'$ arose from $\mathtt{ad}$. This gives the adversary more flexibility, and proving nonmalleability becomes a careful multi-step procedure.

To give some intuition for the algebraic structure, it is important that the powers multiplied by the bits of $\mathtt{ad}$ are below the powers with coefficients 1. If these were switched, one could apply a polynomial tampering function to x and change the associated data to compensate for the resulting changes in the higher powers. Another natural approach is to obfuscate the string $x||\mathtt{ad}$ using a nonmalleable point obfuscation. In addition to the difficulties mentioned above, the algebraic structure is unclear. The concatenation operation would occur on strings before mapping to $\mathbb{Z}_p^*$. However, the tampering function $f : \mathbb{Z}_p^* \to \mathbb{Z}_p^*$ changes elements. The corresponding tampering function on strings depends on the mapping from strings to $\mathbb{Z}_p^*$.

1.2.2 Extending to the Multibit Setting

Next, we integrate the above with the real-or-random approach of Canetti and Dakdouk [21]. The modified algorithm is summarized in Algorithm 1.

On the technical side, this approach requires the polynomials in the group to have more randomized powers, similar to the prior work of Fenteany and Fuller [39]. However, unlike their work, we only use one nonmalleable point obfuscation, the rest simply provide privacy. That is, only C_0 in Algorithm 1 is nonmalleable. As we show, this is sufficient to ensure nonmalleability over the resultant digital locker.

1.2.3 Protecting the Multibit Output

The above instantiation of the real-or-random construction prevents tampering of the input point and associated data but provides no protection over key. Our protection of the associated data allows us to upgrade the NIZK construction of [25] to the plain model. Our technique protects the associated data,

which is set as `crs`, and the security of NIZK protects everything else, so long as `crs` cannot be tampered with. As we discuss in Section 5.3, we are also able to use a weaker NIZK system, specifically true simulation extractible NIZKs, which may be instantiable in pairing-free groups.

1.3 Discussion and Open Questions

This work presents the first constructions in the plain model of nonmalleable digital lockers and low-entropy robust fuzzy extractors. The integration of the nonmalleable point function with associated data with the real-or-random construction is technical and non-black box. Ideally, one would be able to define some necessary condition such that general black box composition of our point obfuscation with associated data and any other point obfuscation or digital locker is possible. One can view our construction as evidence that our particular nonmalleable point obfuscator with associated data is safe under composition with a specific point function.

There are known barriers to constructing digital lockers secure against auxiliary data that is hard to invert (such as a point function) if indistinguishability obfuscation exists [19, 10]. Security in the presence of auxiliary data is the standard method for arguing composition.

In this work, we focus on nonmalleability of digital lockers. Obfuscating wildcards, conjunctions, and hyperplanes use similar techniques [18, 12, 41, 6, 32, 24], so our techniques may apply. We note that some of these objects directly yield non-robust fuzzy extractors [41, 32], so it may be possible to provide robustness by making the obfuscation nonmalleable. It seems less likely the techniques can be used to protect obfuscation of general evasive functions [4], compute-and-compare programs [45, 58, 17] and general obfuscation [42, 43, 51, 50, 44, 2].

We generically use (true simulation extractible) NIZKs. Optimizing this construction is important, since this object will likely represent the dominant computational cost.

2 Preliminaries

Logarithms are base 2. Let $X_i \in \mathcal{Z}$ be random variables. We denote by $X = X_1, ..., X_n$ the tuple $(X_1, \ldots, X_n)$. For a discrete random variable X, the *min-entropy* of X is $\mathrm{H}_\infty(X) = -\log(\max_x \Pr[X = x])$. For a pair of discrete random variables X, Y, the average min-entropy of $X|Y$ is

$$\tilde{\mathrm{H}}_\infty(X|Y) = -\log\left(\mathop{\mathbb{E}}_{y\in Y}\left(2^{-\mathrm{H}_\infty(X|Y)}\right)\right).$$

The notation id is used to denote the identity function: $\forall x, \mathsf{id}(x) = x$. Capitalized letters are used for random variables and lowercase letters for samples. Let $\{\mathsf{D}_\lambda\}$ be an ensemble of sets. Two circuits, C and C', with inputs in D^λ are *functionally equivalent*, denoted $C \equiv C'$, if $\forall x \in \mathsf{D}^\lambda, C(x) = C'(x)$. For a matrix $\mathbf{A}$, let $\mathbf{A}_i$ denote the ith row and $\mathbf{A}_{i,j}$ to denote the entry in the i row and jth column.

Definition 2.1. *An ensemble of distributions $\mathcal{X} = \{X_\lambda\}_{\lambda\in\mathbb{N}}$, where $\mathcal{X}_\lambda$ is over D^λ, is* well-spread *if the function $\mathrm{H}_\infty(X_\lambda)$ mapping λ to non negative reals grows faster than $\omega(\log\lambda)$. That is, $\mathrm{H}_\infty(X_\lambda) = \omega(\log\lambda)$.*

Definition 2.2. *An ensemble of distributions $\mathcal{X} = \{X_\lambda\}_{\lambda\in\mathbb{N}}$, where $\mathcal{X}_\lambda$ is over D^λ, is* efficiently sampleable *if exists a PPT algorithm given 1^λ as input whose output is identically distributed as X_λ.*

Throughout this work, we will use λ to represent the security parameter, ρ to represent the length of the associated data, ℓ to represent the length of the output key, and τ to represent the maximum degree of the polynomial the adversary uses for mauling.

3 Obfuscation Definitions

All obfuscation definitions include require only polynomial slowdown, which is easily verifiable for all presented constructions. The main object we introduce in this work is a nonmalleable point function with associated data. A traditional point function $I_{\mathsf{val}} : \mathbb{Z}_p \mapsto \{0,1\}$ takes a single input $\mathsf{val} \in \mathbb{Z}_p$ and returns 1 if and only if the input x to the function is val. An obfuscator is designed to preserve this functionality while hiding val. The definition of a nonmalleable point function with associated data adds a second input to I denoted as $\mathtt{ad} \in \{0,1\}^\rho$. This input does not need to be hidden by the obfuscator but should be nonmalleable. So the raw functionality is just a point function of the pair $\mathsf{val}, \mathtt{ad}$. That is,

$$I_{\mathsf{val},\mathtt{ad}}(x,y) = \begin{cases} 1 & x = \mathsf{val} \wedge y = \mathtt{ad} \\ 0 & \text{otherwise.} \end{cases}$$

Note that, since in our use cases $\mathtt{ad}$ is public, an honest user may just use the given $\mathtt{ad}$ in using the obfuscated point function. In our further sections, we use $\mathsf{lockPoint}(\cdot)$ to denote point obfuscation algorithm and $\mathsf{unlockPoint}$ as the obfuscated program. As prior work [25, 46, 39], we first present the notion of an obfuscation verifier:

Definition 3.1 (Obfuscation Verifier)**.** *Let $\lambda \in \mathbb{N}$ be a security parameter and let $\mathcal{O}$ input $x \in \mathsf{D}^\lambda$ and output a program $\mathcal{P}$. An algorithm $\mathsf{V}_{\mathtt{obf}}$ is a value verifier if $\forall x \in \mathsf{D}^\lambda$ it is true that*

$$\Pr_{\mathsf{V}_{\mathtt{obf}},\mathcal{O}}[\mathsf{V}_{\mathtt{obf}}(\mathcal{P}) = 1 | \mathcal{P} \leftarrow \mathcal{O}(x)] = 1.$$

Definition 3.2 (Nonmalleable Point Function with Associated Data)**.** *For security parameter $\lambda \in \mathbb{N}$ parameter $\rho \in \mathbb{N}$, let D_λ be a sequence of input domains and $\mathcal{F} : \mathsf{D}_\lambda \to \mathsf{D}_\lambda$ be a family of functions. Let $\mathcal{X}$ be a family of distributions over D_λ. A $(\mathcal{F}, \mathcal{X}, \rho)$*-nonmalleable point function obfuscation with associated data $\mathsf{lockPoint}$ *is a PPT algorithm that inputs a point $\mathsf{val} \in \mathsf{D}_\lambda$ and $\mathtt{ad} \in \{0,1\}^\rho$, and outputs a circuit $\mathsf{unlockPoint}$. Let $\mathsf{V}_{\mathtt{obf}}$ be an obfuscation verifier for $\mathsf{lockPoint}$ as defined in Definition 3.1. The following properties must hold:*

1. ***Completeness**: For all $\mathsf{val} \in \mathsf{D}^\lambda, \mathtt{ad} \in \{0,1\}^\rho$, it holds that*

$$\Pr[\mathsf{unlockPoint}(\cdot,\cdot) \equiv I_{\mathsf{val},\mathtt{ad}}(\cdot,\cdot) | \mathsf{unlockPoint} \leftarrow \mathsf{lockPoint}(\mathsf{val},\mathtt{ad})] \geq 1 - \mathtt{ngl}(\lambda),$$

   where the probability is over the randomness of $\mathsf{lockPoint}$.

2. ***Virtual Black Box Security**: For every PPT $\mathcal{A}$ and any polynomial function p, there exists a simulator $\mathcal{S}$ and a polynomial function $q(\cdot)$ such that, for all large enough $\lambda \in \mathbb{N}$, all $\mathsf{val} \in \mathsf{D}^\lambda, \mathtt{ad} \in \{0,1\}^\rho$ and for any predicate $\mathcal{P} : \mathsf{D}^\lambda \times \{0,1\}^\rho \mapsto \{0,1\}$,*

$$\begin{aligned} &|\Pr[\mathcal{A}(\mathsf{unlockPoint}, \mathtt{ad}) = \mathcal{P}(\mathsf{val}, \mathtt{ad}) | \mathsf{unlockPoint} \leftarrow \mathsf{lockPoint}(\mathsf{val}, \mathtt{ad})] \\ &- \Pr[\mathcal{S}^{I_{\mathsf{val},\mathtt{ad}}(\cdot)}(1^\lambda, \mathtt{ad}) = \mathcal{P}(\mathsf{val}, \mathtt{ad})]| \leq \frac{1}{p(\lambda)}, \end{aligned}$$

   where S is allowed $q(\lambda)$ oracle queries total to $I_{\mathsf{val},\mathtt{ad}}$ and the probabilities are over the internal randomness of $\mathcal{A}$ and $\mathsf{lockPoint}$, and of $\mathcal{S}$, respectively. Here $I_{\mathsf{val},\mathtt{ad}}(\cdot)$ is an oracle that returns 1 when provided input $(\mathsf{val}, \mathtt{ad})$ and 0 otherwise.

3. ***Nonmalleability:*** *For any $X \in \mathcal{X}$, for all $\mathsf{ad} \in \{0,1\}^\rho$, for any PPT $\mathcal{A}$, there exists $\epsilon = \mathtt{ngl}(\lambda)$, such that defining*

$$\begin{aligned}\mathsf{unlockPoint} &\leftarrow \mathsf{lockPoint}(\mathsf{val}, \mathsf{ad}),\\ (C, f, \mathsf{ad}^*) &\leftarrow \mathcal{A}\,(\mathsf{unlockPoint}, \mathsf{ad})\end{aligned}$$

it is true that :

$$\Pr_{\mathsf{val}\leftarrow X}\begin{bmatrix}\mathsf{V}_{\mathtt{obf}}(C) = 1, (I_{f(\mathsf{val}),\mathsf{ad}^*} \equiv C)\\ f \in \mathcal{F} \ \vee (f = \mathsf{id} \wedge \mathsf{ad}^* \neq \mathsf{ad})\end{bmatrix} \leq \epsilon.$$

3.1 Nonmalleable Digital Locker

We recall the definition of a nonmalleable digital locker. To distinguish this from the case of point obfuscation, we use $\mathsf{lock}()$ to denote the multi-bit point obfuscation algorithm and unlock as the (obfuscated) digital locker. In our construction, all tampering of the output key is prevented, so we remove the notion of a key verifier that was used in [39].

Definition 3.3 (Nonmalleable Digital Locker)**.** *For security parameter $\lambda \in \mathbb{N}$, let D_λ be a sequence of domains, let*

1. *$\mathcal{F} : \mathsf{D}_\lambda \to \mathsf{D}_\lambda$ be a function family,*
2. *$\mathcal{X}$ be a family of distributions over D^λ,*
3. *lock be a PPT algorithm that maps points $\mathsf{val} \in \mathsf{D}^\lambda, \mathsf{key} \in \{0,1\}^n$ to a circuit unlock, and*
4. *$\mathsf{V}_{\mathtt{obf}}$ be an obfuscation verifier.*

The algorithm lock is a $(\mathcal{F}, \mathcal{X}, n)$-nonmalleable digital locker *if all of the below are satisfied:*

1. ***Completeness*** *For all $\mathsf{val} \in \mathsf{D}^\lambda, \mathsf{key} \in \{0,1\}^n$ it holds that*

$$\Pr[\mathsf{unlock}(\cdot) \equiv I_{\mathsf{val},\mathsf{key}}(\cdot) | \mathsf{unlock} \leftarrow \mathsf{lock}(\mathsf{val}, \mathsf{key})] \geq 1 - \mathtt{ngl}(\lambda),$$

   where the probability is over the randomness of lock. Here $I_{\mathsf{val},\mathsf{key}}$ is a function that returns key when provided input val, otherwise $I_{\mathsf{val},\mathsf{key}}$ returns $\perp$.

2. ***Virtual Black Box Security****: For all PPT $\mathcal{A}$ and $p = \mathtt{poly}(\lambda)$, $\exists \mathcal{S}$ and $q(\lambda) = \mathtt{poly}(\lambda)$ such that for all large enough $\lambda \in \mathbb{N}$, $\forall \mathsf{val} \in \mathsf{D}_\lambda, \mathsf{key} \in \{0,1\}^n, \mathcal{P} : \mathsf{D}_\lambda \times \{0,1\}^n \mapsto \{0,1\}$,*

$$\left|\Pr[\mathcal{A}(\mathsf{lock}(\mathsf{val}, \mathsf{key})) = \mathcal{P}(\mathsf{val}, \mathsf{key})] - \Pr[\mathcal{S}^{I_{\mathsf{val},\mathsf{key}}}(1^\lambda) = \mathcal{P}(\mathsf{val}, \mathsf{key})]\right| \leq \frac{1}{p(\lambda)},$$

   where S is allowed $q(\lambda)$ oracle queries to $I_{\mathsf{val},\mathsf{key}}$ and the probabilities are over the internal randomness of $\mathcal{A}$ and lock, and of $\mathcal{S}$, respectively.

3. ***Nonmalleability*** *$\forall X \in \mathcal{X}, PPT\ \mathcal{A}, \mathsf{key} \in \{0,1\}^n$, there exists $\epsilon = \mathtt{ngl}(\lambda)$ such that:*

$$\Pr_{\mathsf{val}\leftarrow X}\left[\begin{pmatrix}\mathsf{V}_{\mathtt{obf}}(C) = 1,\\ (f \in \mathcal{F} \wedge \mathsf{key}' \neq \perp) \vee\\ (\mathsf{key}' \notin \{\perp, \mathsf{key}\} \wedge f = \mathsf{id})\end{pmatrix} \middle| \begin{matrix}\mathsf{unlock}_{\mathsf{val},\mathsf{key}} \leftarrow \mathsf{lock}(\mathsf{val}, \mathsf{key})\\ (C, f) \leftarrow \mathcal{A}(\mathsf{unlock}_{\mathsf{val},\mathsf{key}})\\ \mathsf{key}' \leftarrow C(f(\mathsf{val}))\end{matrix}\right] \leq \epsilon.$$

   recall id is the identity function.

Remark 2. *As mentioned in the Introduction, there are alternative notions of nonmalleability. We formally define fixed nonmalleability, a weaker definition which was used in [25], and oblivious nonmalleability, which does not require the adversary to output the targeted function f. In that appendix, we refer to the above nonmalleability definition as* adaptive. *There we show that oblivious nonmalleability is impossible in general. One can bypass this result by using cryptographic tools that extract the tampering function, such as a random oracle or non-falsifiable assumptions.*

3.2 Same Point Definitional Equivalences

The soundness in Definitions 3.2 and 3.3 are virtual black box security [5]. In the majority of this work, we will be using distributional indistinguishability, which says that obfuscations of all well spread distributions X are indistinguishable from obfuscations of random points. Bitanski and Canetti [13] showed that this definition is equivalent to virtual black box obfuscation for point functions (see also [20, 54]). Furthermore, they showed this equivalence holds when given a constant number of obfuscations on related points. Fenteany and Fuller [39] show that this equivalence holds if given a polynomial number of copies $\mathsf{unlockPoint}_1 \leftarrow \mathsf{lockPoint}(X), ..., \mathsf{unlockPoint}_\ell \leftarrow \mathsf{lockPoint}(X)$ as long as the same value is locked in each call to $\mathsf{lockPoint}$. We generalize these results showing that a vector of obfuscations that have output on a single input point are secure when composed with associated data. That is, define the circuit class

$$\mathbf{Point}_{\mathsf{val},\mathsf{key},\mathsf{ad}}(\mathsf{val}', \mathsf{ad}') = \begin{cases} \mathsf{key} & \mathsf{val}' = \mathsf{val} \wedge \mathsf{ad}' = \mathsf{ad} \\ \bot & \text{otherwise} \end{cases}.$$

Note that point functions and digital lockers both with and without associated data variants fall into this class by adjusting whether ad and key are of length 0. These proofs are straightforward extensions of the proofs in [39].

Definition 3.4 (Distributional Indistinguishability)**.** *A* **Point** *obfuscator is called a good distributional indistinguishable (DI) obfuscator if for any PPT $\mathcal{A}$ with binary output and any well-spread distribution $\mathcal{X}$ over points in D^λ, for all vectors $\vec{\mathsf{key}}, \vec{\mathsf{ad}}$ then there exists some negligible function ϵ such that*

$$\begin{aligned} &|\Pr_{\mathsf{val}\leftarrow\mathcal{X}}[\mathcal{A}(\{C_i, \vec{\mathsf{ad}}_i\}_{i=1}^\ell) = 1 | \{C_i \leftarrow \mathbf{Point}(\mathsf{val}, \vec{\mathsf{ad}}_i, \vec{\mathsf{key}}_i)\}_{i=1}^\ell] \\ &- \Pr_{u\xleftarrow{\$}\mathsf{D}^\lambda}[\mathcal{A}(\{C_i, \vec{\mathsf{ad}}_i\}_{i=1}^\ell) = 1 | \{C_i \leftarrow \mathbf{Point}(u, \vec{\mathsf{ad}}_i, \vec{\mathsf{key}}_i)\}_{i=1}^\ell]| \leq \epsilon. \end{aligned}$$

Theorem 3.1. *For the class* **Point** *under $\ell = \mathtt{poly}(\lambda)$ composition where the same val is used in each obfuscation, distributional indistinguishability and virtual black box security (in Definition 3.2) are equivalent.*

This equivalence works through an intermediate definition known as oracle indistinguishability.

Definition 3.5 (Oracle Indistinguishability)**.** *A* **Point***obfuscator is a oracle indistinguishable (OI) obfuscator if for any PPT adversary $\mathcal{A}$ and any polynomial function p, there exists a polynomial size family of sets $\{L_\lambda\}_{\lambda\in\mathcal{N}}$ for all $\mathsf{val} \notin L_\lambda$, such that for all $\vec{\mathsf{key}}, \vec{\mathsf{ad}}$ for all sufficiently large λ,*

$$\begin{aligned} &|\Pr[\mathcal{A}(\{C_i, \vec{\mathsf{ad}}_i\}_{i=1}^\ell) = 1 | \{C_i \leftarrow \mathbf{Point}(\mathsf{val}, \vec{\mathsf{ad}}_i, \mathsf{key}_i)\}_{i=1}^\ell] \\ &- \Pr_{\mathsf{val}'\xleftarrow{\$}\mathsf{D}^\lambda}[\mathcal{A}(\{C_i, \vec{\mathsf{ad}}_i\}_{i=1}^\ell) = 1 | \{C_i \leftarrow \mathbf{Point}(\mathsf{val}', \vec{\mathsf{ad}}_i, \mathsf{key}_i)\}_{i=1}^\ell]| \leq \frac{1}{p(\lambda)}. \end{aligned}$$

Proof of Theorem 3.1. We show Theorem 3.1 in three steps. All of these steps assume the same ℓ for each definition.

Lemma 3.1. *If* **Point** *satisfies Definition 3.4, then it satisfies Definition 3.5.*

Proof of Lemma 3.1. Consider some binary PPT $\mathcal{A}$ and polynomial function p. We proceed by contradiction. Let X_λ be the set of all values $\mathsf{val} \in \mathcal{X}_\lambda$ such that $\exists \vec{\mathsf{key}}, \vec{\mathsf{ad}}$ and for all sufficiently large λ

$$\begin{aligned} &|\Pr[\mathcal{A}(\{C_i, \vec{\mathsf{ad}}_i\}_{i=1}^{\ell}) = 1 | \{C_i \leftarrow \mathbf{Point}(\mathsf{val}, \vec{\mathsf{ad}}_i, \mathsf{key}_i)\}_{i=1}^{\ell}] \\ &- \Pr_{\mathsf{val}' \xleftarrow{\$} \mathsf{D}^\lambda}[\mathcal{A}(\{C_i, \vec{\mathsf{ad}}_i\}_{i=1}^{\ell}) = 1 | \{C_i \leftarrow \mathbf{Point}(\mathsf{val}', \vec{\mathsf{ad}}_i, \mathsf{key}_i)\}_{i=1}^{\ell}]| > \frac{1}{p(\lambda)}. \end{aligned} \tag{1}$$

Fix some such values $\vec{\mathsf{key}}$ and $\vec{\mathsf{ad}}$. Define X_λ^+ as the set of points where Equation 1 is true without the absolute values and X_λ^- as the set of points where the negation is true without the absolute values. Assume towards a contradiction that $|X_\lambda| = \omega(\mathtt{poly}(\lambda))$. Then it must be case for either X_λ^+ or X_λ^-. Assume that without loss of generality that $|X_\lambda^+| = \omega(\mathtt{poly}(\lambda))$. Define Z_λ as the uniform distribution over points in X_λ^+. Then it must be the case that

$$\begin{aligned} &|\Pr_{\mathsf{val} \leftarrow \mathcal{Z}_\lambda}[\mathcal{A}(\{C_i, \vec{\mathsf{ad}}_i\}_{i=1}^{\ell}) = 1 | \{C_i \leftarrow \mathbf{Point}(\mathsf{val}, \vec{\mathsf{ad}}_i, \mathsf{key}_i)\}_{i=1}^{\ell}] \\ &- \Pr_{\mathsf{val}' \xleftarrow{\$} \mathsf{D}^\lambda}[\mathcal{A}(\{C_i, \vec{\mathsf{ad}}_i\}_{i=1}^{\ell}) = 1 | \{C_i \leftarrow \mathbf{Point}(\mathsf{val}', \vec{\mathsf{ad}}_i, \mathsf{key}_i)\}_{i=1}^{\ell}]| > \frac{1}{p(\lambda)}. \end{aligned}$$

This contradicts the definition of distributional indistinguishability. Thus it must be the case that $|X_\lambda| = \mathtt{poly}(\lambda)$. This completes the proof of Lemma 3.1. □

Lemma 3.2. *If* **Point** *satisfies Definition 3.5 then it satisfies VBB security (soundness in Definition 3.2).*

Proof of Lemma 3.2. Let $\mathcal{A}$ be some PPT adversary. Suppose that **Point** satisfies Definition 3.5 oracle indistinguishability when composed ℓ times. Let L_λ be the polynomial size set in Definition 3.5. Define the simulator $S_{\mathcal{A}}(\vec{\mathsf{ad}})$:

1. Define $\vec{\mathsf{key}} = \perp^\ell$.
2. Query the first oracle $I^1_{\mathsf{val},\mathsf{ad}_1,\mathsf{key}_1}$ on each point $x \in L_\lambda$ with the value $\vec{\mathsf{ad}}_1$. If the oracle returns $y_1 \neq \perp$, set $\mathsf{val} = x, \mathsf{key}_1 = y_1$. Query oracles $I^i_{\mathsf{val},\vec{\mathsf{ad}}_i,\mathsf{key}_i}$ retrieving value y_i and set $\mathsf{key}_i = y_i$. If some value was returned go to next step.
3. If $\mathsf{val} = \perp$ set $\mathsf{val} \xleftarrow{\$} \mathsf{D}^\lambda$.
4. If $\vec{\mathsf{key}} = \perp^\ell$ uniformly sample $\vec{\mathsf{key}}$.
5. Run and output $\mathcal{A}(\{\mathbf{Point}(\mathsf{val}, \vec{\mathsf{ad}}_i, \mathsf{key}_i), \vec{\mathsf{ad}}_i\}_{i=1}^{\ell})$.

Note that L_λ may differ for each adversary and length λ. Fix some predicate $\mathcal{P}$. We now analyze the quality of $S_{\mathcal{A}}$ suppose that $\mathsf{val} \in L_\lambda$ then $S_{\mathcal{A}}$ outputs exactly the distribution $\mathcal{A}(\{\mathsf{unlockPoint}_i, \vec{\mathsf{ad}}_i, \mathsf{key}_i\}_{i=1}^{\ell})$

so the simulation is perfect. In the case when $\mathsf{val} \notin L_\lambda$ by the definition of oracle indistinguishability

$$\begin{aligned}
&\Pr[\mathcal{A}(\{C_i, \vec{\mathsf{ad}}_i\}_{i=1}^{\ell}) = \mathcal{P}(\mathsf{val}, \vec{\mathsf{key}}, \vec{\mathsf{ad}}) | \{C_i \leftarrow \mathbf{Point}(\mathsf{val}, \vec{\mathsf{ad}}_i, \mathsf{key}_i)\}_{i=1}^{\ell}] \\
&- \Pr[\mathcal{S}_{\mathcal{A}}^{I_{1,\mathsf{val},\vec{\mathsf{ad}}_1}(\cdot),\ldots,I_{\ell,\mathsf{val},\vec{\mathsf{ad}}_\ell}(\cdot)}(1^\lambda, \vec{\mathsf{ad}}_1, ..., \vec{\mathsf{ad}}_\ell) = \mathcal{P}(\mathsf{val}, \vec{\mathsf{key}}, \vec{\mathsf{ad}})] \leq \\
&|\Pr[\mathcal{A}(\{C_i, \vec{\mathsf{ad}}_i\}_{i=1}^{\ell}) = 1 | \{C_i \leftarrow \mathbf{Point}(\mathsf{val}, \vec{\mathsf{ad}}_i, \mathsf{key}_i), \mathsf{val} \notin L_\lambda\}_{i=1}^{\ell}] \\
- &\Pr_{\mathsf{val}' \xleftarrow{\$} \mathsf{D}^\lambda}[\mathcal{A}(\{C_i, \vec{\mathsf{ad}}_i\}_{i=1}^{t}) = 1 | \{C_i \leftarrow \mathbf{Point}(\mathsf{val}', \vec{\mathsf{ad}}_i, \mathsf{key}_i)\}_{i=1}^{\ell}]| \leq \frac{1}{p(\lambda)}.
\end{aligned}$$

Thus, in all cases for an arbitrary polynomial $p(\lambda)$ the VBB condition is satisfied. □

Lemma 3.3. *If* **Point** *satisfies VBB security (soundness in Definition 3.2) then it satisfies Definition 3.4.*

Proof of Lemma 3.3. We assume that **Point** is not distributional indistinguishable towards showing that **Point** is not VBB. That is, there exists some $\mathcal{A}$ and distribution $\mathcal{X}$ and vectors $\vec{\mathsf{ad}}, \vec{\mathsf{key}}$ such that for all polynomial functions $p(\lambda)$ it is true that

$$\begin{aligned}
&|\Pr_{\mathsf{val} \leftarrow \mathcal{X}}[\mathcal{A}(\{C_i, \vec{\mathsf{ad}}_i\}_{i=1}^{\ell}) = 1 | \{C_i \leftarrow \mathbf{Point}(\mathsf{val}, \vec{\mathsf{ad}}_i, \mathsf{key}_i)\}_{i=1}^{\ell}] \\
&- \Pr_{u \xleftarrow{\$} \mathsf{D}^\lambda}[\mathcal{A}(\{C_i, \vec{\mathsf{ad}}_i\}_{i=1}^{\ell}) = 1 | \{C_i \leftarrow \mathbf{Point}(u, \vec{\mathsf{ad}}_i, \mathsf{key}_i)\}_{i=1}^{\ell}]| > 1/p(\lambda).
\end{aligned}$$

Assume without loss of generality that the statement is true without the absolute values. Define the random variable $\gamma(\mathsf{val}) = \Pr[\mathcal{A}(\{C_i, \vec{\mathsf{ad}}_i\}_{i=1}^{\ell}) = 1 | \{C_i \leftarrow \mathbf{Point}(\mathsf{val}, \vec{\mathsf{ad}}_i, \mathsf{key}_i)\}_{i=1}^{\ell}]$. Then sort val by $\gamma(\mathsf{val})$. There must exist super polynomial size sets $\mathcal{X}_{\max}, \mathcal{X}_{\min}$ such that

$$\begin{aligned}
&|\Pr_{\mathsf{val} \leftarrow \mathcal{X}_{\max}}[\mathcal{A}(\{C_i, \vec{\mathsf{ad}}_i\}_{i=1}^{\ell}) = 1 | \{C_i \leftarrow \mathbf{Point}(\mathsf{val}, \vec{\mathsf{ad}}_i, \mathsf{key}_i)\}_{i=1}^{\ell}] \\
&- \Pr_{\mathsf{val} \leftarrow \mathcal{X}_{\min}}[\mathcal{A}(\{C_i, \vec{\mathsf{ad}}_i\}_{i=1}^{\ell}) = 1 | \{C_i \leftarrow \mathbf{Point}(\mathsf{val}, \vec{\mathsf{ad}}_i, \mathsf{key}_i)\}_{i=1}^{\ell}]| > 1/p(\lambda).
\end{aligned}$$

Define the following values:

$$\begin{aligned}
\mathcal{A}_{\max} &= \Pr_{\mathsf{val} \leftarrow \mathcal{X}_{\max}}[\mathcal{A}(\{C_i, \vec{\mathsf{ad}}_i\}_{i=1}^{\ell}) = 1 | \{C_i \leftarrow \mathbf{Point}(\mathsf{val}, \vec{\mathsf{ad}}_i, \mathsf{key}_i)\}_{i=1}^{\ell}] \\
\mathcal{A}_{\min} &= \Pr_{\mathsf{val} \leftarrow \mathcal{X}_{\min}}[\mathcal{A}(\{C_i, \vec{\mathsf{ad}}_i\}_{i=1}^{\ell}) = 1 | \{C_i \leftarrow \mathbf{Point}(\mathsf{val}, \vec{\mathsf{ad}}_i, \mathsf{key}_i)\}_{i=1}^{\ell}] \\
S_{\max} &= \Pr_{\mathsf{val} \leftarrow \mathcal{X}_{\max}}[\mathcal{S}^{I^1_{\mathsf{val},\mathsf{ad}_1}(\cdot),\ldots,I^\ell_{\mathsf{val},\mathsf{ad}_\ell}(\cdot)}(1^\lambda, \mathsf{ad}_1, ..., \mathsf{ad}_\ell) = \mathcal{P}(\mathsf{val}, \vec{\mathsf{key}})] \\
S_{\min} &= \Pr_{\mathsf{val} \leftarrow \mathcal{X}_{\min}}[\mathcal{S}^{I^1_{\mathsf{val},\mathsf{ad}_1}(\cdot),\ldots,I^\ell_{\mathsf{val},\mathsf{ad}_\ell}(\cdot)}(1^\lambda, \mathsf{ad}_1, ..., \mathsf{ad}_\ell) = \mathcal{P}(\mathsf{val}, \vec{\mathsf{key}})]
\end{aligned}$$

No simulator S can perform differently on the distributions $\mathcal{X}_{\max}$ and $\mathcal{X}_{\min}$ as for any polynomial sequence size set of queries L_λ the probability that $\Pr_{\mathsf{val} \leftarrow \mathcal{X}_{\max}}[\mathsf{val} \in L_\lambda] \leq \mathtt{ngl}(\lambda)$. The same holds for $\mathcal{X}_{\min}$. Thus, it must be the case that $S_{\max} - S_{\min} \leq \mathtt{ngl}(\lambda)$. The contradiction follows by the triangle inequality. That is,

$$1/p(\lambda) < \mathcal{A}_{\max} - \mathcal{A}_{\min} < (\mathcal{A}_{\max} - S_{\max}) + (S_{\max} - S_{\min}) + (S_{\min} - \mathcal{A}_{\min}).$$

This implies that either $\mathcal{A}_{\max} - S_{\max}$ or $S_{\min} - \mathcal{A}_{\min}$ is greater than $1/3p(\lambda)$. This completes the proof by noting that for any desired $1/3p(\lambda)$ one can find an appropriate $1/p(\lambda)$. This completes the proof of Lemma 3.3. □

The application of Lemma 3.1, 3.2, and 3.3 proves Theorem 3.1. □

3.3 Group Theoretic Assumptions

We present our underlying group-theoretic assumptions here. As a reminder, we use the implicit notation [37] to denote encoding in a group with generator g (where $[x]_g$ denotes g^x).

Assumption 3.1. *[7, Assumption 3] Fix some $\psi \in \mathbb{Z}^+$. Let $\mathcal{G} = \{\mathbb{G}_\lambda\}_{\lambda\in\mathbb{N}}$ be a group ensemble with efficient representation and operations where each $\mathbb{G}_\lambda$ is a group of prime order $p(\lambda) \in (2^\lambda, 2^{\lambda+1})$. Let $\{\mathcal{X}_\lambda\}$ be a family of well-spread distributions over $\mathsf{D}^\lambda = \mathbb{Z}_{p(\lambda)}$. Then for any PPT $\mathcal{A}$:*

$$\left|\Pr[\mathcal{A}(\{k_i, [k_i x + x^i]_g\}_{i\in[2,\ldots,\psi]} = 1] - \Pr[\mathcal{A}(\{k_i, [k_i r + r^i]_g\}_{i\in[2,\ldots,\psi]}\right| = \mathtt{ngl}(\lambda).$$

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

Bartusek, Ma, and Zhandry [7] justified Assumption 3.1 by showing it holds in the generic group model even if $\mathcal{X}_\lambda$ depends on g. This model of allowing a distribution to depend on g is related to the non-uniform generic group model [27]. Such an assumption is crucial to arguing plain model security (rather than treating $\mathcal{X}_\lambda$ as independent of g). The second assumption can be proved from Assumption 3.1, see [7, Lemma 8], and is useful for arguing nonmalleability:

Assumption 3.2. *[7, Assumption 4] Fix some $\psi \in \mathbb{Z}^+$. Let $\mathcal{G}$ and $\mathcal{X}_\lambda$ be defined as in Assumption 3.1. For any PPT $\mathcal{A}$,*

$$\Pr[[x]_g \leftarrow \mathcal{A}(\{k_i, [k_i x + x^i]_g\}_{i\in[2,..,\psi]})] = \mathtt{ngl}(\lambda).$$

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

4 Nonmalleable Point Functions with Associated Data

We begin by instantiating a nonmalleable point obfuscation satisfying Definition 3.2.

Construction 4.1. *Let $\lambda \in \mathbb{N}$ be a security parameter, let $\rho \in \mathbb{N}$ be a parameter. Let $\mathcal{G} = \{\mathbb{G}_\lambda\}_{\lambda\in\mathbb{N}}$ be a group ensemble with efficient representation and operations where each $\mathbb{G}_\lambda$ is a group of prime order $p(\lambda) \in (2^\lambda, 2^{\lambda+1})$. Define five polynomials $p_1, ..., p_5$ as follows:*

$$\begin{aligned}
p_{1,\mathsf{ad},c_1}(\mathsf{val}) &= c_1\mathsf{val} + \sum_{i=1}^{\rho} \mathsf{ad}_i\mathsf{val}^{i+1} + \sum_{i=\rho+2}^{\rho+6} \mathsf{val}^i,\\
p_{2,c_2}(\mathsf{val}) &= c_2\mathsf{val} + \mathsf{val}^{\rho+7},\\
p_{3,c_3}(\mathsf{val}) &= c_3\mathsf{val} + \mathsf{val}^{\rho+8},\\
p_{4,c_4}(\mathsf{val}) &= c_4\mathsf{val} + \mathsf{val}^{\rho+9},\\
p_{5,c_5}(\mathsf{val}) &= c_5\mathsf{val} + \mathsf{val}^{\rho+10}.
\end{aligned}$$

In the above, all calculations are conducted modulo $\mathbb{Z}_{p(\lambda)}$.

Let g be a generator of the group $\mathbb{G}_\lambda$. Let $c_1, c_2, c_3, c_4, c_5 \xleftarrow{\$} \mathbb{Z}_{p(\lambda)}$ be input randomness. Let $\rho \stackrel{def}{=} |\mathsf{ad}|$. Consider the following construction:

$$\mathsf{lockPoint}(\mathsf{val}, \mathsf{ad}; a, b, c) \stackrel{def}{=} \begin{pmatrix} c_1, & [p_{1,\mathsf{ad},c_1}(\mathsf{val})]_g \\ c_2, & [p_{2,c_2}(\mathsf{val})]_g \\ c_3, & [p_{3,c_3}(\mathsf{val})]_g \\ c_4, & [p_{4,c_4}(\mathsf{val})]_g \\ c_5, & [p_{5,c_5}(\mathsf{val})]_g \end{pmatrix}$$

$\mathsf{V}_{\mathtt{obf}}$ *is the circuit that checks that* unlockPoint *consists of the appropriate number of values and group elements. If not, it outputs* 0*. Given a program* unlockPoint *consisting of five pairs* $\{(c'_i, g'_i)\}_{i=1}^{5}$[4] *and inputs* val', ad' *compute:*

$$\left[p_{1,\mathtt{ad}',c'_1}(\mathsf{val}')\right]_g \stackrel{?}{=} g'_1,$$
$$\left\{\left[p_{i,c'_i}(\mathsf{val}')\right]_g \stackrel{?}{=} g'_i\right\}_{i=2}^{5}.$$

If all of these checks pass, output 1*. Otherwise, output* 0*.*

Theorem 4.1. *Let all parameters be as in Construction 4.1, let* $\rho \in \mathbb{N}$ *be a parameter. Define* $\mathcal{F} : \mathbb{Z}_{p(\lambda)} \to \mathbb{Z}_{p(\lambda)}$ *as the set of non-constant, non-identity polynomials of maximum power* τ*. Suppose that*

1. *Assumption 3.1 holds for* $\psi = \max\{\tau(\rho+6), \rho+10\}$ *and*
2. $(\rho+6)2^{2\rho}/p(\lambda)^3 = \mathtt{ngl}(\lambda)$*.*

Then, Construction 4.1 is a $(\mathcal{F}, \mathcal{X}, \rho)$*-nonmalleable point function obfuscation with associated data.*

Remark 3. *In the above, the size of associated data is limited to be* $\rho \approx \log(p(\lambda))$*, which is linear in the security parameter* λ*. Our primary application has the associated data as the CRS of some NIZK. Such strings can be quite long. In Section 5.4, we show that it suffices to include a short value in the associated data whose size is* $\Theta(\log \lambda)$*.*

In order to prove that Construction 4.1 satisfies Definition 3.2, we must prove correctness, virtual black box security, and nonmalleability.

- **Correctness** is described in Lemma 4.1.
- **Virtual black box security** is described in Theorem 4.2.
- **Nonmalleability** is described in Theorem 4.3. First, we show val is nonmalleable for low-degree polynomials in Lemma 4.4. Then, in Lemma 4.5, we show that conditioned on val not being changed, ad is completely nonmalleable. To give some intuition we present a simpler argument for nonmalleability in the algebraic group model in Appendix B.

Lemma 4.1. *For any* ρ *such that* $(\rho+6)2^{2\rho}/p(\lambda)^3 + \rho/p(\lambda) = \mathtt{ngl}(\lambda)$*, Construction 4.1 satisfies completeness.*

Proof. Fix some point $x \in \mathbb{Z}_{p(\lambda)}$ and some $\mathsf{ad} \in \{0,1\}^\rho$. It suffices to argue that over the randomness of $\mathsf{unlockPoint} \leftarrow \mathsf{lockPoint}(x, \mathsf{ad})$ that the probability that there exists some distinct y, ad' such that $\mathsf{unlockPoint}(y, \mathsf{ad}') = 1$ as well is $\mathtt{ngl}(\lambda)$.

Recall that the randomness used to construct unlockPoint is the values a, b, c. Fix some $x \in \mathbb{Z}_{p(\lambda)}, \mathsf{ad} \in \{0,1\}^\rho$. Fix some value a and define

$$\alpha \stackrel{def}{=} p_{1,\mathsf{ad},c_1}(x) = c_1 x + \sum_{i=1}^{\rho} \mathsf{ad}_i x^{i+1} + \sum_{i=\rho+1}^{\rho+6} x^i.$$

[4] g is a generator that is a global system parameter along with the group description. Note that it is efficiently checkable 1) whether the order of a group is prime and 2) whether an element g is a generator of the known order group.

The probability that there exists some $\mathtt{ad}'$ such that $p_{1,\mathtt{ad}',c_1}(x) = \alpha$ requires that x is a root of $p_{1,\mathtt{ad}-\mathtt{ad}',c_1}(x)$. For a fixed x, this is a polynomial of degree upper bounded by ρ, and so it is 0 with probability at most $\rho/p(\lambda)$. For the rest of the proof we assume that for the true value x, there is a unique $\mathtt{ad}$ that where $p_{1,\mathtt{ad},c_1}(x) = \alpha$.

We now count the number of pairs $\mathtt{ad}', y$ such that $p_{1,\mathtt{ad}',c_1}(y) = \alpha$. For some other value $y, \mathtt{ad}'$, since $\mathcal{G}$ is prime order the only way for the first element to match is for

$$\alpha = ay + \sum_{i=1}^{\rho} \mathtt{ad}'_i y^{i+2} + \sum_{i=\rho+2}^{\rho+6} y^i.$$

Since this is a polynomial of degree $\rho + 6$ for each value $\mathtt{ad}'$ there are at most $\rho + 6$ such values y where this is the case.

Using the check values in $\mathsf{unlockPoint}$, these must also verify for $y \neq x$. Consider the polynomial $p(c_2) \stackrel{def}{=} c_2(x - y) + x^{\rho+7} - y^{\rho+7}$. For the first check value of Construction 4.1 to verify for y, $P(c_2)$ must equal 0, as it represents the difference in the exponents of the check value for x and y. This is a linear polynomial in c_2 that is zero with probability at most $1/p(\lambda)$. A similar argument holds for all four check values. Thus, a candidate y is a solution to all four equations equations with probability $1/p(\lambda)^4$. Thus means for a fixed $x, \mathtt{ad}, \mathtt{ad}'$ the probability of one of the y's working is at most $(\rho + 6)/p(\lambda)^4$ by union bound. We apply a union bound over the total number of $x, \mathtt{ad}, \mathtt{ad}'$, the probability across all $x, \mathtt{ad}$ of there existing some $y, \mathtt{ad}'$ is at most $(\rho+6)2^{2\rho}/p(\lambda)^3$ as desired. Adding the case where there are multiple $\mathtt{ad}$ for the original value x one has that the overall probability is at most

$$\frac{(\rho + 6)2^{2\rho}}{p(\lambda)^3} + \frac{\rho}{p(\lambda)}.$$

This completes the proof of Lemma 4.1. □

Theorem 4.2. *Let ρ be the length of $\mathtt{ad}$. Suppose that Assumption 3.1 holds for highest power $\psi = \rho+10$. Then, Construction 4.1 satisfies virtual black box security.*

Proof of Theorem 4.2. Before we prove security, we introduce the following lemma which we will use throughout to simplify our arguments of virtual black box security. It considers the generalized functionality **Point** which encompasses point functions and digital lockers with and without associated data (see Section 3.2).

Lemma 4.2. *Let $\lambda, \rho, \ell \in \mathbb{N}$, and let $m = m(\rho, \ell)$ for some polynomial $m(\cdot, \cdot)$ where $m - 1 \geq \ell$. Let $\mathtt{ad}$ be a binary vector of length ρ and let key be a binary vector of length n, and let $\mathcal{M}(\mathtt{ad}, \mathsf{key}, 1^\lambda)$ be a PPT algorithm which samples a matrix $\mathbf{M} \in \mathbb{Z}_{p(\lambda)}^{\ell \times (m-1)}$ for some prime p that has row rank at least ℓ with probability $1 - \mathtt{ngl}(\lambda)$.*

Suppose that Assumption 3.1 holds for highest power m, and let $\vec{k} = (k_2, ..., k_m)$ be the $(m-1)$-long uniformly distributed vector of coefficients. If whenever $\mathbf{M}$ is full rank the output $\mathbf{Point}(x, \mathtt{ad}, \mathsf{key})$ is identically distributed to:

$$\left((\vec{u}||\mathbf{M}), \left[(\vec{u}||\mathbf{M}) \begin{pmatrix} x \\ x^2 \\ \dots \\ x^m \end{pmatrix} \right]_g, \mathtt{ad} \right),$$

where $\mathbf{M} \leftarrow \mathcal{M}_{\mathtt{ad}}$ and $\vec{u} = \mathbf{M}\vec{k}^T$ then **Point** *satisfies virtual black box security.*

Proof of Lemma 4.2. Let **Point** be a point obfuscator Suppose toward contradiction that there exists some well-spread distribution $\mathcal{X}_\lambda$, ad, key and some matrix sampling process $\mathcal{M} = \mathcal{M}(\mathsf{ad}, \mathsf{key}, 1^\lambda)$ described above, we use the notation $\mathbf{Point}_\mathcal{M}$ to emphasize that the distribution of **Point** can be formed using the processed described in the lemma. Suppose that there exists a PPT adversary $\mathcal{A}_\mathcal{M}$, a polynomial $q(\cdot)$ such that

$$\left| \Pr_{x \leftarrow X_\lambda}[\mathcal{A}_\mathcal{M}(\mathbf{Point}_\mathcal{M}(x, \mathsf{ad}, \mathsf{key}), \mathsf{ad}, \mathsf{key}) = 1] - \Pr_{r \leftarrow \mathbb{Z}_{p(\lambda)}} r[\mathcal{A}_\mathcal{M}(\mathbf{Point}_\mathcal{M}(r, \mathsf{ad}, \mathsf{key}), \mathsf{ad}, \mathsf{key}) = 1] \right| > \frac{1}{q(\lambda)},$$

We show how to build an adversary $\mathcal{B}$ that breaks Assumption 3.1 with respect to distribution family $\mathcal{X}_\lambda$ receiving one fewer than $m = m(\ell, \rho)$ elements (corresponding to maximum power m). That is, $\mathcal{B}$ will receive $m - 1$ pairs of the form

$$\{k_i, g^{k_i z + z^i}\}_{i \in \{2, \ldots, m\}},$$

where z is distributed according to either $\mathcal{X}_\lambda$ or uniformly in $\mathbb{Z}_{p(\lambda)}$. Denote by $\{k_i, [h_i]_g\}_i$ the received values, so $h_i = k_i z + z^i$.

First, $\mathcal{B}$ samples $\mathbf{M} \in \mathbb{Z}_{p(\lambda)}^{(m-1)\times \ell}$ from $\mathcal{M}(\mathsf{ad}, \mathsf{key})$. If $\mathbf{M}$ is not full rank $\mathcal{B}$ flips a random bit. Otherwise, it creates two ℓ-long vectors u and v as so, for each $j \in [\ell]$, set

$$u_j = \sum_{i=1}^{m-1} M_{i,j} k_{i+1} \qquad \text{and} \qquad v_j = \prod_{i=1}^{m-1} [h_{i+1}]_g^{M_{i,j}}. \tag{2}$$

Then, $\mathcal{B}$ computes **Point** as $\mathbf{Point} = \{u_j, v_j\}_{j=1}^{\ell}$. It is clear that the values of v_j form an obfuscation for the transformation

$$\left[(\vec{u} || \mathbf{M}) \begin{pmatrix} z \\ z^2 \\ \ldots \\ z^m \end{pmatrix} \right]_g.$$

Because the probability that $\mathbf{M}$ has rank ℓ is $1 - \mathtt{ngl}(\lambda)$, we see that with overwhelming probability we have formed a valid obfuscation to give (along with $\mathsf{ad}, \mathsf{key}$) to $\mathcal{A}_\mathcal{M}$. Denote by

$$\mathtt{Disting}_\mathcal{A} \stackrel{def}{=} |\Pr[\mathcal{A}_\mathcal{M}(\mathbf{Point}(x, \mathsf{ad}, \mathsf{key}), \mathsf{ad}, \mathsf{key}) = 1] - \Pr[\mathcal{A}_\mathcal{M}(\mathbf{Point}(r, \mathsf{ad}, \mathsf{key}), \mathsf{ad}, \mathsf{key}) = 1]|.$$

Then, we have

$$\begin{aligned} \left| \Pr[\mathcal{B}(\{k_i, g^{k_i x + x^i}\}_{i \in [2,m]})] - \Pr[\mathcal{B}(\{k_i, g^{k_i r + r^i}\}_{i \in [2,m]})] \right| = \\ \Pr[M \leftarrow \mathcal{M}_\mathsf{ad} \text{ full rank}] \cdot \mathtt{Disting}_\mathcal{A} = \\ (1 - \mathtt{ngl}(\lambda)) \frac{1}{q(\lambda)} \geq \frac{1}{q'(\lambda)} \end{aligned}$$

for some polynomial $q'(\lambda)$. We arrive at a contradiction, and this completes the proof of Lemma 4.2. □

From Lemma 4.2, we see that we may prove VBB security by being able to form our construction from a particular linear shift as described in the lemma. To this end, given any ad, we construct the following matrix $\mathbf{D}$:

$$\mathbf{D} = \left(\begin{array}{c|ccccc|cccc} \mathsf{ad} & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ \hline \vec{0}^\rho & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ \hline \vec{0}^\rho & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ \hline \vec{0}^\rho & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ \hline \vec{0}^\rho & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array} \right)$$

It is clear that this matrix has a rank of at 5 (for any $\mathtt{ad}$). The output

$$\left((\mathbf{D}\vec{k}^T || \mathbf{D}), \left[(\mathbf{D}\vec{k}^T || \mathbf{D}) \begin{pmatrix} x \\ x^2 \\ \dots \\ x^m \end{pmatrix} \right]_g, \mathtt{ad} \right),$$

is identically distributed to an obfuscation (the rank 5 condition occurs with probability 1). Thus, the construction of **Point** implied by Lemma 4.2 is VBB secure. □

Theorem 4.3. *Let λ be a security parameter. Let $\{\mathcal{X}_\lambda\}$ be a well-spread distribution ensemble and let $\tau, \rho \in \mathbb{Z}^+$ be parameters that are both $\mathtt{poly}(\lambda)$. Let $\mathcal{F}_{poly}$ be the ensemble of functions f_λ where f_λ is the set of non-constant, non-identity polynomials in $\mathbb{Z}_{p(\lambda)}[x]$ with degree at most τ. Suppose that Assumption 3.1 holds for $\psi = \max\{\rho+10, \tau(\rho+6)\}$. Then, the obfuscator in Construction 4.1 is nonmalleable over $\mathcal{F}_{poly}$ with distribution ensemble $\{\mathcal{X}_\lambda\}$, and $\mathcal{AD} = \{0,1\}^\rho$.*

The proof strategy is as follows:

1. **Lemma 4.3.** Showing that any method of incorporating associated data suffices for keeping val from being changed as long as there are enough large powers of val that are not affected by associated data.

2. **Lemma 4.5.** Show that if the value val is not tampered then for Construction 4.1 it is difficult to change $\mathtt{ad} \in \{0,1\}^\rho$.

Lemma 4.3. *Let λ be a security parameter. Let $\{\mathcal{X}_\lambda\}$ be a well-spread distribution ensemble and let $\tau, \ell \in \mathbb{Z}^+$ be $\mathtt{poly}(\lambda)$. Let $\mathcal{F}_{poly}$ be the ensemble of functions f_λ where f_λ is the set of non-constant, non-identity polynomials in $\mathbb{Z}_{p(\lambda)}[x]$ with degree at most τ.*

Let $P(x) = r_1 x + \ldots + r_{\rho-1}x^{\rho-1} + r_\rho x^\rho$ with $r_i \in \mathbb{Z}_{p(\lambda)}$, and let $\vec{P} = \{r_1, \ldots, r_\rho\}$ where any or all of the r_i may be 0. Suppose that Assumption 3.1 holds for $\psi = \max\{\rho+10, \tau(\rho+6)\}$. Define as obfuscation (with c_1, c_2, c_3, c_4, c_5 uniformly distributed in $\mathbb{Z}_{p(\lambda)}$)

$$\mathsf{lockPoint}_P(\mathsf{val}, \vec{P}; c_1, c_2, c_3, c_4, c_5) \stackrel{def}{=} \vec{P}, \begin{bmatrix} c_1, & \left[c_1\mathsf{val} + \mathsf{val}P(\mathsf{val}) + \sum_{i=\rho+2}^{\rho+6} \mathsf{val}^i\right]_g \\ c_2, & \left[c_2\mathsf{val} + \mathsf{val}^{\rho+7}\right]_g \\ c_3, & \left[c_3\mathsf{val} + \mathsf{val}^{\rho+8}\right]_g \\ c_4, & \left[c_4\mathsf{val} + \mathsf{val}^{\rho+9}\right]_g \\ c_5, & \left[c_5\mathsf{val} + \mathsf{val}^{\rho+10}\right]_g \end{bmatrix}.$$

Consider $\mathcal{F}_{poly}$ and distribution ensemble $\{\mathcal{X}_\lambda\}$. For any nonmalleability PPT adversary $\mathcal{A}$ in Definition 3.2, $\mathcal{A}$ outputs a valid $f, P', \mathsf{unlockPoint}_{P'}$ with negligible probability.

Proof. We look to contradict Assumption 3.2. Consider a mauling adversary $\mathcal{A}_P$ that, given an obfuscation with polynomial $P(x)$, can output a new obfuscation of $f(x)$ for $f \in \mathcal{F}_{poly}$. Consider τ to be the degree of f and ρ to be the (maximum) degree of P. For k_i, define $h_i \stackrel{def}{=} k_i x + x^i$. We build an adversary $\mathcal{B}_P$, which given the ensemble $\{k_i, [h_i]_g\}_{i=2,\ldots,\psi}$ and access to $\mathcal{A}_P$ can recover g^x with non-negligible probability.

We will separately consider cases where $\tau > 1$ and where $\tau = 1$ (a linear polynomial). In both cases $\mathcal{B}_P$ will prepare the obfuscation in the same manner, forwarding the distribution $\mathcal{X}_\lambda$ and keeping the

coefficients r_i received from $\mathcal{A}_P$. Upon receiving the ensemble $\{k_i, [h_i]_g\}_{i=2\ldots,\psi}$, we create an obfuscation of x as so:

$$\begin{aligned}
(c_1, g_1) &= \left(\sum_{i=2}^{\rho+1} r_i k_i + \sum_{i=\rho+2}^{\rho+6} k_i, \prod_{i=1}^{\rho} [h_i]_g^{r_i} \cdot \prod_{i=\rho+2}^{\rho+6} [h_i]_g \right) \\
(c_2, g_2) &= (k_{\rho+7}, [h_{\rho+7}]_g) \\
(c_3, g_3) &= (k_{\rho+8}, [h_{\rho+8}]_g) \\
(c_4, g_4) &= (k_{\rho+9}, [h_{\rho+9}]_g) \\
(c_5, g_5) &= (k_{\rho+10}, [h_{\rho+10}]_g)
\end{aligned}$$

Send these to $\mathcal{A}_P$ alongside $\vec{P}$ itself, which returns (without loss of generality)

$$(f, P', (c_1', g_1', c_2', g_2', c_3', g_3', c_4', g_4', c_5', g_5'))$$

where P' is the description of a polynomial of degree at most ρ. $\mathcal{B}_P$ ignores everything but f, P', c_1', and g_1'.

Non Linear Case. First, we consider the case when the degree of f denoted as $\tau > 1$. Define the polynomial $\ell(x)$ as the coefficients of:

$$\ell(x) = \sum_{i=0}^{\tau(\rho+6)} \vec{l}_i x^i \stackrel{def}{=} c_1' f(x) + f(x) P'(f(x)) + f(x)^{\rho+2} \sum_{i=0}^{4} (f(x))^i.$$

That is, $\ell(x)$ is the polynomial $\mathcal{A}_P$ claims they have mauled the exponent of the term g_1' to. Note that the coefficients of ℓ are computable by $\mathcal{B}_P$. Let h^* denote the exponent of g_1'. In order for the $\mathcal{A}_P$ to succeed in tampering, it must be the case that $\ell(x) = h^*$ with noticeable probability (over the choice of x and the $\mathcal{A}_P$'s randomness).

Since $\mathcal{B}_P$ has properly prepared the obfuscation for $\mathcal{A}_P$, $\mathcal{A}_P$ returns a valid obfuscation of $f(x)$ with probability at least $1/\mathtt{poly}(\lambda)$. That is, that $h^* = \ell(x)$ with the same probability. $\mathcal{B}_P$ computes and returns

$$\left(g_1' \left([\ell_0]_g \cdot \prod_{i=2}^{\tau(\rho+6)} [h_i]_g^{\ell_i} \right)^{-1} \right)^{1/\left(l_1 - \sum_{i=2}^{\tau(\rho+6)} k_i l_i \right)}.$$

With noticeable probability $g_1' = [h^*]_g$ with $h^* = \sum_{i=0}^{\tau(\rho+6)} \ell_i x^i$. When this occurs,

$$\begin{aligned}
\left(g_1' \left([\ell_0]_g \cdot \prod_{i=2}^{\tau(\rho+6)} \ell_i [h_i]_g^{\ell} \right)^{-1} \right) &= \left[\sum_{i=0}^{\tau(\rho+6)} \ell_i x^i - \ell_0 - \sum_{i=2}^{\tau(\rho+6)} \ell_i (k_i x + x^i) \right]_g \\
&= \left[x \left(\ell_1 - \sum_{i=2}^{\tau(\rho+6)} k_i \ell_i \right) \right]_g.
\end{aligned}$$

Thus, in the case that $\mathcal{A}_P$ returns the correct value and $\ell_1 - \sum_{i=2}^{\tau(\rho+6)} k_i \ell_i \neq 0$, $\mathcal{B}_P$ computes and return the correct value. Since $f(x)$ is of degree τ, $l_{\tau(\rho+6)}$ must be nonzero. $\mathcal{A}_P$'s view is independent of $k_{\tau(\rho+6)}$.[5] So, the probability that the sum

$$\Pr_{k_{\tau(\rho+6)}}\left[\sum_{i=2}^{\tau(\rho+6)} k_i \ell_i = \ell_1\right] = \frac{1}{p(\lambda)-1}.$$

So, $\mathcal{B}_P$ returns the correct value with probability $1/\texttt{poly}(\lambda) - 1/(p(\lambda)-1) = 1/\texttt{poly}(\lambda)$ contradicting Assumption 3.2.

Linear Case. We now consider the case where $\tau = 1$, or for linear functions f. In this case, we only use the ensemble $\{k_i, [k_i x + x^i]_g\}_{i=2,\ldots,\rho+10}$. This time, of the ensemble (f, P', c_1', g_1') which $\mathcal{B}_P$ receives and reserves from $\mathcal{A}_P$, we see $f = \alpha x + \beta$. Once again, $\mathcal{B}$ computes the coefficients of the polynomial ℓ, this time as

$$\sum_{i=0}^{\rho+6} \ell_i x^i \stackrel{def}{=} c_1' f(x) + f(x) \cdot P'(f(x)) + f(x)^{\rho+2} \sum_{i=0}^{4} f(x)^i.$$

In this case, $\mathcal{B}_P$ computes and outputs:

$$\left(g_1'\left([l_0]_g \cdot \prod_{i=2}^{\rho+6} h_i^{l_i}\right)^{-1}\right)^{1/\left(l_1 - \sum_{i=2}^{\rho+6} k_i l_i\right)}.$$

As before let h^* denote the exponent of g_1'. Then,

$$h^* = c_1' f(x) + f(x) P'(f(x)) + f(x)^{\rho+2} \sum_{i=0}^{4} f(x)^i$$

with noticeable probability, and $\mathcal{B}_P$'s computation evaluates to g^x unless $l_1 - \sum_{i=2}^{\rho+6} k_i l_i = 0$. To see that this happens with negligible probability, note that the original c_1 is formed as

$$c_1 = \sum_{i=2}^{\rho+1} r_{i-1} k_i + \sum_{i=\rho+2}^{\rho+6} k_i$$

for random values $k_2, ..., k_{\rho+6}$.

As all k_i are uniformly chosen, the only information $\mathcal{A}$ learns about $k_{\rho+2}, ..., k_{\rho+6}$ is in the value c_1. Since these are the terms we will be relying on for nonmalleability, we re-index, letting $\delta = \rho + 2$. We additionally define $c = l_1 - \sum_{i=2}^{\delta-1} k_i l_i$, so that we want to show $c - \sum_{i=\delta}^{\delta+4} k_i l_i \neq 0$ except with negligible probability.

Without loss of generality, we assume that the adversary knows the values $k_2, \ldots, k_{\delta-1}$ exactly. That is, assume the adversary sees the values $(k_2, \ldots, k_{\delta-1}, z)$ where $z := k_\delta + \ldots + k_{\delta+4}$. Equivalently, we have $k_{\delta+4} = z - k_{\delta+3} - k_{\delta+2} - k_{\delta+1} - k_\delta$. Using these substitutions, we may rewrite $c - \sum_{i=\delta}^{\delta+4} k_i l_i$ as

$$c - l_{\delta+4} z + (l_{\delta+4} - l_{\delta+3}) k_{\delta+3} + (l_{\delta+4} - l_{\delta+2}) k_{\delta+2} + (l_{\delta+4} - l_{\delta+1}) k_{\delta+1} + (l_{\delta+4} - l_\delta) k_\delta$$

[5] For all $\rho > 0$ if $\tau > 1$ then the value $\tau(\rho+6) > \rho + 8$.

So, it remains to show that this expression is zero with negligible probability over the adversary's choices of l. Suppose not. Because f is linear, $f(x) = \alpha x + \beta$, and so the adversary must find α, β, γ such that $l_{\delta+i} = \gamma$ for all $i \in \{0,1,2,3,4\}$ (as they know z but not the k-values in the expression). Rewriting the ℓ values in terms of terms of the binomial coefficients of α and β, the desired linear combination is:

$$\begin{bmatrix}\ell_{\delta+4}\\ \ell_{\delta+3}\\ \ell_{\delta+2}\\ \ell_{\delta+1}\\ \ell_{\delta}\end{bmatrix}^{\mathsf{T}}\begin{bmatrix}k_{\delta+4}\\ k_{\delta+3}\\ k_{\delta+2}\\ k_{\delta+1}\\ k_{\delta}\end{bmatrix} = \begin{bmatrix}\alpha^{\delta+4}\\ \alpha^{\delta+3}\left(\binom{\delta+4}{1}\beta + \binom{\delta+3}{0}\right)\\ \alpha^{\delta+2}\left(\binom{\delta+4}{2}\beta^2 + \binom{\delta+3}{1}\beta + \binom{\delta+2}{0}\right)\\ \alpha^{\delta+1}\left(\sum_{i=0}^{3}\left(\binom{\delta+4-i}{3-i}\beta^{3-i}\right)\right)\\ \alpha^{\delta}\left(\sum_{i=0}^{4}\left(\binom{\delta+4-i}{4-i}\beta^{4-i}\right)\right)\end{bmatrix}^{\mathsf{T}}\begin{bmatrix}k_{\delta+4}\\ k_{\delta+3}\\ k_{\delta+2}\\ k_{\delta+1}\\ k_{\delta}\end{bmatrix} = \gamma\begin{bmatrix}k_{\delta+4}\\ k_{\delta+3}\\ k_{\delta+2}\\ k_{\delta+1}\\ k_{\delta}\end{bmatrix}.$$

Lemma 4.4. *For $\alpha, \beta, \delta, \gamma \in \mathbb{Z}_{p(\lambda)}$ the only solutions of*

$$\begin{bmatrix}\alpha^{\delta+4}\\ \alpha^{\delta+3}\left(\binom{\delta+4}{1}\beta + \binom{\delta+3}{0}\right)\\ \alpha^{\delta+2}\left(\binom{\delta+4}{2}\beta^2 + \binom{\delta+3}{1}\beta + \binom{\delta+2}{0}\right)\\ \alpha^{\delta+1}\left(\sum_{i=0}^{3}\left(\binom{\delta+4-i}{3-i}\beta^{3-i}\right)\right)\\ \alpha^{\delta}\left(\sum_{i=0}^{4}\left(\binom{\delta+4-i}{4-i}\beta^{4-i}\right)\right)\end{bmatrix}^{\mathsf{T}}\begin{bmatrix}k_{\delta+4}\\ k_{\delta+3}\\ k_{\delta+2}\\ k_{\delta+1}\\ k_{\delta}\end{bmatrix} = \gamma\begin{bmatrix}k_{\delta+4}\\ k_{\delta+3}\\ k_{\delta+2}\\ k_{\delta+1}\\ k_{\delta}\end{bmatrix}$$

that hold for all values of $k_\delta, ..., k_{\delta+4}$ are $(\alpha = 0, \beta = 0)$, $(\alpha = 1, \beta = 0)$, $\alpha = -1, \delta = -5$ and $\alpha - 1, \delta = -6$.

We note that since $\rho = \mathtt{poly}(\lambda)$ for large enough λ one can be sure that $\rho \notin \{-8, -9\} \mod p(\lambda)$ and thus, $\delta \notin \{-6, -7\} \mod p(\lambda)$. So, the only functions that $\mathcal{A}$ can maul to with noticable probability are the constant and identity functions, neither of which are in $\mathcal{F}_{poly}$. So, with non-negligible probability, $\mathcal{B}_P$ returns some $f \in \mathcal{F}$ (given Assumption 3.2). This completes the proof of Lemma 4.3. □

Lemma 4.5. *Let λ be a security parameter. Let $\{\mathcal{X}_\lambda\}$ be a well-spread distribution ensemble and let $\tau, \rho \in \mathbb{Z}^+$ be $\mathtt{poly}(\lambda)$. Let $\mathcal{F}_{poly}$ be the ensemble of functions f_λ where f_λ is the set of non-constant, non-identity polynomials in $\mathbb{Z}_{p(\lambda)}[x]$ with degree at most τ.*

Let $P_{\vec{b}}(x) = b_\rho x^\rho + b_{\rho-1}x^{\rho-1} + \ldots + b_1 x$ where $b_i \in \{0,1\}$. Suppose that Assumption 3.2 holds for $\psi = \max\{\rho + 10, \tau(\rho+6)\}$. Define as an obfuscation (with c_1, c_2, c_3, c_4, c_5 uniformly distributed in $p(\lambda)$):

$$\mathsf{lockPoint}(\mathsf{val}, \vec{b}; c_1, c_2, c_3, c_4, c_5) \stackrel{def}{=} \vec{b}, \begin{bmatrix}c_1, & \left[c_1\mathsf{val} + \mathsf{val}P_{\vec{b}}(\mathsf{val}) + \sum_{i=\rho+2}^{\rho+6}\mathsf{val}^i\right]_g\\ c_2, & \left[c_2\mathsf{val} + \mathsf{val}^{\rho+7}\right]_g\\ c_3, & \left[c_3\mathsf{val} + \mathsf{val}^{\rho+8}\right]_g\\ c_4, & \left[c_4\mathsf{val} + \mathsf{val}^{\rho+9}\right]_g\\ c_5, & \left[c_5\mathsf{val} + \mathsf{val}^{\rho+10}\right]_g\end{bmatrix}.$$

Consider $\mathcal{F}_{poly}$ and distribution ensemble $\{\mathcal{X}_\lambda\}$. The probability that any PPT algorithm outputs a valid obfuscation with the identity function f and some $P_{\vec{b}'}$ with $\vec{b}' \in \{0,1\}^\rho, \vec{b}' \neq \vec{b}$ is negligible.

Proof. Our proof follows the same structure as the proof of Lemma 4.3. However, we now assume a mauling adversary $\mathcal{A}_{\vec{b}}$ that, given an obfuscation with polynomial $P(x)$ (with binary coefficients) outputs a new obfuscation

$$(f, \vec{b}', (c_1', g_1', c_2', g_2', c_3', g_3', c_4', g_4', c_5', g_5'))$$

where f is the identity function, $\vec{b}' \neq \vec{b}$ and the obfuscation being correct with noticeable probability. As before, we use this $\mathcal{A}_{\vec{b}}$ to construct a $\mathcal{B}_{\vec{b}}$ that breaks Assumption 3.2. Let ρ to be the degree of P. For k_i, define $h_i \stackrel{def}{=} k_i x + x^i$. We build an adversary $\mathcal{B}_{\vec{b}}$, which given the ensemble $\{k_i, [h_i]_g\}_{i=2,\ldots,\psi}$ and access to $\mathcal{A}_{\vec{b}}$ can recover g^x with non-negligible probability. $\mathcal{B}_{\vec{b}}$ prepares the obfuscation as before in the same manner, forwarding the distribution $\mathcal{X}_\lambda$ and keeping the values $r_1, ..., r_\rho$ received in the description of P from $\mathcal{A}_{\vec{b}}$. Upon receiving the ensemble $\{k_i, [h_i]_g\}_{i=2\ldots,\psi}$, we create an obfuscation of x as so:

$$\begin{aligned}
(c_1, g_1) &= \left(\sum_{i=2}^{\rho+1} r_i k_i + \sum_{i=\rho+2}^{\rho+6} k_i, \prod_{i=1}^{\rho} [h_i]_g^{r_i} \cdot \prod_{i=\rho+2}^{\rho+6} [h_i]_g \right) \\
(c_2, g_2) &= (k_{\rho+7}, [h_{\rho+7}]_g) \\
(c_3, g_3) &= (k_{\rho+8}, [h_{\rho+8}]_g) \\
(c_4, g_4) &= (k_{\rho+9}, [h_{\rho+9}]_g) \\
(c_5, g_5) &= (k_{\rho+10}, [h_{\rho+10}]_g)
\end{aligned}$$

Send these to $\mathcal{A}_{\vec{b}}$, which returns (without loss of generality)

$$(f, \vec{b}', c_1', g_1', c_2', g_2', c_3', g_3', c_4', g_4', c_5', g_5'))$$

where $\vec{b}' \in \{0,1\}^\rho$ and f is the identity. $\mathcal{B}_{\vec{b}}$ only uses $f, \vec{b}', c_1'$, and g_1' from here on. Define the polynomial $\ell(x)$ as the coefficients of

$$\ell(x) = \sum_{i=0}^{\rho+6} \vec{l}_i x^i \stackrel{def}{=} c_1' x + x P_{\vec{b}'}(x) + x^{\rho+2} \sum_{i=0}^{4} x^i.$$

Note that the coefficients of ℓ are computable by $\mathcal{B}_{\vec{b}}$. With the exception of c_1', the coefficients of ℓ are all binary.

Let h^* denote the exponent of g_1'. In order for the $\mathcal{A}_{\vec{b}}$ to succeed in tampering, it must be the case that $\ell(x) = h^*$ with noticeable probability (over the choice of x and the $\mathcal{A}_{\vec{b}}$'s randomness).

Since $\mathcal{B}_{\vec{b}}$ has properly prepared the obfuscation for $\mathcal{A}_{\vec{b}}$, $\mathcal{A}_{\vec{b}}$ returns a valid obfuscation of $\vec{b}', x$ with probability at least $1/\mathtt{poly}(\lambda)$. That is, that $h^* = \ell(x)$ with the same probability. $\mathcal{B}_{\vec{b}}$ computes and outputs

$$\left(g_1' \left(\prod_{i=2}^{\rho+6} h_i^{l_i} \right)^{-1} \right)^{1/\left(l_1 - \sum_{i=2}^{\rho+6} k_i l_i \right)}.$$

As before let h^* denote the exponent of g_1'. Then, $h^* = c_1' x + x P_{\vec{b}'}(x) + x^{\rho+2} \sum_{i=0}^{4} x^i$ with noticeable probability, and $\mathcal{B}_{\vec{b}}$'s computation evaluates to $[x]_g$ unless $l_1 - \sum_{i=2}^{\rho+6} k_i l_i = 0$. For $\vec{b}' \neq \vec{b}$, there must be at least one bit i where $b_i = 1 \wedge b_i' = 0$ or $b_i' = 0 \wedge b_i = 1$. Note that, for all $i \in [2, \rho+1]$, $l_i = b_i'$. We now split our analysis into two cases:

$\exists i$ **such that** $b_i = 0 \wedge b'_i = 1$**.** Let i^* denote one such location. Then, $\ell_{i^*} = 1$. Furthermore, $\mathcal{A}_{\vec{b}}$'s view is independent of k_{i^*} since $b_i = 0$. So, the probability that the sum

$$\Pr_{k_{i^*}}\left[\left(\ell_1 - \sum_{i=2, i\neq i^*}^{\rho+6} k_i \ell_i\right) = k_{i^*}\right] = \frac{1}{p(\lambda) - 1}.$$

So, $\mathcal{B}_{\vec{b}}$ returns the correct value with probability $1/\texttt{poly}(\lambda) - 1/(p(\lambda) - 1) = 1/\texttt{poly}(\lambda)$ contradicting Assumption 3.2.

$\forall i, b_i = 0 \rightarrow b'_i = 0$ **and** $\exists i$ **such that** $b_i = 1 \wedge b'_i = 0$**.** Let i^* denote one such index. All information about k_{i^*} known by $\mathcal{A}_{\vec{b}}$ is contained in the value $\sum_{i=2}^{\rho+1} r_i k_i + \sum_{i=\rho+2}^{\rho+6} k_i$ since $\{k_i\}_{\rho+2 \leq i \leq \rho+6}$ are uniformly distributed one has

$$\forall s, \Pr\left[k_{i^*} = s \middle| \sum_{i=2}^{\rho+1} r_i k_i + \sum_{i=\rho+2}^{\rho+6} k_i\right] = \Pr[k_{i^*} = s] = \frac{1}{p(\lambda)}.$$

Thus, as before one has

$$\Pr_{k_{i^*}}\left[\left(\ell_1 - \sum_{i=2}^{\rho+6} k_i \ell_i\right) = 0\right] = \frac{1}{p(\lambda) - 1}.$$

Thus, in both cases,

$$\exists i^*, \Pr_{k_{i^*}}\left[\left(\ell_1 - \sum_{i=2}^{\rho+6} k_i \ell_i\right) = 0\right] = \frac{1}{p(\lambda) - 1}.$$

This completes the proof of Lemma 4.5 by noting that these cases are exhaustive. □

Corollary 4.1. *Construction 4.1 satisfies nonmalleability under Assumption 3.2.*

5 Standard Model Digital Lockers

We will now construct a nonmalleable digital locker in two steps.

- In Section 5.1 we amend our previous construction of a $\texttt{NMPO}_{\mathsf{ad}}$ to instead output a predetermined key rather than a single bit. Nonmalleability of the input val and ad must still be preserved, but *no* nonmalleability is guaranteed for key.
- In Section 5.2, we use this intermediate digital locker with a non-interactive zero knowledge proof, to guarantee complete nonmalleability over key.

Of course, correctness and security must hold for val, key as well. The end result of these efforts (Construction 5.2) will be a digital locker with: 1) input val nonmalleable over low-degree polynomials, 2) public helper string ad nonmalleable over any tampering, and 3) output key nonmalleable over any tampering. As we will see in Section 6, these tampering classes have meaningful applications.

5.1 Digital Lockers Nonmalleable over val and ad

We integrate our $\mathtt{NMPO}_{\mathsf{ad}}$ with the real-or-random construction [21] in Figure 1. The essential idea is that we may encode each bit of key as a real (encoding val) or random (encoding a random point) point obfuscation, with an additional obfuscation of val to ensure that is the point being tested. We encode the attestation of ad in this additional obfuscation.

In order to adapt our techniques to a real-or-random digital locker with $|\mathsf{key}| = \ell$, then, it is clear that we must ensure that each point obfuscation retains security in the presence of up to ℓ other copies of the same point (i.e., if $\mathsf{key} = 1^\ell$). The previous construction is clearly not sufficient, providing two copies of the obfuscation breaks security (see discussion in [39]), but we may use similar techniques as so. We begin by defining the intermediate cryptographic object.

Definition 5.1 (Input Nonmalleable Digital Locker with Associated Data)**.** *For security parameter $\lambda \in \mathbb{N}$, let $\{\mathsf{D}^\lambda\}$ be an ensemble of finite sets, let $\rho \in \mathbb{N}$ be a parameter. Let*

1. *$\mathcal{F} : \mathsf{D}^\lambda \to \mathsf{D}^\lambda$ be a function family,*
2. *$\mathcal{X}$ be a family of distributions over D^λ,*
3. *iLock be a PPT algorithm that maps points $\mathsf{val} \in \mathsf{D}^\lambda, \mathsf{ad} \in \{0,1\}^\rho, \mathsf{key} \in \{0,1\}^n$ to a circuit $\mathsf{iUnlock}$, and*
4. *$\mathsf{V}_{\mathtt{obf}}$ be an obfuscation verifier.*

The algorithm iLock is a $(\mathcal{F}, \mathcal{X}, \rho, n)$-input nonmalleable digital locker *with associated data if all of the below are satisfied:*

1. ***Completeness*** *For all $\mathsf{val} \in \mathsf{D}^\lambda, \mathsf{ad} \in \{0,1\}^\rho, \mathsf{key} \in \{0,1\}^n$ it holds that*

$$\Pr[\mathsf{iUnlock}(\cdot) \equiv I_{\mathsf{val},\mathsf{ad},\mathsf{key}}(\cdot) | \mathsf{iUnlock} \leftarrow \mathsf{iLock}(\mathsf{val}, \mathsf{ad}, \mathsf{key})] \geq 1 - \mathtt{ngl}(\lambda),$$

   where the probability is over the randomness of iLock. Here $I_{\mathsf{val},\mathsf{ad},\mathsf{key}}$ is a function that returns key when provided input $(\mathsf{val}, \mathsf{ad})$, otherwise $I_{\mathsf{val},\mathsf{ad},\mathsf{key}}$ returns $\perp$.

2. ***Virtual Black Box Security****: For all PPT $\mathcal{A}$ and $p(\lambda) = \mathtt{poly}(\lambda)$, $\exists \mathcal{S}$ and $q(\lambda) = \mathtt{poly}(\lambda)$ such that for all large enough $\lambda \in \mathbb{N}$, $\forall \mathsf{val} \in \mathsf{D}^\lambda, \mathsf{ad} \in \{0,1\}^\rho, \mathsf{key} \in \{0,1\}^n, \mathcal{P} : \mathsf{D}^\lambda \times \{0,1\}^{\rho+n} \mapsto \{0,1\}$,*

$$\Big| \Pr[\mathcal{A}(\mathsf{iLock}(\mathsf{val}, \mathsf{ad}, \mathsf{key}), \mathsf{ad}) = \mathcal{P}(\mathsf{val}, \mathsf{ad}, \mathsf{key})] - \Pr[\mathcal{S}^{I_{\mathsf{val},\mathsf{ad},\mathsf{key}}}(1^\lambda, \mathsf{ad}) = \mathcal{P}(\mathsf{val}, \mathsf{ad}, \mathsf{key})]\Big| \leq \frac{1}{p(\lambda)},$$

   where S is allowed $q(\lambda)$ oracle queries to $I_{\mathsf{val},\mathsf{ad},\mathsf{key}}$ and the probabilities are over the internal randomness of $\mathcal{A}$ and lock, and of $\mathcal{S}$, respectively.

3. ***Input Nonmalleability*** *For all $X \in \mathcal{X}$, PPT $\mathcal{A}, \mathsf{ad} \in \{0,1\}^\rho, \mathsf{key} \in \{0,1\}^n$, there exists $\epsilon = \mathtt{ngl}(\lambda)$ such that:*

$$\Pr_{\mathsf{val} \leftarrow X}\left[\begin{array}{r} \mathsf{V}_{\mathtt{obf}}(C) = 1, \\ f \in \mathcal{F} \vee (f = \mathsf{id} \wedge \mathsf{ad}' \neq \mathsf{ad}) \\ C(f(\mathsf{val}), \mathsf{ad}') \neq \perp \end{array} \middle| \begin{array}{l} \mathsf{unlock}_{\mathsf{val},\mathsf{key}} \leftarrow \mathsf{iLock}(\mathsf{val}, \mathsf{ad}, \mathsf{key}) \\ (C, f, \mathsf{ad}') \leftarrow \mathcal{A}(\mathsf{unlock}_{\mathsf{val},\mathsf{key}}, \mathsf{ad}) \end{array}\right] \leq \epsilon.$$

Remark 4. *Note that input nonmalleability does not protect against* key *tampering. In fact, an adversary that arbitrarily mauls* key *to $\mathsf{key}' \in \{0,1\}^n$ is allowed for this object, so long as* val *and* ad *are not tampered.*

$\mathsf{iLock}(\mathsf{val}, \mathsf{ad}, \mathsf{key})$:

1. Define $\ell = |\mathsf{key}|$,
2. Sample $z \leftarrow \mathbb{Z}_{p(\lambda)}$,
3. Sample $\mathbf{C} \leftarrow \mathbb{Z}_{p(\lambda)}^{(\ell+1)\times(\ell+1)}$, $c_{0,\ell+2}, c_{0,\ell+3}, c_{0,\ell+4}, c_{0,\ell+5} \leftarrow \mathbb{Z}_{p(\lambda)}$,
4. Compute $\mathsf{unlockPoint}_{\mathsf{ad}} =$

$$\begin{bmatrix} & [p_{0,\mathsf{ad},\mathbf{C}_0}(\mathsf{val})]_g \\ c_{0,\ell+2}, & [p_{0,1,c_{0,\ell+2}}(\mathsf{val})]_g \\ c_{0,\ell+3}, & [p_{0,2,c_{0,\ell+3}}(\mathsf{val})]_g \\ c_{0,\ell+4}, & [p_{0,3,c_{0,\ell+4}}(\mathsf{val})]_g \\ c_{0,\ell+5}, & [p_{0,4,c_{0,\ell+5}}(\mathsf{val})]_g \end{bmatrix}$$

5. For $i = 1$ to ℓ:

$$\mathsf{unlockPoint}_i = \begin{cases} [p^*_{\mathbf{C}_i}(\mathsf{val})]_g & \mathsf{key}_i = 1 \\ [p^*_{\mathbf{C}_i}(z)]_g & \mathsf{key}_i = 0 \end{cases}.$$

6. Output $\mathbf{C}, \mathsf{unlockPoint}_{\mathsf{ad}}, \{\mathsf{unlockPoint}_i\}_{i=1}^{\ell}$.

$\mathsf{iUnlock}(\mathbf{C}', \mathsf{unlockPoint}'_{\mathsf{ad}}, \{\mathsf{unlockPoint}'_i\}_{i=1}^{\ell}, \mathsf{val}', \mathsf{ad}')$:

1. Parse $\mathsf{unlockPoint}'_{\mathsf{ad}}$ as $c'_{0,\ell+2}, c'_{0,\ell+3}, c'_{0,\ell+4}, c'_{0,\ell+5}, g'_{0,1}, g'_{0,2}, g'_{0,3}, g'_{0,4}, g'_{0,5}$.
2. Verify

$$[p_{1,\mathsf{ad}',\mathbf{C}'_0}(\mathsf{val}')]_g = g'_{0,1},$$
$$\left\{ \left[p_{i,c'_{0,i}}(\mathsf{val}')\right]_g = g'_{0,i} \right\}_{i=2}^{5}.$$

   If one checks does not pass output $\perp$.
3. Initialize $\mathsf{key} = \vec{0}^{\ell}$.
4. For $i = 1$ to ℓ:
   (a) If $[p^*_{\mathbf{C}'_i}(\mathsf{val}')]_g = \mathsf{unlockPoint}'_i$ set $\mathsf{key}_i = 1$.
5. Output key.

Figure 1: Real-or Random-Instantiation of Input Nonmalleable Digital Locker with Associated Data.

Before introducing the construction, we define some polynomials that will be used in the construction as follows:

$$p_{0,\mathsf{ad},\vec{c_0}}(\mathsf{val}) = c_{0,1}\mathsf{val} + \sum_{i=1}^{\ell} c_{0,i+1}\mathsf{val}^{i+1} + \sum_{i=1}^{\rho} \mathsf{ad}_i\mathsf{val}^{\ell+1+i} + \sum_{i=1}^{5} \mathsf{val}^{\ell+\rho+1+i}, \tag{3}$$

$$p_{0,1,c_{0,\ell+2}}(\mathsf{val}) = c_{0,\ell+2}\mathsf{val} + \mathsf{val}^{\ell+\rho+7}, \tag{4}$$

$$p_{0,2,c_{0,\ell+3}}(\mathsf{val}) = c_{0,\ell+3}\mathsf{val} + \mathsf{val}^{\ell+\rho+8}, \tag{5}$$

$$p_{0,3,c_{0,\ell+4}}(\mathsf{val}) = c_{0,\ell+4}\mathsf{val} + \mathsf{val}^{\ell+\rho+9}, \tag{6}$$

$$p_{0,4,c_{0,\ell+5}}(\mathsf{val}) = c_{0,\ell+5}\mathsf{val} + \mathsf{val}^{\ell+\rho+10}, \tag{7}$$

$$p^*_{\vec{c}}(\mathsf{val}) = c_{j,1}\mathsf{val} + \sum_{i=1}^{\ell} c_{j,i+1}\mathsf{val}^{i+1}. \tag{8}$$

Construction 5.1. *Let $\lambda \in \mathbb{N}$ be a security parameter, let $\rho, \ell \in \mathbb{N}$ be parameters. Let $\mathcal{G} = \{\mathbb{G}_\lambda\}$ be a group ensemble with efficient representation and operations where each $\mathbb{G}_\lambda$ is a group of prime order $p(\lambda) \in (2^\lambda, 2^{\lambda+1})$. Let $\mathsf{D}^\lambda = \mathbb{Z}_{p(\lambda)}$. Let g be a generator of $\mathbb{G}_\lambda$. Let $\rho, \ell \in \mathbb{Z}^+$ such that $\rho = O(\log \lambda)$ and $\ell = \mathtt{poly}(\lambda)$. Define the Construction of* $(\mathsf{iLock}, \mathsf{iUnlock})$ *as in Figure 1.*

Theorem 5.1. *Let all parameters be as in Construction 5.1. Let $\tau \in \mathbb{N}$ and $\rho \in \mathbb{N}$ be parameters.*

1. *Suppose that Assumption 3.1 holds for maximum power* $\max\{\ell + \rho + 10, \tau(\ell + \rho + 6)\}$,

2. *Let* $\mathcal{F}_{\mathtt{poly}}$ *be the family of polynomials over* $\mathbb{Z}_{p(\lambda)}$ *with maximum degree* τ*, and*

3. $(\ell+\rho+10)2^{2\rho}/p(\lambda)^3 = \mathtt{ngl}(\lambda)$.

Then, Construction 5.1 is a $(\mathcal{F}_{\mathtt{poly}}, \mathcal{X}, \rho, \ell)$*-input nonmalleable digital locker with associated data.*

Proof. As before, we prove completeness, security, and nonmalleability separately.

Completeness: The following lemma proves the necessary completeness guarantee.[6]

Lemma 5.1. *For any* ρ, ℓ *such that* $(\ell+\rho+10)2^{2\rho}/p(\lambda)^3 + \rho/p(\lambda) = \mathtt{ngl}(\lambda)$*, Construction 5.1 satisfies completeness.*

Proof. This argument is as in Lemma 4.1 with the polynomials now being of degree at most $\ell+\rho+10$. □

Security: Let $\mathcal{M}$ be a process that forms obfuscations of the type in Construction 5.1. We can create $\mathcal{M}$ as so: Sample a ℓ-long vector $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_{p(\lambda)}^{\ell}$. Then, sample an $\ell \times \ell$ random matrix $\mathbf{B} \leftarrow \mathbb{Z}_{p(\lambda)}^{\ell\times\ell}$. Define the matrix $\mathbf{D} \in \mathbb{Z}_{p(\lambda)}^{(\ell+5)\times(\ell+\rho+9)}$ as:

$$\mathbf{D} = \left(\begin{array}{c|c|ccccc|ccccc} \mathbf{A} & \mathtt{ad} & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ \hline 0^{1\times\ell} & 0^{1\times\rho} & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0^{1\times\ell} & 0^{1\times\rho} & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0^{1\times\ell} & 0^{1\times\rho} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0^{1\times\ell} & 0^{1\times\rho} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ \hline \mathbf{B} & 0^{\ell\times\rho} & 0^{\ell\times\rho} & 0^{\ell\times1} & 0^{\ell\times1} & 0^{\ell\times1} & 0^{\ell\times1} & 0^{\ell\times1} & 0^{\ell\times1} & 0^{\ell\times1} & 0^{\ell\times1} \end{array}\right). \tag{9}$$

It is clear that, for all settings of $\mathbf{A}, \mathbf{B}$, and $\mathtt{ad}$, the first five rows of $\mathbf{D}$ have rank at least 5. Clearly, the last ℓ rows cannot be in the span of the first 5 so, it suffices to show that $\mathbf{B}$ has full rank with overwhelming probability. Because $\mathbf{B} \in \mathbb{Z}_{p(\lambda)}^{\ell\times\ell}$ is a random matrix sampled independently of $\mathbf{A}$,

$$\Pr[\mathtt{rank}(\mathbf{B}) = \ell] \geq 1 - \ell/p(\lambda) = 1 - \mathtt{ngl}(\lambda)$$

see [26]. So, for all $\mathtt{ad}$, the matrix $\mathbf{D}$ has rank at least $\ell+5$ with probability $1-\mathtt{ngl}(\lambda)$. By Lemma 4.2, the **Point** construction detailed there satisfies VBB security. Finally, we note that the construction of $\mathbf{D}$ results in this **Point** program matching Construction 5.1 exactly.

Nonmalleability: In order to prove input nonmalleability, we will use a similar structure to Theorem 4.3. That is, we will first show that, for any low degree polynomial, it is difficult to maul the underlying val, even if $\mathtt{ad}$ may change arbitrarily. Then, assuming val is unchanged, we show that it is difficult to output a valid obfuscation with $\mathtt{ad}' \neq \mathtt{ad}$.

Theorem 5.2. *Let* λ *be a security parameter Let* $\{\mathcal{X}_\lambda\}$ *be a well-spread distribution ensemble and let* $\tau, \ell, \rho \in \mathbb{Z}^+$ *be parameters that are* $\mathtt{poly}(\lambda)$*. Let* $\mathcal{F}_{poly}$ *be the ensemble of functions* f_λ *where* f_λ *is the set of non-constant, non-identity polynomials in* $\mathbb{Z}_{p(\lambda)}[x]$ *with degree at most* τ*. Suppose that Assumption 3.2 holds for maximum power* $\psi = \tau(\ell+\rho+6)$*. Then, the obfuscator in Construction 5.1 is input-nonmalleable for* $\mathcal{F}_{poly}$*, distribution ensemble* $\{\mathcal{X}_\lambda\}$*, and* $\mathtt{ad}$ *of length* ρ*.*

[6] In Section 5.4 we show how to encode a long CRS as long as one can support associated data of size $\Theta(\log \lambda)$.

Proof. As before, we adopt a two-step proof structure:

1. **Lemma 5.2.** Even given the ℓ point obfuscations as extra advice, and given the power to arbitrarily maul ad (even to arbitrary vectors), no adversary can maul the underlying val in the underlying nonmalleable point obfuscation with associated data.

2. **Lemma 5.3.** Assuming that val is not tampered, then our construction additionally prevents tampering to binary ad.

As their proof structure follows in a technically similar manner as before. We state them below.

Lemma 5.2. *Let λ be a security parameter. Let $\{\mathcal{X}_\lambda\}$ be a well-spread distribution ensemble and let x be sampled from it. Let $\tau, \ell, \rho \in \mathbb{Z}^+$ be $\mathtt{poly}(\lambda)$. Let $\mathcal{F}_{poly}$ be the set of non-constant, non-identity polynomials in $\mathbb{Z}_{p(\lambda)}[x]$ with maximum degree τ. For a vector $r_\rho, \ldots, r_1 \in \mathbb{Z}_{p(\lambda)}$, define the polynomial $P(x) \stackrel{def}{=} r_\rho x^\rho + r_{\rho-1}x^{\rho-1} + \ldots + r_1 x$. Let iLock be as in Construction 5.1, except replacing $\left[p_{0,\mathtt{ad},c_{\vec{0}}}(\mathsf{val})\right]_g$ with*

$$\left[c_{0,1}\mathsf{val} + \sum_{i=1}^{\ell} c_{0,i+1}\mathsf{val}^{i+1} + P(\mathsf{val})\mathsf{val}^{\ell+1} + \sum_{i=1}^{5} \mathsf{val}^{\ell+\rho+1+i}\right]_g$$

Suppose Assumption 3.2 holds for maximum power $\tau(\ell+\rho+6)$. Then, for any adversary $\mathcal{A}$ for input nonmalleability for iLock, the probability that $f \in \mathcal{F}_{poly}$ is $\mathtt{ngl}(\lambda)$.

Proof. We look to contradict Assumption 3.2. Consider a mauling adversary $\mathcal{A}$ that, given a distribution $\mathcal{X}_\lambda$, vector $\mathsf{ad}_1, ..., \mathsf{ad}_\rho$, and an obfuscation iLock of the form in Construction 5.1, can output a new obfuscation of $f(x)$ for $f \in \mathcal{F}_{poly}$ with non-negligible probability. In particular, $\mathcal{A}$ may output an obfuscation with arbitrary polynomial $P'(x)$ with degree at most ρ. Let τ be maximum degree of f.

For k_i, define $h_i = k_i x + x^i$. We will build an adversary $\mathcal{B}$ which, given $\{k_i, [h_i]_g\}_{i=2,\ldots,\tau(\ell+\rho+10)}$ and access to $\mathcal{A}$, can recover g^x with non-negligible probability. As before, we will separately consider the cases where $\tau > 1$ and $\tau = 1$.

In either case, $\mathcal{B}$ will prepare the obfuscation in a similar manner. In the proof of security, we see how an obfuscation of iLock can be formed from the ensemble of k_i and $[h_i]_g$. Specifically, we sample $\mathbf{A} \stackrel{\$}{\leftarrow} \mathbb{Z}^\ell_{p(\lambda)}$ and $\mathbf{B} \stackrel{\$}{\leftarrow} \mathbb{Z}^{\ell\times\ell}_{p(\lambda)}$, and using the coefficients $\vec{P}$, we are able to form $\mathbf{D}$ in a similar way to Equation 9, except replacing the entries of ad with $\vec{P}$.

With all of this, $\mathcal{B}$ gives the resultant iLock, along with the coefficients $\vec{P}$, to $\mathcal{A}$. We note here that the constructed iLock corresponds to the all-1 key, i.e., $\mathsf{key} = \vec{1}$. Not only is this done as it gives the most information on x to $\mathcal{A}$, but we see that $\mathcal{A}$ may also create terms corresponding to 0-bits independently, if such terms would prove beneficial.

Then, $\mathcal{A}$ returns, with non-negligible probability,

$$\left(f, \vec{P}', \mathsf{unlockPoint}_{\vec{P}'} = \begin{bmatrix} \vec{c_0}', & \left[p_{0,\vec{P}',\vec{c_0}}(f(\mathsf{val}))\right]_g \\ c'_{0,\ell+2}, & \left[p_{0,1,c_{0,\ell+2}}(f(\mathsf{val}))\right]_g \\ c'_{0,\ell+3}, & \left[p_{0,2,c_{0,\ell+3}}(f(\mathsf{val}))\right]_g \\ c'_{0,\ell+4}, & \left[p_{0,3,c_{0,\ell+4}}(f(\mathsf{val}))\right]_g \\ c'_{0,\ell+5}, & \left[p_{0,4,c_{0,\ell+5}}(f(\mathsf{val}))\right]_g \\ \{\vec{c_i}', & \left[p^*_{\vec{c_i}'}(z'_i)\right]_g\}_{i=1}^{\ell} \end{bmatrix}\right)$$

where $f \in \mathcal{F}_{poly}$, P' is of degree at most ρ, z'_i may be either $f(\mathsf{val})$ or an arbitrary point for each i, and the polynomials are as defined in Equations 3, 4, 5, 6, 7, and 8, with emphasis that each $\mathtt{ad}_i$ in Equation 3 is replaced by P'_i here. For the purposes of nonmalleability, we are only interested in $f, \vec{P}', \vec{c_0}'$, and $g' = \left[p_{0,\vec{P}',\vec{c_0}}(f(\mathsf{val}))\right]_g$. The rest of $\mathcal{A}$'s output is discarded.

Non-Linear Case: We first consider the case where $\tau > 1$, i.e., when f is non-linear. This argument follows as in Lemma 4.3 with different indices to reflect the augmented construction. Define the polynomial $G(x)$ as the coefficients of:

$$G(x) = \sum_{i=0}^{\tau(\ell+\rho+6)} G_i x^i \stackrel{def}{=} \sum_{i=1}^{\ell+1} a_i f(x)^i + f(x)^{\ell+1} P'(f(x)) + f(x)^{\ell+\rho+1} \sum_{i=1}^{5} (f(x))^i.$$

That is, $G(x)$ is the polynomial $\mathcal{A}$ claims they have mauled the exponent of the term g'_1 to. Note that the coefficients of G are computable by $\mathcal{B}$. Let h^* denote the exponent of g'_1. In order for the $\mathcal{A}$ to succeed in tampering, it must be the case that $G(x) = h^*$ with noticeable probability (over the choice of x and randomness of $\mathcal{A}_P$). $\mathcal{B}$ computes and returns

$$\left(g'_1 \left([G_0]_g \cdot \prod_{i=2}^{\tau(\ell+\rho+6)} [h_i]_g^{G_i}\right)^{-1}\right)^{1/\left(G_1 - \sum_{i=2}^{\tau(\ell+\rho+6)} k_i G_i\right)}.$$

When $h^* = \sum_{i=0}^{\tau(\ell+\rho+6)} G_i x^i$, then

$$\begin{aligned}\left(g'_1 \left([G_0]_g \cdot \prod_{i=2}^{\tau(\ell+\rho+6)} G_i [h_i]_g^{G_i}\right)^{-1}\right) &= \left[\sum_{i=0}^{\tau(\ell+\rho+6)} G_i x^i - G_0 - \sum_{i=2}^{\tau(\ell+\rho+6)} G_i (k_i x + x^i)\right]_g \\ &= \left[x\left(G_1 - \sum_{i=2}^{\tau(\ell+\rho+6)} k_i G_i\right)\right]_g\end{aligned}$$

Thus, in the case that $\mathcal{A}$ returns the correct value and $G_1 - \sum_{i=2}^{\tau(\ell+\rho+6)} k_i G_i \neq 0$, $\mathcal{B}$ computes and returns the correct value. Since $f(x)$ is of degree τ, $G_{\tau(\ell+\rho+6)}$ must be nonzero. $\mathcal{A}$'s view is independent of $k_{\tau(\ell+\rho+6)}$. So, the probability that the sum

$$\Pr_{k_{\tau(\ell+\rho+6)}}\left[\sum_{i=2}^{\tau(\ell+\rho+6)} k_i G_i = G_1\right] = \frac{1}{p(\lambda) - 1}.$$

So, $\mathcal{B}$ returns the correct value with probability $1/\mathtt{poly}(\lambda) - 1/(p(\lambda) - 1) = 1/\mathtt{poly}(\lambda)$ contradicting Assumption 3.2.

Linear Case This argument follows similarly to the linear case in Lemma 4.3 with different indices to reflect the augmented construction. This time, $\mathcal{B}$ receives the ensemble of $f, \vec{P}'$, and $\mathsf{unlockPoint}_{\vec{P}'}$ as

above, but now $f = \alpha x + \beta$. As before, $\mathcal{B}$ only considers $\alpha, \beta, \vec{P}', \vec{c_0}'$, and $g' = \left[p_{0,\vec{P}',\vec{c_0}}(f(\mathsf{val}))\right]_g$ from here. It computes the coefficients of the polynomial G as

$$G(x) = \sum_{i=0}^{\ell+\rho+6} G_i x^i \stackrel{def}{=} \sum_{i=1}^{\ell+1} a_i f(x)^i + f(x)^{\ell+1} P'(f(x)) + f(x)^{\ell+\rho+1} \sum_{i=1}^{5} (f(x))^i.$$

as in the nonlinear case. In this case, $\mathcal{B}$ computes and outputs:

$$\left(g_1' \left([G_0]_g \cdot \prod_{i=2}^{\ell+\rho+6} [h_i]_g^{G_i} \right)^{-1} \right)^{1/\left(G_1 - \sum_{i=2}^{\ell+\rho+6} k_i G_i\right)}.$$

We see $\mathcal{B}$'s computation evaluates to g^x unless

$$G_1 - \sum_{i=2}^{\ell+\rho+6} k_i G_i = 0.$$

To see that this happens with negligible probability, for the coefficient x^1 of the first group element for the i-th key bit polynomial of iLock received is c_i for values c_i that depend on k_j for $j < \ell + 1$. So, $\mathcal{A}$ sees no information about the values $\{k_i\}_{i=\ell+\rho+2}^{\ell+\rho+6}$ except for that of the coefficient of x^1 in the first term $\vec{c_0}'$ Without loss of generality, we assume that an adversary knows the values $k_2, ..., k_{\ell+\rho+1}$. The adversary must find α, β, γ such that

$$\sum_{i=\ell+\rho+2}^{\ell+\rho+6} (\alpha x + \beta)^i = \gamma \sum_{i=\ell+\rho+2}^{\ell+\rho+6} x^i.$$

Setting $\delta = \ell+\rho+6$, by application of Lemma 4.4 the only solutions where $f \in \mathcal{F}_{poly}$ are when $\delta \in \{-5, -6\}$ mod $p(\lambda)$. Since $\ell + \rho = \mathtt{poly}(\lambda)$, for large enough λ one can be sure that $\delta \notin \{-5, -6\} \mod p(\lambda)$. So, the only functions that $\mathcal{A}$ can maul to with noticeable probability are the constant and identity functions, neither of which are in $\mathcal{F}_{poly}$. This means that $\mathcal{A}$ returns a solution where $G_1 - \sum_{i=\ell+\rho+2}^{\ell+\rho+6} k_i G_i = 0.$ with negligible probability. So, with non-negligible probability, $\mathcal{B}$ returns some g^x. This is a contradiction of Assumption 3.2 and completes the proof of Lemma 5.2. □

Lemma 5.3. *Let λ be a security parameter. Let $\{\mathcal{X}_\lambda\}$ be a well-spread distribution ensemble and let $\rho, \ell \in \mathbb{Z}^+$ be $\mathtt{poly}(\lambda)$. Let $\mathcal{F}_{poly}$ be the ensemble of functions f_λ where f_λ is the set of non-constant, non-identity polynomials in $\mathbb{Z}_{p(\lambda)}[x]$ with degree at most τ. Let iLock be as in Construction 5.1. Suppose Assumption 3.2 holds for maximum power $\ell+\rho+6$. Consider a PPT adversary $\mathcal{A}$ for input nonmalleability in Definition 5.1 which outputs $f \in \mathcal{F}_{poly}$ with negligible probability. Then, the probability that $\mathcal{A}$ outputs the identity function f and some $\mathtt{ad}' \neq \mathtt{ad}$ with coefficients in $\{0,1\}$ is $\mathtt{ngl}(\lambda)$.*

Proof. Let $\mathtt{ad} \in \{0,1\}^\rho$ be arbitrary. We show how, given some $\mathcal{A}_{\mathtt{ad}}$ that is capable of producing some $\mathtt{ad}' \neq \mathtt{ad}$ with noticeable probability, we may build an algorithm $\mathcal{B}$ that finds g^x given the ensemble $\{k_i, [k_i x + x^i]_g\}_{i=2,...,\ell+\rho+6}$. First, $\mathcal{B}$ receives $\mathcal{X}_\lambda$ from $\mathcal{A}$. Then, $\mathcal{B}$ prepares the obfuscations from $\{k_i, [k_i x + x^i]_g\}_{i=2,...,\ell+\rho+6}$ as in Equation 9 and Lemma 5.2.

We now assume that with non-negligible probability, $\mathcal{A}$ outputs

$$\left(f, \vec{ad}', \mathsf{unlockPoint}_{\vec{ad}'} = \begin{bmatrix} \vec{c_0}', & \left[p_{0,\vec{ad}',\vec{c_0}}(f(\mathsf{val}))\right]_g \\ c'_{0,\ell+2}, & \left[p_{0,1,c_{0,\ell+2}}(f(\mathsf{val}))\right]_g \\ c'_{0,\ell+3}, & \left[p_{0,2,c_{0,\ell+3}}(f(\mathsf{val}))\right]_g \\ c'_{0,\ell+4}, & \left[p_{0,3,c_{0,\ell+4}}(f(\mathsf{val}))\right]_g \\ c'_{0,\ell+5}, & \left[p_{0,4,c_{0,\ell+5}}(f(\mathsf{val}))\right]_g \\ \{\vec{c_i}', & \left[p^*_{\vec{c_i}'}(z'_i)\right]_g\}_{i=1}^{\ell} \end{bmatrix}\right)$$

where the following conditions hold:

- f is the identity function,
- $|\mathsf{ad}| = \rho$, and each entry in ad' is in $\{0,1\}$ such that $\mathsf{ad}' \neq \mathsf{ad}$ as vectors,
- the obfuscation is correct.

$\mathcal{B}$ only considers $\vec{c_0}'$ and $g'_1 = \left[p_{0,\vec{ad}',\vec{c_0}}(\mathsf{val})\right]_g$. Let $P'(x) = \sum_{i=1}^{\rho} \mathsf{ad}'_i x^i$. Define the polynomial $G(x)$ as the coefficients of:

$$G(x) = \sum_{i=0}^{\ell+\rho+6} G_i x^i \stackrel{def}{=} \sum_{i=1}^{\ell+1} c'_{0,i} x^i + x^{\ell+1} P'(x) + x^{\ell+\rho+1} \sum_{i=1}^{5} x^i.$$

Note that the coefficients of P' and G are computable by $\mathcal{B}$. Let h^* denote the exponent of g'_1. It must be the case that $G(x) = h^*$ with noticeable probability. As such, $\mathcal{B}$ computes and outputs:

$$\left(g'_1 \left(\prod_{i=2}^{\ell+\rho+6} [h_i]_g^{G_i}\right)^{-1}\right)^{1/\left(G_1 - \sum_{i=2}^{\ell+\rho+6} k_i G_i\right)}.$$

$\mathcal{B}$'s computation evaluates to g^x unless

$$G_1 - \sum_{i=2}^{\ell+\rho+6} k_i G_i = 0.$$

For $\mathsf{ad}' \neq \mathsf{ad}$ there must be at least one bit i where $\mathsf{ad}_i = 1 \wedge \mathsf{ad}'_i = 0$ or $\mathsf{ad}'_i = 0 \wedge \mathsf{ad}_i = 1$. Furthermore note that $\mathsf{ad}'_i = G_{\ell+1+i}$ for $i = 1, ..., \rho$. We now split our analysis into these two cases (which are identical up to change of indices of the proof of Lemma 4.5):

$\exists i$ such that $\mathsf{ad}_i = 0 \wedge \mathsf{ad}'_i = 1$. Let i^* denote one such location. Recall that by definition $G_{i^*} = 1$. Furthermore, $\mathcal{A}$'s view is independent of k_{i^*} since $\mathsf{ad}_i = 0$. So, the probability that the sum

$$\Pr_{k_{i^*}}\left[\left(G_1 - \sum_{i=2, i\neq i^*}^{\ell+\rho+6} k_i G_i\right) = k_{i^*}\right] = \frac{1}{p(\lambda) - 1}.$$

$\forall i, \mathtt{ad}_i = 0 \rightarrow \mathtt{ad}'_i = 0$ **and** $\exists i$ **such that** $\mathtt{ad}_i = 1 \wedge \mathtt{ad}'_i = 0$**.** Let i^* denote one such index. All information about k_{i^*} known by $\mathcal{A}_{\mathtt{ad}}$ is contained in the two values (corresponding to $\mathtt{ad}_1$ and when $\mathtt{ad} = 0$)

$$v_1 = \sum_{i=\ell+2}^{\ell+\rho+1} r_i k_i + \sum_{i=\ell+\rho+2}^{\ell+\rho+6} k_i$$
$$v_2 = \sum_{i=\ell+\rho+2}^{\ell+\rho+6} k_i$$

since $\{k_i\}_{3\ell+\rho+2 \leq i \leq 3\ell+\rho+6}$ are uniformly distributed, one has

$$\forall s, \Pr\left[k_{i^*} = s | v_1 \wedge v_2\right] = \Pr[k_{i^*} = s] = \frac{1}{p(\lambda)}.$$

Thus, as before, one has

$$\Pr_{k_{i^*}}\left[\left(G_1 - \sum_{i=2}^{\ell+\rho+6} k_i G_i\right) = 0\right] = \frac{1}{p(\lambda) - 1}.$$

Thus, in both cases,

$$\exists i^*, \Pr_{k_{i^*}}\left[\left(G_1 - \sum_{i=2}^{\ell+\rho+6} k_i G_i\right) = 0\right] = \frac{1}{p(\lambda) - 1}.$$

This completes the proof of Lemma 5.3 by noting that these cases are exhaustive, and that for any given $\mathtt{ad}$ we may construct the reduction in this way. □

Finally, we conclude by noting that application of Lemma 5.2 and Lemma 5.3 yields Theorem 5.2.

□

□

5.2 Adding Key Nonmalleability

We now show that the input nonmalleable digital locker with associated data suffices to build a fully nonmalleable digital locker for the same function class. Let iLock be such an object and $\Pi = (\mathsf{Setup}, P, V)$ be some appropriate non-interactive proof system (described in Section 5.3) using a $\mathtt{crs}$ of length ρ for the following language that proves well-formness of iLock:

$$\mathcal{L} = \{\mathsf{iUnlock} : \exists(\mathsf{val}, \mathtt{crs}, \mathsf{key}, r) \text{ such that } \mathsf{iUnlock} = \mathsf{iLock}(\mathsf{val}, \mathtt{crs}, \mathsf{key}; r)\}$$

Construction 5.2. *For security parameter $\lambda \in \mathbb{N}$, let $\mathcal{F} : \mathsf{D}^\lambda \rightarrow \mathsf{D}^\lambda$ be a family of functions, let $\rho, \ell \in \mathbb{N}$ be parameters, $\mathcal{X}$ be a family of distributions over D^λ. Suppose that*

1. iLock *is a $(\mathcal{F}_{\mathtt{poly}}, \mathcal{X}, \rho, \ell)$-input-nonmalleable digital locker with associated data with associated obfuscation verifier $\mathsf{V}_{\mathtt{obf\,Input}}$, and*
2. $\Pi = (\mathsf{Setup}, P, V)$ *is an NIZK system for the language $\mathcal{L}$ with short non-tamperable CRS.*[7] *We formally define this property and show a generic construction in Section 5.3.*

Then define $(\mathsf{lock}, \mathsf{unlock}, \mathsf{V}_{\mathtt{obf}})$ *as in Figure 2.*

$\mathsf{lock}(\mathsf{val}, \mathsf{key})$:

1. Sample $(\mathtt{crs} = (\mathtt{crs}_1, \mathtt{crs}_2), \mathsf{TK}, \mathsf{EK}) \leftarrow \mathsf{Setup}(1^\lambda)$.
2. Compute $\mathsf{iUnlock} \leftarrow \mathsf{iLock}(\mathsf{val}, \mathtt{crs}_1, \mathsf{key}; r)$.
3. Compute $\pi \leftarrow P(\mathsf{unlock}'; \mathsf{val}, \mathsf{key}, r, \mathtt{crs})$.
4. Output $(\mathsf{iUnlock}, \pi, \mathtt{crs})$.

$\mathsf{V}_{\mathtt{obf}}(\mathsf{iUnlock}, \pi, \mathtt{crs})$:

1. If $\mathsf{V}_{\mathtt{obf\,Input}}(\mathsf{iUnlock}) = 0$ output 0.
2. If $\mathtt{crs} = \vec{0}$ output 0.
3. if $V(\pi, \mathsf{unlock}', \mathtt{crs}) = 0$ output 0.
4. Output 1.

$\mathsf{unlock}(\mathsf{val}, \mathsf{iUnlock}, \pi, \mathtt{crs} = (\mathtt{crs}_1, \mathtt{crs}_2))$: Output $\mathsf{iUnlock}(\mathsf{val}, \mathtt{crs}_1)$

Figure 2: Digital Locker Construction.

Theorem 5.3. *Let notation be as in Construction 5.2. Suppose that*

1. iLock *is a* $(\mathcal{F}_{\mathtt{poly}}, \mathcal{X}, \rho, \ell)$*-input-nonmalleable digital locker with associated data with associated obfuscation verifier* $\mathsf{V}_{\mathtt{obf\,Input}}$*, and*
2. $\Pi = (\mathsf{Setup}, P, V)$ *is a true simulation extractable non-interactive zero knowledge proof system as described in Section 5.3,*
3. *That every function* $f \in \mathcal{F}$ *is entropy preserving; i.e., for any well-spread* X*,* $f(X)$ *is also well-spread.*

Then $\mathsf{lock}, \mathsf{unlock}$ *is a* $(\mathcal{F}, \mathcal{X}, n)$*-nonmalleable digital locker.*

Proof of Theorem 5.3. Following Definition 3.3, we need to prove completeness, soundness, and nonmalleability. Completeness can be easily verified, so we just focus on the non-trivial parts, i.e., proof of soundness and nonmalleability.

Soundness To prove soundness, we first observe that according to Theorem 3.1, for this class of circuits being obfuscated DI is equivalent to VBB, so for the rest of the proof, we focus on proving the DI. We prove soundness by contradiction. Suppose there exists a PPT adversary $\mathcal{A}$, a key $\mathsf{key} \in \{0,1\}^\ell$, and a well-spread distribution X such that

$$|\Pr_{\mathsf{val} \leftarrow \mathcal{X}}[\mathcal{A}(\mathsf{lock}(\mathsf{val}, \mathsf{key})) = 1] - \Pr_{r \xleftarrow{\$} \mathsf{D}^\lambda}[\mathcal{A}(\mathsf{lock}(r, \mathsf{key})) = 1] > \epsilon$$

for some non-negligible ϵ, then there exists an adversary $\mathcal{B}$ that breaks the DI security of the input-nonmalleable digital locker. This reaches a contradiction.

$\mathcal{B}$ receives the distribution X samples $(\mathtt{crs}_1, \mathtt{crs}_2, \mathsf{TK}, \mathsf{EK}) \leftarrow \mathsf{Setup}(1^\lambda)$ for the proof system, and sets associated data as $\mathtt{crs}_1$. $\mathcal{B}$ sends this distribution to the reduction for iLock with the input distribution the same X, and associated data, $\mathtt{crs}_1$. The reduction samples some $\mathsf{val} \leftarrow X$ or uniform r. $\mathcal{B}$ receives $\mathsf{iUnlock}$. Next $\mathcal{B}$ creates a simulated π. It sends $\mathsf{iUnlock}, \pi, \mathtt{crs}$ to $\mathcal{A}$ and outputs $\mathcal{A}$'s decision.

Clearly, if val is from the distribution X, then the reduction has simulated an indistinguishable $\mathsf{lock}(\mathsf{val}, \mathsf{key})$ (assuming the simulated proof π is indistinguishable), or otherwise, $\mathsf{lock}(r, \mathsf{key})$. That is, in both cases, the obfuscation is properly prepared assuming the indistinguishability of the simulated

[7] That is, $\mathtt{crs}$ can be split into $(\mathtt{crs}_1, \mathtt{crs}_2)$ where $\mathtt{crs}_1$ has length independent of the language, i.e., $O(\lambda)$, and only $\mathtt{crs}_1$ is required to be non-tamperable. $\mathtt{crs}_2$ cannot be modified (computationally infeasible) given the original $\mathtt{crs}_1$.

proof. Thus, the advantage of the adversary $\mathcal{A}$ translates to the advantage of $\mathcal{B}$ in breaking the DI of the nonmalleable point obfuscation. By the equivalence of DI and VBB of point obfuscation, this breaks the soundness of the nonmalleable point obfuscation.

Nonmalleability Now we prove nonmalleability. As before, we prove by contradiction. Suppose there exists a PPT adversary $\mathcal{A}$ and $\mathsf{key} \in \{0,1\}^\ell$, a well-spread distribution X such that $\mathcal{A}$ breaks the nonmalleability experiment with non-negligible probability ϵ. Then there exists an adversary $\mathcal{B}$ that breaks the nonmalleability of the underlying $\mathsf{iLock}(\cdot)$.

$\mathcal{B}$ follows exactly the same procedure in preparing the input to the adversary $\mathcal{A}$ as in soundness proof above. Now $\mathcal{A}$ would return a triple $(C, f, \mathtt{crs}^* = (\mathtt{crs}_1^*, \mathtt{crs}_2^*))$ that passes the checking conditions with a non-negligible probability ϵ. Assume C is different from the original obfuscation given to $\mathcal{A}$ (as we don't allow identity tampering). $\mathcal{B}$ does the following:

- If the $\mathtt{crs}_1$ is modified to a different $\mathtt{crs}_1^*$, then the reduction just outputs the $C, f, \mathtt{crs}_1^*$ which correspond to a tamper according to nonmalleability of $\mathsf{iLock}(\cdot)$.
- If the $\mathtt{crs}_1$ is kept intact but $\mathtt{crs}_2$ is modified to a different $\mathtt{crs}_2^*$, then this breaks the underlying NIZK as it is computationally infeasible to obtain a consistent but different $\mathtt{crs}_2^*$.
- If the $\mathtt{crs} = \mathtt{crs}^*$ in C is intact yet the statement-proof pair is modified, then $\mathcal{B}$ runs the witness extractor to extract a valid witness, i.e., val' used to generate C. As the input obfuscated circuits received by $\mathcal{B}$ are properly prepared by the challenger, the simulated proof given to the adversary $\mathcal{A}$ is with respect to a true statement. In this case, the notion of true simulation extractability allows $\mathcal{B}$ to extract a valid witness by running the extractor. Thus, given $\mathsf{val}' = f(\mathsf{val})$. $\mathcal{B}$ can prepare an obfuscation (with an arbitrary associated data of val'), breaking the nonmalleability of $\mathsf{iLock}(\cdot)$.

Since $\mathcal{A}$ wins the nonmalleable experiment with a non-negligible probability, one of the above case must happen with a non-negligible probability. This would imply the contradiction we expect. The above two arguments complete the proof of Theorem 5.3.

□

5.3 The Building Block – True Simulation Extractable NIZK

In this section, we present the building block used in Construction 5.2 – true simulation extractable NIZK. The notion was introduced by Dodis et al. [33] as a relaxation of *all* simulation extractable NIZK. We describe the notion in what follows.

Definition 5.2. *Let R be an NP relation on pairs (x, w) with corresponding language $L_R = \{x : \exists w \text{ such that } (x, w) \in R\}$. A true-simulation extractable non-interactive zero-knowledge (NIZK) argument for a relation R consists of three algorithms* $(\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify})$ *with the following syntax:*

- $(\mathtt{crs}, \mathsf{TK}, \mathsf{EK}) \leftarrow \mathsf{Setup}(1^\lambda)$*: creates a common reference string* $\mathtt{crs}$*, a trapdoor* TK*, and an extraction key* EK*.*
- $\pi \leftarrow \mathsf{Prove}(\mathtt{crs}, x, w)$*: creates an argument π that $R(x, w) = 1$.*
- $0/1 \leftarrow \mathsf{Verify}(\mathtt{crs}, x, \pi)$*: verifies whether or not the argument π is correct.*

For presentation simplicity, we omit $\mathtt{crs}$ *in the* Prove *and* Verify*. We require that the following three basic properties hold:*

- ***Completeness.*** *For any* $(x, w) \in R$*, if* $(\mathtt{crs}, \mathsf{TK}, \mathsf{EK}) \leftarrow \mathsf{Setup}(1^\lambda)$*,* $\pi \leftarrow \mathsf{Prove}(x, w)$*, then* $\mathsf{Verify}(x, \pi) = 1$*.*
- ***Soundness.*** *For any PPT adversary* $\mathcal{A}$*, the following probability is negligible: for* $(\mathtt{crs}, \mathsf{TK}, \mathsf{EK}) \leftarrow \mathsf{Setup}(1^\lambda)$*,* $(x^*, \pi^*) \leftarrow \mathcal{A}(\mathtt{crs})$ *such that* $x^* \notin L_R$ *but* $\mathsf{Verify}(x^*, \pi^*) = 1$*.*
- ***Composable Zero-knowledge.*** *There exists a PPT simulator* $\mathcal{S}$ *such that for any PPT* $\mathcal{A}$*, the advantage (the probability* $\mathcal{A}$ *wins minus one half) is negligible in the following game.*
  - *The challenger samples* $(\mathtt{crs}, \mathsf{TK}, \mathsf{EK}) \leftarrow \mathsf{Setup}(1^\lambda)$ *and sends* $(\mathtt{crs}, \mathsf{TK})$ *to* $\mathcal{A}$
  - $\mathcal{A}$ *chooses* $(x, w) \in R$ *and sends to the challenger.*
  - *The challenger generates* $\pi_0 \leftarrow \mathsf{Prove}(x, w)$*,* $\pi_1 \leftarrow \mathsf{Sim}(x, \mathsf{TK})$*, and then samples a random bit* $b \leftarrow \{0, 1\}$*. Then he sends* π_b *to* $\mathcal{A}$*.*
  - $\mathcal{A}$ *outputs a guess bit* b'*, and wins if* $b' = b$*.*
- ***Extractibility.*** *Additionally, true simulation extractability requires that there exists a PPT extractor* Ext *such that for any PPT adversary* $\mathcal{A}$*, the probability* $\mathcal{A}$ *wins is negligible in the following game:*
  - *The challenger samples* $(\mathtt{crs}, \mathsf{TK}, \mathsf{EK}) \leftarrow \mathsf{Setup}(1^\lambda)$ *and sends* $\mathtt{crs}$ *to* $\mathcal{A}$*.*
  - $\mathcal{A}$ *is allowed to make oracle queries to the simulation algorithm* $\mathsf{Sim}'((x, w), \mathsf{TK})$ *adaptively.* Sim' *first checks if* $(x, w) \in R$ *and returns* $\mathsf{Sim}(x, TK)$ *if that is the case.*
  - $\mathcal{A}$ *outputs a tuple* x^*, L^*, π^**.*
  - *The challenger runs the extractor* $w^* \leftarrow \mathsf{Ext}(L^*, (x^*, \pi^*), \mathsf{EK})$*.*
  - $\mathcal{A}$ *wins if (1) the pair* (x^*, L^*) *was not part of the simulator query, (2) the proof* π^* *verifies, and (3)* $R(x^*, w^*) = 0$*.*

Briefly speaking, a true simulation extractable NIZK requires that the adversary can only query the simulation oracle only on true statements, whereas all simulation extractability allows the adversary to query on any (perhaps false) statement. As shown by the work [33], the true simulation extractable NIZK can be constructed in a fairly simple way as summarized by the following theorem.

Theorem 5.4 ([33])**.** *Assume that there exists a CCA2 encryption and a regular NIZK argument for NP languages, then there exists a true simulation extractable NIZK for NP languages.*

The work [33] showed how to instantiate the building blocks under the SXDH assumption over bilinear groups. There is plausible evidence that the regular NIZK can be constructed without the need of pairing groups, c.f. [29], under some non-standard assumptions.

5.4 NIZK with Short Non-Tamperable CRS

The generic use of the NIZK from Dodis et al. [33] requires long CRS that would depend on the language being proved, and this is a general fact for NIZKs. In our application, however, this poses a challenge when we combine this with our non-malleable obfuscation with associate data. Particularly, the correctness of Theorem 4.1 requires a group that has a length larger than that of associated data. We notice that the language $\mathcal{L}$ used in Construction 5.2 requires a long CRS, as the statement and the witness are long. So,

putting CRS as the associated data in the non-malleable digital locker would require a significantly larger group, which is undesirable.

To handle this technical subtlety, we present a simple transformation from any NIZK into one whose CRS has the following structure: $\mathtt{crs} = (\mathtt{crs}_1, \mathtt{crs}_2)$, where only $\mathtt{crs}_1$ is short and non-tamperable, $\mathtt{crs}_2$ can be arbitrarily long but cannot be tampered consistently (computationally infeasible) as long as $\mathtt{crs}_1$ is kept intact. In this way, we can put $\mathtt{crs}_1$ as the associated data into our non-malleable digital locker, and keep $\mathtt{crs}_2$ public, as we presented in the prior section. Thus, the underlying group of the non-malleable obfuscation can be significantly smaller.

To achieve this, given any $\mathtt{crs}'$ from the underlying NIZK, we define a new NIZK which is essentially the same as the original one, except in the CRS generation: first it samples a collision resistant hash function h and computes $z = h(\mathtt{crs})$. It outputs $\mathtt{crs} = (\mathtt{crs}_1 = (h, z), \mathtt{crs}_2 = \mathtt{crs}')$ as the new CRS. The verifier will always check whether $h(\mathtt{crs}_2) = z$ and rejects immediately if it does not hold. The security (zero-knowledge, soundness) is not affected by $\mathtt{crs}_1$, as it can be generated just given $\mathtt{crs}'$.

6 Application to Fuzzy Extractors

In this section, we show that a nonmalleable digital locker suffices to build a robust fuzzy extractor [14, 16, 15, 34] when combined with a standard secure sketch. We note information-theoretic robust fuzzy extractor in the plain model or CRS models requires the source to have an entropy of at least half its length [36]. In this work, we consider computational robust fuzzy extractors in the plain model. We begin with a few definitions.

Definition 6.1 (Secure Sketch)**.** *Let λ be a security parameter. Let $\mathcal{W} = \mathcal{W}_\lambda$ be a family of random variables over metric space $(\mathcal{M}, \mathsf{dis}) = (\mathcal{M}_\lambda, \mathsf{dis}_\lambda)$. Then $\mathsf{SS}, \mathsf{Rec}$ is a $(\mathcal{M}, \mathcal{W}, t, \delta)$-secure sketch if the following hold:*

Correctness *For all $w, w' \in \mathcal{M}$ such that $\mathsf{dis}(w, w') \leq t$,*

$$\Pr[\mathsf{Rec}(w', \mathsf{SS}(w)) = w] \geq 1 - \delta.$$

Security *For all distributions $W \in \mathcal{W}$ it is true that*

$$\tilde{\mathrm{H}}_\infty(W|\mathsf{SS}(W)) \geq \omega(\log \lambda).$$

Definition 6.2 (Robust Fuzzy extractor)**.** *An $(\mathcal{M}, \mathcal{W}, \ell, t)$-computationally robust fuzzy extractor is a pair of PPT algorithms $(\mathsf{Gen}, \mathsf{Rep})$ where for all $w, w' \in \mathcal{M}$,*

- $(\mathsf{key}, \mathsf{pub}) \leftarrow \mathsf{Gen}(w)$*, where* $\mathsf{key} \in \{0,1\}^\ell$ *and* $\mathsf{pub} \in \{0,1\}^*$
- $\mathsf{key}' \leftarrow \mathsf{Rep}(\mathsf{pub}, w')$

such that the following properties are true:

- ***Correctness :*** *For all $w, w' \in \mathcal{M}$ such that $\mathsf{dist}(w, w') \leq t$,*

$$\Pr\left[\mathsf{key}' = \mathsf{key} \;\middle|\; (\mathsf{key}, \mathsf{pub}) \leftarrow \mathsf{Gen}(w), \mathsf{key}' \leftarrow \mathsf{Rep}(\mathsf{pub}, w')\right] \geq 1 - \mathtt{ngl}(\lambda).$$

- ***Security :*** *For any distribution $W \in \mathcal{W}$, and for $(\mathsf{key}, \mathsf{pub}) \leftarrow \mathsf{Gen}(W)$, for all PPT $\mathcal{A}$ there exists some $\mathtt{ngl}(\lambda)$ function such that*

$$|\Pr[\mathcal{A}(\mathsf{key}, \mathsf{pub}) = 1] - \Pr[\mathcal{A}(U_\ell, \mathsf{pub}) = 1] \leq \mathtt{ngl}(\lambda).$$

  where U_ℓ is a uniformly distributed random variable on $\{0,1\}^\ell$.

- ***Robustness:*** *Let $W, W' \in \mathcal{M}$ be (correlated) distributions such that*

$$\Pr_{(w,w') \leftarrow (W,W')}[\mathsf{dis}(w, w') \leq t] = 1$$

  and $W, W' \in \mathcal{W}$. For all $W, W' \in \mathcal{W}$ and for all adversaries $\mathcal{A}$, the advantage of $\mathcal{A}$ in the following experiment is at most $\mathtt{ngl}(\lambda)$:

  1. *Sample $(w, w') \leftarrow (W, W')$.*
  2. *Compute $(\mathsf{key}, \mathsf{pub}) \leftarrow \mathsf{FE.Gen}(w)$ and send it to $\mathcal{A}$.*
  3. *$\mathcal{A}$ outputs pub' and wins if $\mathsf{pub}' \neq \mathsf{pub}$ and $\mathsf{FE.Rep}(\mathsf{pub}', w') \notin \{\perp, \mathsf{key}\}$.*

Before introducing a common secure sketch which uses code syndromes we introduce the notation of $\mathsf{Wgt}(x) = \mathsf{dis}(x, 0)$ as the Hamming weight of x.

Definition 6.3 (Syndrome)**.** *Let $\mathbf{A} : \mathbb{F}_q^k \to \mathbb{F}_q^n$ be a $(n, k, d = 2t + 1)$-linear error code, then there exists a matrix $\mathsf{Syn} : \mathbb{F}_q^n \to \mathbb{F}_q^{n-k}$ with two properties:*

1. *For all values x where $\mathsf{Wgt}(x) \leq t$ the value $\mathsf{Syn}(x)$ is unique.*
2. *There is an efficient mapping from $s \in \mathbb{F}_q^{n-k}$ to the value x of weight at most t if one exists. Let Invert denote this mapping. If no such value exists then the output of Invert is $\perp$.*
3. *For any two values s, s' where $\mathsf{Wgt}(s), \mathsf{Wgt}(s'), \mathsf{Wgt}(s - s') \leq t$ it is true that*

$$\begin{aligned}\mathsf{Invert}(\mathsf{Syn}(s - s')) &= \mathsf{Invert}(\mathsf{Syn}(s) - \mathsf{Syn}(s')) \\ &= \mathsf{Invert}(\mathsf{Syn}(s)) - \mathsf{Invert}(\mathsf{Syn}(s')) = s - s'.\end{aligned}$$

Definition 6.4 (Syndrome Secure Sketch [11, 31, 35])**.** *Let $\mathcal{W} \in \mathbb{F}_q^n$ be the set of all distributions W where $\mathrm{H}_\infty(W) = (n-k)\log q + \omega(\log \lambda)$. Let Syn be the Syndrome of an $(n, k, d = 2t+1)$-error correcting code. Then define $\mathsf{SS}(w) = \mathsf{Syn}(w)$ and*

$$\mathsf{Rec}(w', s) = w' - \mathsf{Invert}(\mathsf{Syn}(w') - s) = w' - \mathsf{Invert}(\mathsf{Syn}(w' - w)) = w.$$

Then $(\mathsf{SS}, \mathsf{Rec})$ is a $(\mathbb{F}_q^n, \mathcal{W}, t, 0)$-secure sketch.

Theorem 6.1. *Assume the following:*

1. *$(\mathsf{SS}, \mathsf{Rec})$ be a syndrome secure sketch for distance $2t$, that is, $d = 4t + 1$,*

$\mathsf{Gen}(w)$:

1. Compute $ss \leftarrow \mathsf{SS}(w)$.
2. Sample random $\mathsf{key} \in \{0,1\}^\ell$.
3. Obfuscate $\mathsf{unlock}_{w,\mathsf{key}} \leftarrow \mathsf{lock}(w, \mathsf{key})$.
4. Output $\mathsf{key}, \mathsf{pub} = (ss, \mathsf{unlock}_{w,\mathsf{key}})$.

$\mathsf{Rep}(w', ss', \mathsf{unlock}')$:

1. If $V_{\mathsf{key}}(\mathsf{unlock}') = 0$ output $\perp$.
2. Let $w^* \leftarrow \mathsf{Rec}(w', ss')$
3. If $\mathsf{dis}(w', w^*) > t$ or $w^* \notin \mathbb{F}_q^n$ output $\perp$.
4. Output $\mathsf{unlock}'(w^*)$.

Figure 3: Robust Fuzzy Extractor from nonmalleable digital locker and syndrome secure sketch.

2. *$\mathcal{W}$ is the set of all efficiently sampleable distributions W where*
$$\tilde{\mathrm{H}}_\infty(W|\mathsf{SS}(W)) \geq \omega(\log \lambda),$$
3. *$(\mathsf{lock}, \mathsf{unlock}, V_{\mathsf{key}})$ is a nonmalleable digital locker for $(\mathcal{F}, \mathcal{X})$ where $\mathcal{F}$ includes all functions $f : \mathbb{F}_q^n \to \mathbb{F}_q^n$ of the form $f(x) = x + a$ and where $\mathcal{X}$ is the set of all distributions X where $\mathrm{H}_\infty(X) = \omega(\log \lambda)$.*

Then $(\mathsf{Gen}, \mathsf{Rep})$ described in Figure 3 is a $(\mathcal{M}, \mathcal{W}, \ell, t)$-robust fuzzy extractor (Definition 6.2).

Proof of Theorem 6.1. **Correctness:** Correctness is immediate from the correctness guarantees of the secure sketch and digital locker respectively.

Before proceeding to security and nonmalleability, it is more convenient to work with a worst-case definition of min-entropy. Fix some distribution $W \in \mathcal{W}_\lambda$ and let $\alpha = \omega(\log \lambda)$ be some value such that $\tilde{\mathrm{H}}_\infty(W|\mathsf{SS}(W)) \geq \alpha$. Such an α exists by Definition 6.1.
By [35, Lemma 2.2a] over the choice of ss,
$$\mathrm{H}_\infty(W|\mathsf{SS}(W) = ss) \geq \tilde{\mathrm{H}}_\infty(W|\mathsf{SS}(W)) - \log 1/\delta$$
with probability at least $1 - \delta$. Fix $\delta = 2^{-\alpha/2}$ and note that $1 - \delta = \mathtt{ngl}(\lambda)$. That is, there exists some set E such that $\Pr[\mathsf{SS}(W) \in E] \geq 1 - \mathtt{ngl}(\lambda)$ and for all $ss \in E$, $\mathrm{H}_\infty(W|\mathsf{SS}(W) = ss) \geq \alpha/2$. For any adversary, $\mathcal{A}$ that distinguishes two tuples across $\mathsf{SS}(W)$ with a noticeable probability it must be the case that $\mathcal{A}$ distinguishes two tuples across E with noticeable probability.

Security: We proceed by contrapositive. Suppose that the resulting fuzzy extractor is not secure. That is, there exists some PPT $\mathcal{A}$ and polynomial function $p(\lambda)$ such that
$$\Big| \Pr_{\mathsf{key},\mathsf{SS},\mathsf{unlock},\mathcal{A}}[\mathcal{A}(\mathsf{key}, \mathsf{SS}, \mathsf{unlock}) = 1|\mathsf{SS} \in E] - \Pr_{U_\ell,\mathsf{SS},\mathsf{unlock},\mathcal{A},}[\mathcal{A}(U_\ell, \mathsf{SS}, \mathsf{unlock}) = 1|\mathsf{SS} \in E]\Big| > 1/p(\lambda).$$

To argue security, note that lock is virtual black box secure. Define $r(\lambda) = 3 * p(\lambda)$ and let $\mathcal{S}$ be the simulator of $\mathcal{A}$ for polynomial $r(\lambda)$. Then it is the case that for all $\mathsf{SS} \in E$
$$\Big|\Pr[\mathcal{A}(\mathsf{lock}(w, \mathsf{key}), \mathsf{key}, \mathsf{SS}) = 1] - \Pr[\mathcal{S}^{I_{w,\mathsf{key}}}(1^\lambda, \mathsf{key}, \mathsf{SS}) = 1]\Big| \leq \frac{1}{3p(\lambda)}, \tag{10}$$
Note that the above is true if key is replaced by U_ℓ, a uniform random variable over $\{0,1\}^\ell$. It is true that for any polynomial number of queries and for all key, U_ℓ
$$\Big|\Pr[\mathcal{S}^{I_{w,\mathsf{key}}}(\mathsf{key}, \mathsf{SS}, 1^\lambda) = 1] - \Pr[\mathcal{S}^{I_{w,\mathsf{key}}}(U_\ell, \mathsf{SS}, 1^\lambda) = 1]\Big| \leq \frac{1}{3p(\lambda)}. \tag{11}$$

This is proved in [22, Lemma 2].
Using the triangle inequality on Equation 10, Equation 11, and Equation 10 with key replaced by U_ℓ yields that

$$\left| \Pr_{\mathsf{key},\mathsf{SS},\mathsf{unlock},\mathcal{A}}[\mathcal{A}(\mathsf{key},\mathsf{SS}) = 1 | \mathsf{SS} \in E] - \Pr_{U_\ell,\mathsf{SS},\mathsf{unlock},\mathcal{A},}[\mathcal{A}(U_\ell,\mathsf{SS}) = 1 | \mathsf{SS} \in E] \right| \leq 1/p(\lambda).$$

this is a contradiction and completes the security argument.

Robustness: We proceed by contradiction. Let $\mathcal{A}$ be some adversary that breaks the robustness property of the fuzzy extractor. We assume that $\mathcal{A}$ outputs a fixed pair of distributions W, W' (there is a nonuniform algorithm that is programmed with the best pair of distributions) where W is efficiently samplable (see Definition 2.2). As in the security argument, we restrict our consideration where $\mathsf{SS}(W) \in E$ which occurs with overwhelming probability. In this setting, the distribution $W|\mathsf{SS}(W)$ is in the family of distributions $\mathcal{X}$ for which the digital locker should be nonmalleable.

We proceed to show that there exists some PPT adversary $\mathcal{A}'$ that breaks the nonmalleability of the digital locker:

1. Initialize $\mathcal{A}$.
2. Receive W, W' from $\mathcal{A}$.
3. Sample a random $\mathsf{key} \leftarrow \{0,1\}^\ell$.
4. Sample $w^* \leftarrow W$ and compute $ss \leftarrow \mathsf{SS}(w^*)$.
5. Provide $W|ss$ to the challenger and receive unlock in response.
6. Run $(ss', \mathsf{unlock}') \leftarrow \mathcal{A}(ss, \mathsf{unlock}, \mathsf{key})$.
7. If $\mathsf{unlock}' = \mathsf{unlock}$ output $\perp$.
8. If $ss' = ss$ set $f(x) = x$. That is, $f = \mathsf{id}$.
9. Else if $\mathsf{Invert}(ss' - ss) = \perp$ output $\perp$.
10. Else set $f(x) = x + \mathsf{Invert}(ss' - ss)$.
11. Output (unlock', f).

We now turn to analyzing $\mathcal{A}'$s performance. We first remark that if $\mathcal{A}$ breaks robustness of the fuzzy extractor then it must be the case that

$$\Pr_{(w,w') \leftarrow (W,W')} \left[\mathsf{Rep}(w', \mathsf{SS}', \mathsf{unlock}') \neq \{\perp, \mathsf{key}\} \middle| \begin{array}{r} (\mathsf{key}, \mathsf{SS}, \mathsf{unlock}) \leftarrow \mathsf{FE.Gen}(w) \\ \mathsf{SS} \in E \\ (\mathsf{SS}', \mathsf{unlock}') \leftarrow \mathcal{A}(\mathsf{SS}, \mathsf{unlock}) \end{array} \right] > 1/p(\lambda).$$

for some polynomial $p(\lambda)$. Then it is true that

$$\Pr_{(w,w') \leftarrow (W|ss,W')} \left[\mathsf{Rep}(w', \mathsf{SS}', \mathsf{unlock}') \neq \{\perp, \mathsf{key}\} \middle| \begin{array}{r} (\mathsf{key}, ss, \mathsf{unlock}) \leftarrow \mathsf{FE.Gen}(w) \\ ss \in E \\ (\mathsf{SS}', \mathsf{unlock}') \leftarrow \mathcal{A}(ss, \mathsf{unlock}) \end{array} \right] > 1/p(\lambda).$$

By definition the probability that there exists a point beside w that causes unlock to output a value other than $\perp$ is negligible in the randomness of the lock algorithm. This means that if $\mathsf{unlock}' = \mathsf{unlock}$ then $\mathsf{Rep}(w', \mathsf{SS}', \mathsf{unlock}) \in \{\perp, \mathsf{key}\}$ with overwhelming probability. This means that

$$\Pr_{(w,w')\leftarrow(W|ss,W')}\left[\mathsf{Rep}(w', \mathsf{SS}', \mathsf{unlock}') \neq \{\perp, \mathsf{key}\} \;\middle|\; \begin{array}{r}(\mathsf{key}, ss, \mathsf{unlock}) \leftarrow \mathsf{FE.Gen}(w) \\ ss \in E \\ (\mathsf{SS}', \mathsf{unlock}') \leftarrow \mathcal{A}(ss, \mathsf{unlock}) \\ \mathsf{unlock}' \neq \mathsf{unlock}\end{array}\right] > 1/p(\lambda).$$

for some polynomial $p'(\lambda)$. We start by observing that in the setting when $ss = ss'$. Since it is true with probability 1 that $\mathsf{dis}(w, w') \leq t$, this means that $\mathsf{Rec}(w', ss) = w$ with probability 1. This means that the input provided to unlock' will be w and corresponding tampering function is the identity.

We now consider the setting when $ss \neq ss'$. Recall that $\mathsf{dis}(w, w') \leq t$ with probability 1. Note that $w = \mathsf{Rec}(w', ss)$. Let $f(w) \stackrel{def}{=} \mathsf{Rec}(w', ss')$ respectively. Note that by definition $\mathsf{Invert}(\mathsf{Syn}(w') - ss) \neq \perp$ with probability 1. It must also be the case that $\mathsf{Invert}(\mathsf{Syn}(w') - ss') \neq \perp$ with noticeability probability and thus there exists some point $f(w)$ such that $\mathsf{dis}(w', f(w)) \leq t$ and $\mathsf{Syn}(f(w)) = ss'$. In more detail this means that,

$$\begin{aligned} w &= w' - \mathsf{Invert}(\mathsf{Syn}(w') - s) \\ f(w) &= w' - \mathsf{Invert}(\mathsf{Syn}(w') - s') \end{aligned}$$

Substituting one has that

$$\begin{aligned} f(w) &= w' - \mathsf{Invert}(\mathsf{Syn}(w') - ss') \\ &= w + \mathsf{Invert}(\mathsf{Syn}(w') - ss) - \mathsf{Invert}(\mathsf{Syn}(w') - ss') \\ &= w + \mathsf{Invert}(ss' - ss). \end{aligned}$$

Note for the last step is implied by Property 3 of the Syndrome (Definition 6.3) since $\mathsf{dis}(w', w) \leq t$ and $\mathsf{dis}(w', f(w)) \leq t$ and thus $\mathsf{dis}(w, f(w)) \leq 2t$ together these facts imply that

$$\mathsf{Invert}(\mathsf{Syn}(w') - ss) - \mathsf{Invert}(\mathsf{Syn}(w') - ss') = \mathsf{Invert}(ss' - ss).$$

Thus, in both cases (when $ss = ss'$ and when $ss \neq ss'$) $\mathcal{A}'$ is able to extract the tampering function f and unlock' constitutes a break of the nonmalleability property of the digital locker. This completes the robustness argument.

This completes the proof of Theorem 6.1. □

Aligning tampering functions There is a subtlety when we instantiate the fuzzy extractor of Theorem 6.1 – the digital locker in Theorem 6.1 requires a function class of the domain $\mathbb{F}_q^n$, whereas the digital locker constructed in Figure 2 works in $\mathbb{Z}_{p(\lambda)}$. It is unclear whether there is an additively homomorphic mapping between these spaces for arbitrary p, q, n. Therefore, a trivial plug-in of the digital locker of Figure 2 does not work. In Section 6.1, we show that how to align readings in a simple way at the cost of increased leakage of the secure sketch.

$\mathsf{SS}'(w)$:

1. Compute $ss \leftarrow \mathsf{SS}(w)$.
2. Obfuscate $\mathsf{unlockPoint}_w \leftarrow \mathsf{lockPoint}(w)$.
3. Output (ss, unlock_w).

$\mathsf{Rec}'(w', ss', \mathsf{unlockPoint}')$:

1. If $\mathsf{V}_{\mathtt{obf}}(\mathsf{unlockPoint}') = 0$ output $\perp$.
2. Compute $w^* \leftarrow \mathsf{Rec}(w', ss')$
3. If $\mathsf{dis}(w', w^*) > t$ or $w^* \notin \mathbb{F}_q^n$ output $\perp$.
4. Else if $\mathsf{unlockPoint}'(w^*) = 0$ output $\perp$.
5. Else output w^*.

Figure 4: Robust Secure Sketch from nonmalleable point obfuscation and syndrome secure sketch.

An alternative to efficiently sampleable W. Theorem 6.1 required W to be efficiently sampleable. This is because in the proof the reduction samples a $w \leftarrow W$ to compute a secure sketch ss and create the conditional distribution $W|\mathsf{SS}(w)$. An alternative approach is to define all of the objects throughout our main technical sections to be nonmalleable in the presence of auxiliary information Z such that $\mathrm{H}_\infty(W|Z) \geq \omega(\log \lambda)$. In this case, $\mathcal{A}'$ can receive ss as auxiliary information and directly forward it to $\mathcal{A}$.

All of the proofs contained in this work naturally extend to the setting of auxiliary information. The major work needed to have confidence in the auxiliary input approach is to show that [7, Assumption 3] holds in the non-uniform generic group model [27] in the presence of auxiliary information. Importantly, the distribution W has average min-entropy conditioned on $\mathsf{SS}(W)$. There are strong impossibility results on digital lockers that are secure against hard to invert auxiliary information [19].

Applications of nonmalleable point function obfuscation. Nonmalleable point obfuscation and nonmalleable point obfuscation with associated data (Definition 3.2) can be used to build robust secure sketches and robust fuzzy extractors, respectively.

- **Robust secure sketch:** Robustness for secure sketches is defined in a similar fashion as for fuzzy extractors. For correlated distributions W, W', the adversary receives $\mathsf{SS}(w)$ from the challenger and outputs SS'. The adversary wins the robustness game if he succeeds in finding a value SS' such that $\mathsf{Rec}(\mathsf{SS}', w') \notin \{\perp, w\}$. Informally, suppose $(\mathsf{lockPoint}, \mathsf{unlockPoint})$ is a nonmalleable point obfuscation and $(\mathsf{SS}, \mathsf{Rec})$ is a syndrome-based secure sketch. Then Figure 4 describes a robust secure sketch. The formal theorem and proof can be found in Section 6.2.
- **Robust fuzzy extractor:** Let $(\mathsf{lockPoint}, \mathsf{unlockPoint})$ be a nonmalleable point obfuscation with associated data, $(\mathsf{SS}, \mathsf{Rec})$ be a syndrome-based secure sketch and $\mathtt{ext}$ be a randomness extractor. Then Figure 5 describes a robust fuzzy extractor. We stress that this construction requires the remaining entropy of W to be high conditioned on both the produced key which is produced using a randomness extractor [49, 53] and $\mathsf{SS}(w)$. There is no limitation on the key length in the robust fuzzy extractor from the nonmalleable digital locker (in Theorem 6.1). The formal theorem and proof can be found in Section 6.3.

6.1 Instantiations – Aligning the Tampering Function Classes

In this section, we show how to align the tampering function classes required by the fuzzy extractor of Theorem 6.1 and the construction of Figure 2. This deals with the mismatch in input domain for the

$\mathsf{Gen}(w)$:

1. Sample random $\mathsf{seed} \in \{0,1\}^\rho$.
2. Generate $\mathsf{key} \leftarrow \mathtt{ext}(w; \mathsf{seed})$.
3. Compute $ss \leftarrow \mathsf{SS}(w)$.
4. Obfuscate $\mathsf{unlockPoint}_{w,\mathsf{seed}} \leftarrow \mathsf{lockPoint}(w, \mathsf{seed})$.
5. Output key and $\mathsf{pub} = (ss, \mathsf{unlockPoint}_{w,\mathsf{seed}}, \mathsf{seed})$.

$\mathsf{Rep}(w', ss', \mathsf{unlockPoint}', \mathsf{seed}')$:

1. If $V_{\mathsf{key}}(\mathsf{unlockPoint}') = 0$ output $\perp$.
2. Compute $w^* \leftarrow \mathsf{Rec}(w', ss')$
3. If $\mathsf{dis}(w', w^*) > t$ or $w^* \notin \mathbb{F}_q^n$ output $\perp$.
4. if $\mathsf{unlockPoint}'(w^*, \mathsf{seed}') = 0$ output $\perp$.
5. Output $\mathsf{key} \leftarrow \mathtt{ext}(w^*; \mathsf{seed}')$.

Figure 5: Robust Fuzzy Extractor from nonmalleable point obfuscation with associated data, syndrome secure sketch and randomness extractor.

syndrome (which takes inputs in $\mathbb{F}_q^n$) and the nonmalleable digital locker (which takes inputs in $\mathbb{Z}_p$).

Assume that the input readings w, w' are q-ary strings of length n. Instead of using a q-ary error correcting code ($\mathbf{A} \in \mathbb{F}_q^{n \times k}$ and $\mathsf{Syn} : \mathbb{F}_q^{n \times (n-k)}$), we consider an error correcting code with entries in $\mathbb{F}_{q'}$ for some prime $q' \geq 2(q-1)+1$. That is, let $\mathbf{A}' \in \mathbb{F}_q^{n \times k}$ be a $(n, k, d = 4t+1)$ linear error correcting code, and let Syn' be the corresponding syndrome. Furthermore, we make the restriction $p \geq q^n$, where $\mathbb{Z}_p$ is the input domain of the digital locker of Figure 2. In the construction of Rep, note there is a check if the recovered value, w^*, is not q-ary, in which case we output $\perp$. Thus, for the adversary to successfully break robustness they must produce a q-ary output.

Now we encode every string $x \in \mathbb{F}_q^n$ as the natural q-ary representation, i.e., $x \mapsto \sum_{i \in [n]} x_i q^{i-1} \in \mathbb{Z}_p$, denoted as $\mathsf{Enc}(x)$. Moreover, the digital locker takes input an encoded version of w, i.e.,

$$\mathsf{lockPoint}(\mathsf{Enc}(w), \mathsf{seed}) \text{ and } \mathsf{unlockPoint}(\mathsf{Enc}(w^*), \mathsf{seed}').$$

By setting things up in this way, Theorem 6.1 holds even if the underlying digital locker is non-malleable for shift functions in $\mathbb{Z}_p$.

In Theorem 6.1 the reduction extracts a tampering function $f : \mathbb{F}_q^n \to \mathbb{F}_q^n$ where $f(w) = w + \mathsf{Invert}(ss' - ss)$. With the modified syndrome construction, the function $f : \mathbb{F}_q^n \to \mathbb{F}_q^n$, as above, the reduction can extract a function $f(w) = w + \mathsf{Invert}(ss' - ss)$. By the check of $w^* \in \mathbb{F}_q^n$ and the initial condition that $w \in \mathbb{F}_q^n$, this implies that $w_i + \mathsf{Invert}(ss' - ss)_i \in \mathbb{F}_q$, we can first conclude that $\mathsf{Invert}(ss - ss')$ can be represented in $\{-(q-1), ..., (q-1)\}^n$. Under this representation, we conclude that for each i, $w_i + \mathsf{Invert}(ss' - ss)_i \in \mathbb{F}_q$ using standard integer addition. So, for each i, we are guaranteed an element in $\mathbb{F}_q$ (i.e., $\mathsf{Enc}(w^*) = \mathsf{Enc}(w) + \mathsf{Enc}(\mathsf{Invert}(ss' - ss))$), which corresponds exactly to a shift tampering function in $\mathbb{Z}_p$, and thus the reduction can break the underlying non-malleable digital locker.

The effect of this transform is to increase the required entropy on the distribution W. The standard analysis of the secure sketch assumes that $\mathsf{SS}(W)$ leaks $(n-k)\log q$ bits of information about W. By increasing the syndrome from q to q' this increases the leakage of the secure sketch by $(n-k)\log(q'/q) \approx (n-k)\log 2$. This transform applies to the constructions in Figures 4 and 5 as well. We do not include it in our proofs to show the general connection between syndrome secure sketches and nonmalleable point obfuscation variants.

6.2 Application of nonmalleable point function obfuscation

Definition 6.5 (Robust Secure Sketch)**.** *Let* $(\mathsf{SS}, \mathsf{Rec})$ *be a* $(\mathcal{M}, \mathcal{W}, t, \delta)$*-secure sketch as described in Definition 6.1. Let* $W, W' \in \mathcal{M}$ *be (correlated) distributions such that*

$$\Pr_{(w,w') \leftarrow (W,W')}[\mathsf{dis}(w, w') \leq t] = 1$$

and $W, W' \in \mathcal{W}$*. Then* $(\mathsf{SS}', \mathsf{Rec}')$ *is a robust secure sketch if for all* $W, W' \in \mathcal{W}$ *and for all adversaries* $\mathcal{A}$*, the advantage of* $\mathcal{A}$ *in the following experiment is at most* $\mathtt{ngl}(\lambda)$*:*

1. *Sample* $(w, w') \leftarrow (W, W')$*.*
2. *Compute* $ss \leftarrow \mathsf{SS}(w)$ *and send it to* $\mathcal{A}$*.*
3. $\mathcal{A}$ *outputs* ss' *and wins if* $ss' \neq ss$ *and* $\mathsf{Rec}(ss', w') \notin \{\perp, w\}$*.*

Theorem 6.2. *Let* $\mathcal{W}$ *be the set of all efficiently sampleable distributions* W *where* $\tilde{H}_\infty(W|\mathsf{SS}(W)) \geq \omega(\log \lambda)$ *and* $(\mathsf{SS}, \mathsf{Rec})$ *be an* $(\mathcal{M}, \mathcal{W}, 2t, \delta)$ *syndrome-based secure sketch. Let* $(\mathsf{lockPoint}, \mathsf{unlockPoint})$ *be an* $(\mathcal{F}, \mathcal{X}, \rho)$*-nonmalleable point function obfuscation where* $\mathcal{F}$ *includes all functions of the form* $f(x) = x + a$ *and* $\mathcal{X}$ *is the set of all distributions* X *such that* $H_\infty(X) = \omega(\log \lambda)$*. Then* $(\mathsf{SS}, \mathsf{Rec})$ *described in Figure 4 is an* $(\mathcal{M}, \mathcal{W}, t, \delta)$*-robust secure sketch.*

Proof of Theorem 6.2.

Correctness & security Correctness and security are immediate from the correctness guarantees of the secure sketch and point function obfuscation, and the security of the secure sketch respectively.

Robustness Again, it is more convenient to work with a worst-case definition of min-entropy. For any adversary $\mathcal{A}$ that distinguishes two tuples across $\mathsf{SS}(W)$ with a noticeable probability it must be the case that $\mathcal{A}$ distinguishes two tuples across E with noticeable probability, where E is the set such that $\Pr[\mathsf{SS}(W) \in E] \geq 1 - \mathtt{ngl}(\lambda)$ and for all $ss \in E$, $\mathrm{H}_\infty(W|\mathsf{SS}(W) = ss) \geq \alpha/2$.

We then proceed by contradiction. Let $\mathcal{A}$ be some adversary that breaks the robustness property of the secure sketch $(\mathsf{SS}', \mathsf{Rec}')$. We assume that $\mathcal{A}$ outputs a fixed pair of distributions W, W' (there is a nonuniform algorithm that is programmed with the best pair of distributions) where W is efficiently sampleable (see Definition 2.2). As in the security argument, we restrict our consideration where $\mathsf{SS}(W) \in E$ which occurs with overwhelming probability. In this setting, the distribution $W|\mathsf{SS}(W)$ is in the family of distributions $\mathcal{X}$ for which the point function obfuscation should be nonmalleable.

We proceed to show that there exists some PPT adversary $\mathcal{A}'$ that breaks the nonmalleability of the point function obfuscation:

1. Initialize $\mathcal{A}$.
2. Receive W, W' from $\mathcal{A}$.
3. Sample $w^* \leftarrow W$ and compute $ss \leftarrow \mathsf{SS}(w^*)$.
4. Provide $W|ss$ to the challenger and receive $\mathsf{unlockPoint}$ in response.

5. Run $(ss', \mathsf{unlockPoint}') \leftarrow \mathcal{A}(ss, \mathsf{unlockPoint})$.
6. If $\mathsf{unlockPoint}' = \mathsf{unlockPoint}$ output $\perp$.
7. If $ss' = ss$ set $f(x) = x$. That is, $f = \mathsf{id}$.
8. Else if $\mathsf{Invert}(ss' - ss) = \perp$ output $\perp$.
9. Else set $f(x) = x + \mathsf{Invert}(ss' - ss)$.
10. Output $(\mathsf{unlockPoint}', f)$.

The argument for robustness follows identically to the proof of Theorem 6.1. It is below for completeness. We now turn to analyzing $\mathcal{A}'$s performance. We first remark that if $\mathcal{A}$ breaks robustness of the secure sketch then it must be the case that

$$\Pr_{(w,w') \leftarrow (W,W')} \left[\mathsf{Rec}'(w', \mathsf{SS}', \mathsf{unlockPoint}') \neq \perp \middle| \begin{array}{r} (\mathsf{SS}, \mathsf{unlockPoint}) \leftarrow \mathsf{SS}'(w) \\ \mathsf{SS} \in E \\ (\mathsf{SS}', \mathsf{unlockPoint}') \leftarrow \mathcal{A}(\mathsf{SS}, \mathsf{unlockPoint}) \end{array} \right] > 1/p(\lambda).$$

for some polynomial $p(\lambda)$. Then it is true that

$$\Pr_{(w,w') \leftarrow (W|ss,W')} \left[\mathsf{Rec}'(w', \mathsf{SS}', \mathsf{unlockPoint}') \neq \perp \middle| \begin{array}{r} (ss, \mathsf{unlockPoint}) \leftarrow \mathsf{SS}'(w) \\ ss \in E \\ (\mathsf{SS}', \mathsf{unlockPoint}') \leftarrow \mathcal{A}(ss, \mathsf{unlockPoint}) \end{array} \right] > 1/p(\lambda).$$

By definition the probability that there exists a point beside w that causes $\mathsf{unlockPoint}$ to output a value other than $\perp$ is negligible in the randomness of the $\mathsf{lockPoint}$ algorithm. This means that if $\mathsf{unlockPoint}' = \mathsf{unlockPoint}$ then $\mathsf{Rec}'(w', \mathsf{SS}', \mathsf{unlockPoint}) \in \{\perp, w\}$ with overwhelming probability. This means that

$$\Pr_{(w,w') \leftarrow (W|ss,W')} \left[\mathsf{Rec}'(w', \mathsf{SS}', \mathsf{unlockPoint}') \neq \perp \middle| \begin{array}{r} (ss, \mathsf{unlockPoint}) \leftarrow \mathsf{SS}'(w) \\ ss \in E \\ (\mathsf{SS}', \mathsf{unlockPoint}') \leftarrow \mathcal{A}(ss, \mathsf{unlockPoint}) \\ \mathsf{unlockPoint}' \neq \mathsf{unlockPoint} \end{array} \right] > 1/p(\lambda).$$

for some polynomial $p'(\lambda)$. We start by observing that in the setting when $ss = ss'$, since it is true with probability 1 that $\mathsf{dis}(w, w') \leq t$, this means that $\mathsf{Rec}(w', ss) = w$ with probability 1. This means that the input provided to $\mathsf{unlockPoint}'$ will be w and corresponding tampering function is the identity.

We now consider the setting when $ss \neq ss'$. Recall that $\mathsf{dis}(w, w') \leq t$ with probability 1. Note that $w = \mathsf{Rec}(w', ss)$. Let $f(w) \stackrel{def}{=} \mathsf{Rec}(w', ss')$ respectively. Note that by definition $\mathsf{Invert}(\mathsf{Syn}(w') - ss) \neq \perp$ with probability 1. It must also be the case that $\mathsf{Invert}(\mathsf{Syn}(w') - ss') \neq \perp$ with noticeability probability and thus there exists some point $f(w)$ such that $\mathsf{dis}(w', f(w)) \leq t$ and $\mathsf{Syn}(f(w)) = ss'$. As before,

$$f(w) = w + \mathsf{Invert}(ss' - ss).$$

Thus, in both cases (when $ss = ss'$ and when $ss \neq ss'$) $\mathcal{A}'$ is able to extract the tampering function f and $\mathsf{unlockPoint}'$ constitutes a break of the nonmalleability property of the point function obfuscation. This completes the robustness argument. This completes the proof of Theorem 6.2. $\square$

6.3 Application of Nonmalleable Point Obfuscation with Associated Data

Definition 6.6 (Strong average case randomness extractor [49, 53])**.** *Let* $\texttt{ext} : \{0,1\}^n \to \{0,1\}^\ell$ *be a a PPT algorithm that uses* ρ *bits of randomness.* $\texttt{ext}$ *is an efficient* (n, m, ℓ, ϵ)*-strong randomness extractor if for all distributions* W *on* $\{0,1\}^n$ *and* Y*, such that* $\tilde{H}_\infty(W|Y) \leq m$*,*

$$\mathsf{SD}\big((\texttt{ext}(W;\mathsf{seed}), \mathsf{seed}, Y), (U_\ell, \mathsf{seed}, Y)\big) \leq \epsilon$$

where $\mathsf{seed} \in \{0,1\}^\rho$ *is uniform and* U_ℓ *denotes the random uniform variable on* $\{0,1\}^\ell$*.*

Theorem 6.3. *Let* $\mathcal{W}$ *be the set of all efficiently sampleable distributions* W *where* $\tilde{H}_\infty(W|\mathsf{SS}(W)) \geq \omega(\log\lambda)$ *and* $(\mathsf{SS}, \mathsf{Rec})$ *be an* $(\mathcal{M}, \mathcal{W}, 2t, \delta)$ *syndrome-based secure sketch. Let* $(\mathsf{lockPoint}, \mathsf{unlockPoint})$ *be an* $(\mathcal{F}, \mathcal{X}, \rho)$*-nonmalleable point function obfuscation where* $\mathcal{F}$ *includes all functions of the form* $f(x) = x + a$ *and* $\mathcal{X}$ *is the set of all distributions* X *such that* $H_\infty(X) = \omega(\log\lambda)$*. Let* $\texttt{ext}$ *be an* (n, m, ℓ, ϵ)*-strong average-case randomness extractor. Then* $(\mathsf{Gen}, \mathsf{Rep})$ *described in Figure 5 is an* $(\mathcal{M}, \mathcal{W}, \ell, t)$*-robust fuzzy extractor (Definition 6.2).*

Proof of Theorem 6.3. We argue correctness, security, and robustness separately.

Correctness: Correctness is immediate from the correctness guarantees of the secure sketch, point function obfuscation and randomness extractor.

Again, it is more convenient to work with a worst-case definition of min-entropy. For any adversary $\mathcal{A}$ that distinguishes two tuples across $\mathsf{SS}(W)$ with a noticeable probability it must be the case that $\mathcal{A}$ distinguishes two tuples across E with noticeable probability, where E is the set such that $\Pr[\mathsf{SS}(W) \in E] \geq 1 - \texttt{ngl}(\lambda)$ and for all $ss \in E$, $\mathrm{H}_\infty(W|\mathsf{SS}(W) = ss) \geq \alpha/2$.

Security: We proceed by contrapositive. Suppose that the resulting fuzzy extractor is not secure. That is, there exists some PPT $\mathcal{A}$ and polynomial function $p(\lambda)$ such that

$$\left| \Pr_{\mathsf{key},\mathsf{pub},\mathcal{A}}[\mathcal{A}(\mathsf{key}, \mathsf{pub}) = 1 | \mathsf{SS} \in E] - \Pr_{U_\ell,\mathsf{pub},\mathcal{A},}[\mathcal{A}(U_\ell, \mathsf{pub}) = 1 | \mathsf{SS} \in E] \right| > 1/p(\lambda)$$

where $\mathsf{pub} = (\mathsf{SS}, \mathsf{unlockPoint}, \mathsf{seed})$. To argue security, note that $\mathsf{lockPoint}$ is virtual black box secure. Define $r(\lambda) = 3 * p(\lambda)$ and let $\mathcal{S}$ be the simulator of $\mathcal{A}$ for polynomial $r(\lambda)$. Then it is the case that for all $\mathsf{SS} \in E$

$$\left| \Pr[\mathcal{A}(\mathsf{unlockPoint}, \mathsf{seed}, \mathsf{SS}) = 1] - \Pr[\mathcal{S}^{I_{w,\mathsf{seed}}}(\mathsf{seed}, 1^\lambda, \mathsf{SS}) = 1] \right| \leq \frac{1}{3p(\lambda)}.$$

Also note that since $\texttt{ext}$ is a strong average-case randomness extractor then

$$\mathsf{SD}\big((\mathsf{key}, \mathsf{seed}, \mathsf{SS}), (U_\ell, \mathsf{seed}, \mathsf{SS})\big) \leq \epsilon$$

and thus

$$\left| \Pr[\mathcal{A}(\mathsf{unlockPoint}, \mathsf{seed}, \mathsf{SS}, \mathsf{key}) = 1] - \Pr[\mathcal{S}^{I_{w,\mathsf{seed}}}(\mathsf{seed}, \mathsf{key}, 1^\lambda, \mathsf{SS}) = 1] \right| \leq \frac{1}{3p(\lambda)}. \tag{12}$$

The above is true if key is replaced by U_ℓ, a uniform random variable over $\{0,1\}^\ell$.

It is true that for any polynomial number of queries and for all key, U_ℓ

$$\left|\Pr[\mathcal{S}^{I_{w,\mathsf{seed}}}(\mathsf{key},\mathsf{pub},1^\lambda)=1]-\Pr[\mathcal{S}^{I_{w,\mathsf{seed}}}(U_\ell,\mathsf{pub},1^\lambda)=1]\right|\leq\frac{1}{3p(\lambda)}. \tag{13}$$

This is proved in [22, Lemma 2].
Using the triangle inequality on Equation 12, Equation 13, and Equation 12 with key replaced by U_ℓ yields that

$$\left|\Pr_{\mathsf{key},\mathsf{pub},\mathcal{A}}[\mathcal{A}(\mathsf{key},\mathsf{pub})=1|\mathsf{SS}\in E]-\Pr_{U_\ell,\mathsf{pub},\mathcal{A},}[\mathcal{A}(U_\ell,\mathsf{pub})=1|\mathsf{SS}\in E]\right|\leq 1/p(\lambda).$$

this is a contradiction and completes the security argument.

Robustness: We proceed by contradiction. Let $\mathcal{A}$ be some adversary that breaks the robustness property of the fuzzy extractor. We assume that $\mathcal{A}$ outputs a fixed pair of distributions W, W' (there is a nonuniform algorithm that is programmed with the best pair of distributions) where W is efficiently samplable (see Definition 2.2). As in the security argument, we restrict our consideration where $\mathsf{SS}(W)\in E$ which occurs with overwhelming probability. In this setting, the distribution $W|(\mathsf{SS}(W),\mathsf{seed},\mathsf{key})$ is in the family of distributions $\mathcal{X}$ for which the point function obfuscation should be nonmalleable.

We proceed to show that there exists some PPT adversary $\mathcal{A}'$ that breaks the nonmalleability of the point function obfuscation:

1. Initialize $\mathcal{A}$.
2. Receive W, W' from $\mathcal{A}$.
3. Sample a random $\mathsf{seed}\leftarrow\{0,1\}^\rho$.
4. Sample $w^*\leftarrow W$ and compute $ss\leftarrow\mathsf{SS}(w^*)$.
5. Compute $\mathsf{key}\leftarrow\mathtt{ext}(w^*;\mathsf{seed})$.
6. Provide $W|(ss,\mathsf{seed},\mathsf{key})$ to the challenger and receive $\mathsf{unlockPoint}$ in response.
7. Run $(ss',\mathsf{seed}',\mathsf{unlockPoint}')\leftarrow\mathcal{A}(ss,\mathsf{seed},\mathsf{unlockPoint},\mathsf{key})$.
8. If $\mathsf{unlockPoint}'=\mathsf{unlockPoint}$ output $\perp$.
9. If $ss'=ss$ set $f(x)=x$. That is, $f=\mathsf{id}$.
10. Else if $\mathsf{Invert}(ss'-ss)=\perp$ output $\perp$.
11. Else set $f(x)=x+\mathsf{Invert}(ss'-ss)$.
12. Output $(\mathsf{unlockPoint}', f, \mathsf{seed}')$.

The argument for robustness follows identically to the proof of Theorem 6.1. It is below for completeness. We now turn to analyzing $\mathcal{A}'$s performance. We first remark that if $\mathcal{A}$ breaks robustness of the fuzzy extractor then it must be the case that

$$\Pr_{(w,w')\leftarrow(W,W')}\left[\mathsf{Rep}(w',\mathsf{pub}')\neq\{\perp,\mathsf{key}\}\middle|\begin{array}{r}(\mathsf{key},\mathsf{pub})\leftarrow\mathsf{FE.Gen}(w)\\ \mathsf{SS}\in E\\ (\mathsf{pub}')\leftarrow\mathcal{A}(\mathsf{SS},\mathsf{pub})\end{array}\right]>1/p(\lambda).$$

for some polynomial $p(\lambda)$ and $\mathsf{pub}' = (\mathsf{SS}', \mathsf{seed}', \mathsf{unlockPoint}')$, $\mathsf{pub} = (\mathsf{SS}, \mathsf{seed}, \mathsf{unlockPoint})$. Then it is true that

$$\Pr_{(w,w')\leftarrow(W|(ss,\mathsf{seed},\mathsf{key}),W')}\left[\mathsf{Rep}(w',\mathsf{pub}') \neq \{\perp,\mathsf{key}\} \middle| \begin{array}{r}(\mathsf{key},\mathsf{pub}) \leftarrow \mathsf{FE.Gen}(w)\\ ss \in E\\ (\mathsf{pub}') \leftarrow \mathcal{A}(\mathsf{pub})\end{array}\right] > 1/p(\lambda)$$

with $\mathsf{pub} = (ss, \mathsf{seed}, \mathsf{unlockPoint})$ this time.
By definition the probability that there exists a point beside w that causes $\mathsf{unlockPoint}$ to output a value other than 0 is negligible in the randomness of the $\mathsf{lockPoint}$ algorithm. This means that if $\mathsf{unlockPoint}' = \mathsf{unlockPoint}$ and $\mathsf{seed}' = \mathsf{seed}$ then $\mathsf{Rep}(w', \mathsf{SS}', \mathsf{seed}', \mathsf{unlockPoint}) \in \{\perp, \mathsf{key}\}$ with overwhelming probability. This means that

$$\Pr_{(w,w')\leftarrow(W|(ss,\mathsf{seed},\mathsf{key}),W')}\left[\mathsf{Rep}(w',\mathsf{pub}') \neq \{\perp,\mathsf{key}\} \middle| \begin{array}{r}(\mathsf{key},\mathsf{pub}) \leftarrow \mathsf{FE.Gen}(w)\\ ss \in E\\ (\mathsf{pub}') \leftarrow \mathcal{A}(\mathsf{pub})\\ \mathsf{unlockPoint}' \neq \mathsf{unlockPoint}\end{array}\right] > 1/p(\lambda).$$

for some polynomial $p'(\lambda)$.

We start by observing that in the setting when $ss = ss'$, since it is true with probability 1 that $\mathsf{dis}(w, w') \leq t$, this means that $\mathsf{Rec}(w', ss) = w$ with probability 1. This means that the input provided to $\mathsf{unlockPoint}'$ will be w and corresponding tampering function is the identity. Then if $\mathsf{seed}' \neq \mathsf{seed}$, this breaks the point function obfuscation with associated data nonmalleability property.

We now consider the setting when $ss \neq ss'$. Recall that $\mathsf{dis}(w, w') \leq t$ with probability 1. Note that $w = \mathsf{Rec}(w', ss)$. Let $f(w) \stackrel{def}{=} \mathsf{Rec}(w', ss')$ respectively. Note that by definition $\mathsf{Invert}(\mathsf{Syn}(w') - ss) \neq \perp$ with probability 1. It must also be the case that $\mathsf{Invert}(\mathsf{Syn}(w') - ss') \neq \perp$ with noticeability probability and thus there exists some point $f(w)$ such that $\mathsf{dis}(w', f(w)) \leq t$ and $\mathsf{Syn}(f(w)) = ss'$. As before $f(w) = w + \mathsf{Invert}(ss' - ss)$. Thus, in both cases (when $ss = ss'$ and when $ss \neq ss'$) $\mathcal{A}'$ is able to extract the tampering function f and $(\mathsf{unlockPoint}', \mathsf{seed}')$ constitutes a break of the nonmalleability property of the point function obfuscation. This completes the robustness argument which completes the proof of Theorem 6.3. □

Acknowledgements

The authors thank the reviewers for their helpful comments and suggestions. The authors wish to thank Leonid Reyzin for important discussions. The work of D.A. was done while at NIST. C.C. is supported by NSF Awards #1849904 and 2141033 and a fellowship from Synchrony Inc. B.F. is supported by NSF Awards #1849904, and 2141033. The work of F.L. is supported by the NSF Award #1942400.

This research is based upon work supported in part by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via Contract No. 2019-19020700008. This material is based upon work supported by the Defense Advanced Research Projects Agency under Air Force Contract No. FA8702-15-D-0001. Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of DARPA, ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

References

[1] Q. Alamélou, P.-E. Berthier, C. Cachet, S. Cauchie, B. Fuller, P. Gaborit, and S. Simhadri. Pseudoentropic isometries: A new framework for fuzzy extractor reusability. In *AsiaCCS*, 2018.

[2] P. Ananth and A. Jain. Indistinguishability obfuscation from compact functional encryption. In *Advances in Cryptology – CRYPTO*, pages 308–326. Springer, 2015.

[3] L. Bahler, G. Di Crescenzo, Y. Polyakov, K. Rohloff, and D. B. Cousins. Practical implementation of lattice-based program obfuscators for point functions. In *2017 International Conference on High Performance Computing & Simulation (HPCS)*, pages 761–768. IEEE, 2017.

[4] B. Barak, N. Bitansky, R. Canetti, Y. T. Kalai, O. Paneth, and A. Sahai. Obfuscation for evasive functions. In Y. Lindell, editor, *TCC*, volume 8349 of *Lecture Notes in Computer Science*, pages 26–51. Springer, 2014.

[5] B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, S. Vadhan, and K. Yang. On the (im) possibility of obfuscating programs. *Journal of the ACM (JACM)*, 59(2):6, 2012.

[6] J. Bartusek, T. Lepoint, F. Ma, and M. Zhandry. New techniques for obfuscating conjunctions. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 636–666. Springer, 2019.

[7] J. Bartusek, F. Ma, and M. Zhandry. The distinction between fixed and random generators in group-based assumptions. In *Advances in Cryptology – CRYPTO*, 2019.

[8] B. Bauer, G. Fuchsbauer, and J. Loss. A classification of computational assumptions in the algebraic group model. In *Annual International Cryptology Conference*, pages 121–151. Springer, 2020.

[9] M. Bellare and P. Rogaway. Random oracles are practical: a paradigm for designing efficient protocols. In *CCS 1993 — Proceedings of the First ACM Conference on Computer and Communications Security*, 1993.

[10] M. Bellare, I. Stepanovs, and S. Tessaro. Contention in cryptoland: obfuscation, leakage and UCE. In *Theory of Cryptography Conference*, pages 542–564. Springer, 2016.

[11] C. H. Bennett, G. Brassard, C. Crépeau, and M.-H. Skubiszewska. Practical quantum oblivious transfer. In *Annual international cryptology conference*, pages 351–366. Springer, 1991.

[12] A. Bishop, L. Kowalczyk, T. Malkin, V. Pastro, M. Raykova, and K. Shi. A simple obfuscation scheme for pattern-matching with wildcards. In *Annual International Cryptology Conference*, pages 731–752. Springer, 2018.

[13] N. Bitansky and R. Canetti. On strong simulation and composable point obfuscation. In *Advances in Cryptology–CRYPTO 2010*, pages 520–537. Springer, 2010.

[14] X. Boyen. Reusable cryptographic fuzzy extractors. In *Proceedings of the 11th ACM conference on Computer and Communications Security*, pages 82–91, 2004.

[15] X. Boyen. Robust and reusable fuzzy extractors. In *Security with Noisy Data*, pages 101–112. Springer, 2007.

[16] X. Boyen, Y. Dodis, J. Katz, R. Ostrovsky, and A. Smith. Secure remote authentication using biometric data. In *Advances in Cryptology–EUROCRYPT 2005*, pages 147–163. Springer-Verlag, 2005.

[17] Z. Brakerski and G. N. Rothblum. Obfuscating conjunctions. *Journal of Cryptology*, 30(1):289–320, 2017.

[18] Z. Brakerski, V. Vaikuntanathan, H. Wee, and D. Wichs. Obfuscating conjunctions under entropic ring LWE. In *Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science*, pages 147–156, 2016.

[19] C. Brzuska, P. Farshim, and A. Mittelbach. Indistinguishability obfuscation and UCEs: The case of computationally unpredictable sources. In *Annual Cryptology Conference*, pages 188–205. Springer, 2014.

[20] R. Canetti. Towards realizing random oracles: Hash functions that hide all partial information. In *Advances in Cryptology—-CRYPTO'97*, pages 455–469. Springer, 1997.

[21] R. Canetti and R. R. Dakdouk. Obfuscating point functions with multibit output. In *Advances in Cryptology–EUROCRYPT 2008*, pages 489–508. Springer, 2008.

[22] R. Canetti, B. Fuller, O. Paneth, L. Reyzin, and A. Smith. Reusable fuzzy extractors for low-entropy distributions. *Journal of Cryptology*, 34(1):1–33, 2021.

[23] R. Canetti, O. Goldreich, and S. Halevi. The random oracle methodology, revisited. *J. ACM*, 51(4):557–594, jul 2004.

[24] R. Canetti, G. N. Rothblum, and M. Varia. Obfuscation of hyperplane membership. In *Theory of Cryptography Conference*, pages 72–89. Springer, 2010.

[25] R. Canetti and M. Varia. Non-malleable obfuscation. In O. Reingold, editor, *Theory of Cryptography*, pages 73–90, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[26] C. Cooper. On the rank of random matrices. *Random Structures & Algorithms*, 16(2):209–232, 2000.

[27] S. Coretti, Y. Dodis, and S. Guo. Non-uniform bounds in the random-permutation, ideal-cipher, and generic-group models. In *Annual International Cryptology Conference*, pages 693–721. Springer, 2018.

[28] D. B. Cousins, G. Di Crescenzo, K. D. Gür, K. King, Y. Polyakov, K. Rohloff, G. W. Ryan, and E. Savas. Implementing conjunction obfuscation under entropic ring LWE. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 354–371. IEEE, 2018.

[29] G. Couteau, S. Katsumata, and B. Ursu. Non-interactive zero-knowledge in pairing-free groups from weaker assumptions. *Eurocrypt*, 2020.

[30] R. Cramer, Y. Dodis, S. Fehr, C. Padró, and D. Wichs. Detection of algebraic manipulation with applications to robust secret sharing and fuzzy extractors. In *Advances in Cryptology–EUROCRYPT 2008*, pages 471–488. Springer, 2008.

[31] C. Crépeau. Efficient cryptographic protocols based on noisy channels. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 306–317. Springer, 1997.

[32] L. Demarest, B. Fuller, and A. Russell. Code offset in the exponent. In *2nd Conference on Information-Theoretic Cryptography (ITC 2021)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2021.

[33] Y. Dodis, K. Haralambiev, A. López-Alt, and D. Wichs. Efficient public-key cryptography in the presence of key leakage. In *Advance in Cryptology – ASIACRYPT*, pages 613–631. Springer, 2010.

[34] Y. Dodis, B. Kanukurthi, J. Katz, L. Reyzin, and A. Smith. Robust fuzzy extractors and authenticated key agreement from close secrets. *IEEE Transactions on Information Theory*, 58(9):6207–6222, 2012.

[35] Y. Dodis, R. Ostrovsky, L. Reyzin, and A. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM Journal on Computing*, 38(1):97–139, 2008.

[36] Y. Dodis and D. Wichs. Non-malleable extractors and symmetric key cryptography from weak secrets. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 601–610. ACM, 2009.

[37] A. Escala, G. Herold, E. Kiltz, C. Ràfols, and J. Villar. An algebraic framework for Diffie–Hellman assumptions. *Journal of cryptology*, 30(1):242–288, 2017.

[38] H. Feng and Q. Tang. Computational robust (fuzzy) extractors for crs-dependent sources with minimal min-entropy. In *Theory of Cryptography Conference*, pages 689–717. Springer, 2021.

[39] P. Fenteany and B. Fuller. Same point composable and nonmalleable obfuscated point functions. In *International Conference on Applied Cryptography and Network Security*, pages 124–144. Springer, 2020.

[40] G. Fuchsbauer, E. Kiltz, and J. Loss. The algebraic group model and its applications. In *Annual International Cryptology Conference*, pages 33–62. Springer, 2018.

[41] S. D. Galbraith and L. Zobernig. Obfuscated fuzzy hamming distance and conjunctions from subset product problems. In *Theory of Cryptography Conference*, pages 81–110. Springer, 2019.

[42] S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, and B. Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 40–49. IEEE, 2013.

[43] S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, and B. Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. *SIAM Journal on Computing*, 45(3):882–929, 2016.

[44] C. Gentry, A. B. Lewko, A. Sahai, and B. Waters. Indistinguishability obfuscation from the multilinear subgroup elimination assumption. In *2015 IEEE 56th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 151–170. IEEE, 2015.

[45] R. Goyal, V. Koppula, and B. Waters. Lockable obfuscation. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 612–621. IEEE, 2017.

[46] I. Komargodski and E. Yogev. Another step towards realizing random oracles: Non-malleable point obfuscation. In *Advances in Cryptology – EUROCRYPT*, pages 259–279. Springer, 2018.

[47] I. Komargodski and E. Yogev. Another step towards realizing random oracles: Non-malleable point obfuscation. Cryptology ePrint Archive, Report 2018/149, 2018. Version 20190226:074205, `https://eprint.iacr.org/2018/149`.

[48] B. Lynn, M. Prabhakaran, and A. Sahai. Positive results and techniques for obfuscation. In *Advances in Cryptology–EUROCRYPT 2004*, pages 20–39. Springer, 2004.

[49] N. Nisan and D. Zuckerman. Randomness is linear in space. *Journal of Computer and System Sciences*, 52(1):43–52, 1996.

[50] R. Pass, K. Seth, and S. Telang. Indistinguishability obfuscation from semantically-secure multilinear encodings. In *Advances in Cryptology – CRYPTO*, pages 500–517. Springer, 2014.

[51] A. Sahai and B. Waters. How to use indistinguishability obfuscation: deniable encryption, and more. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 475–484. ACM, 2014.

[52] S. Simhadri, J. Steel, and B. Fuller. Cryptographic authentication from the iris. In *International Conference on Information Security*, pages 465–485. Springer, 2019.

[53] S. P. Vadhan et al. *Pseudorandomness*, volume 7. Now Delft, 2012.

[54] M. H. Varia. *Studies in program obfuscation*. PhD thesis, Massachusetts Institute of Technology, 2010.

[55] Y. Wen and S. Liu. Robustly reusable fuzzy extractor from standard assumptions. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 459–489. Springer, 2018.

[56] Y. Wen, S. Liu, and D. Gu. Generic constructions of robustly reusable fuzzy extractor. In *IACR International Workshop on Public Key Cryptography*, pages 349–378. Springer, 2019.

[57] Y. Wen, S. Liu, Z. Hu, and S. Han. Computational robust fuzzy extractor. *The Computer Journal*, 61(12):1794–1805, 2018.

[58] D. Wichs and G. Zirdelis. Obfuscating compute-and-compare programs under LWE. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 600–611. IEEE, 2017.

A Alternative Notions of Nonmalleability

We consider an alternative notion, which we call *fixed nonmalleability*, to contrast with the above adaptive definition. This definition is drawn from verifiable nonmalleability in the work of Canetti and Varia [25, Definition 6].

Definition A.1 (Fixed Nonmalleability)**.** *Let all notation be as in Definition 3.2 except that Nonmalleability is replaced with the following:*

3. ***Fixed Nonmalleability:*** *For any $X \in \mathcal{X}$, for all $\vec{\mathsf{ad}} \in \mathcal{AD}$, for any PPT $\mathcal{A}$, where $\forall f \in \mathcal{F}$, there exists $\epsilon = \mathtt{ngl}(\lambda)$, such that defining*

$$\{\mathsf{unlockPoint}_i \leftarrow \mathsf{lockPoint}(\mathsf{val}, \vec{\mathsf{ad}}_i)\}_{i=1}^{\ell},$$
$$(C, \mathsf{ad}^*) \leftarrow \mathcal{A}\left(\{\mathsf{unlockPoint}_i, \vec{\mathsf{ad}}_i\}_{i=1}^{\ell}\right)$$

it is true that :

$$\Pr_{\mathsf{val} \leftarrow X}\left[\begin{array}{r}\mathsf{V}_{\mathtt{obf}}(C) = 1, (I_{f(\mathsf{val}),\mathsf{ad}^*} \equiv C) \\ \vee((I_{\mathsf{val},\mathsf{ad}^*} \equiv C) \wedge (\forall j, \mathsf{ad}^* \neq \vec{\mathsf{ad}}_j))\end{array}\right] \leq \epsilon.$$

Note that adaptive nonmalleability as defined in Nonmalleability of Definition 3.3 is stronger than fixed nonmalleability as shown by [46, Claim 3.5]. Lastly, we consider a stronger notion possible notion which we call oblivious nonmalleability.

Definition A.2 (Oblivious Nonmalleability)**.** 3. ***Oblivious Nonmalleability:*** *For any $X \in \mathcal{X}$, for all $\vec{\mathsf{ad}} \in \mathcal{AD}$, for any PPT $\mathcal{A}$, where there exists $\epsilon = \mathtt{ngl}(\lambda)$, such that defining*

$$\{\mathsf{unlockPoint}_i \leftarrow \mathsf{lockPoint}(\mathsf{val}, \vec{\mathsf{ad}}_i)\}_{i=1}^{\ell},$$
$$(C, \mathsf{ad}^*) \leftarrow \mathcal{A}\left(\{\mathsf{unlockPoint}_i, \vec{\mathsf{ad}}_i\}_{i=1}^{\ell}\right)$$

it is true that :

$$\Pr_{\mathsf{val} \leftarrow X}\left[\begin{array}{r}\mathsf{V}_{\mathtt{obf}}(C) = 1, \exists f \in \mathcal{F}, (I_{f(\mathsf{val}),\mathsf{ad}^*} \equiv C) \\ \vee((I_{\mathsf{val},\mathsf{ad}^*} \equiv C) \wedge (\forall j, \mathsf{ad}^* \neq \vec{\mathsf{ad}}_j))\end{array}\right] \leq \epsilon.$$

Lemma A.1. *For any $\mathcal{F}$ that includes shifts by constants, no black box reduction can show that* lock *satisfies Definition A.2 in the plain model.*

Proof. Consider an $\mathcal{A}$ that is provided with $\mathsf{lockPoint}(\mathsf{val})$ and responds only with an obfuscation of an independent point obfuscator $\mathsf{lockPoint}(\mathsf{val}')$ for a random val'. Then $\mathcal{A}$ has successfully mauled to $\mathsf{lockPoint}(f(\mathsf{val}))$ for $f(x) = x - \mathsf{val} + \mathsf{val}'$, which is a valid low-degree but nonconstant polynomial. □

Note above $\mathsf{lockPoint}(\mathsf{val}')$ can be constructed without even seeing $\mathsf{lockPoint}(\mathsf{val})$. The above result also seems likely to hold in the algebraic group model (see [40] and Definition B.1). Recall that in the algebraic group model, an adversary outputs how they formed the obfuscation as a linear combination of the given obfuscation. Oblivious nonmalleability is unlikely to be attainable if the space of valid $\mathsf{unlockPoint}$ is dense as the adversary can just raise the provided group elements to a random power and output this as a new obfuscation.

B A simple nonmalleability argument in the Algebraic Group Model

In this section, we provide a simple intuitive proof on nonmalleability of our primary construction in the algebraic group model along with Assumption 3.2. The purpose of this proof is to give the reader intuition for the argument. Lemma 4.3 does not require use of the algebraic group model. Before introducing the new lemma we give some background definitions.

We introduce a weaker variant that is clearly implied by the Assumption 3.2:

Assumption B.1. *Let $\mathcal{G}$ and $\mathcal{X}_\lambda$ be defined as in Assumption 3.1. For any $\ell = \mathtt{poly}(\lambda)$ for any PPT $\mathcal{A}$,*

$$\Pr[x \leftarrow \mathcal{A}(\{k_i, [k_i x + x^i]_g\}_{i\in[2,..,\ell]})] = \mathtt{ngl}(\lambda).$$

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

We now restate the flexible uber assumption from [8] adapted to consider non-uniform distributions and distributions over input polynomials. Before doing that we have to define an algebraic algorithm.

Definition B.1. *An algorithm $\mathcal{A}$ executed in a game $\mathbb{G}$ is called* algebraic *if for all group elements $[\vec{z}]_g \in \mathbb{G}_\lambda$ that $\mathcal{A}$ outputs, it additionally tells how those elements were formed in terms of the received group elements. That is, $\mathcal{A}([\vec{x}]_g)$ provides a matrix $\mathbf{A}$ such that*

$$[\vec{z}]_g = [\mathbf{A}\vec{x}]_g$$

We use $\mathbb{Z}_p[X, r]$ to represent the set of polynomials over the $\mathbb{Z}_p$ of degree at most r.

Definition B.2 (Linear Independence of Polynomials)**.** *Let $\vec{\mathbf{R}} \in \mathbb{Z}_p[X, r]^k$ and let $\mathbf{W} \in \mathbb{Z}_p[X, r]$. The polynomial W is* linearly dependent *on $\vec{\mathbf{R}}$ if $\exists \alpha_1, ..., \alpha_k$ such that*

$$s\mathbf{W} = \sum_{i=1}^{k} \alpha_i \vec{\mathbf{R}}_i.$$

Otherwise $\mathbf{W}$ is called linearly independent *of $\mathbf{R}$.*

Assumption B.2 (Flexible Uber Assumption for Well-Spread Distributions)**.** *Let $\mathcal{G} = \{\mathbb{G}_\lambda\}_{\lambda\in\mathbb{N}}$ be a group ensemble with efficient representation and operations where each $\mathbb{G}_\lambda$ is a group of prime order $p \in (2^\lambda, 2^{\lambda+1})$. We assume that for every $\lambda \in \mathbb{N}$ there is a canonical (efficiently computable) group and canonical and efficient mapping between the elements of $\{0,1\}^\lambda$ to $\mathbb{G}_\lambda$. Let $\{\mathcal{X}_\lambda\}$ be a family of well-spread distributions over $\{0,1\}^\lambda$. Let $\vec{\mathbf{R}}$ be a distribution over vectors of polynomials over $\mathbb{Z}_p$ of degree at most ℓ. The $(\vec{\mathbf{R}}, \iota)$ flexible uber assumption is that no PPT algebraic adversary $A_{\mathtt{alg}}$ can output vectors $\mathbf{S}, \mathbf{T} \in \mathbb{Z}_p[X]^\iota$ and value g^* such that*

$$\Pr_{x\leftarrow\mathcal{X}_\lambda}\left[\begin{array}{r|l}(\mathbf{S}, \mathbf{T}, g^*) \leftarrow \mathcal{A}(\vec{\mathbf{R}}, [\vec{\mathbf{R}}(x)]_g) & \mathbf{T} \text{ l.i. of } \vec{\mathbf{R}} \\ \wedge g^* = [\mathbf{T}(x)]_g & \wedge \mathbf{S}\vec{\mathbf{R}} = \mathbf{T}\end{array}\right] = \mathtt{ngl}(\lambda).$$

We are now ready to provide an algebraic version of our nonmalleability argument. An algebraic version of Lemma **??** can also be stated.

Lemma B.1. *Let λ be a security parameter. Let $\{\mathcal{X}_\lambda\}$ be a well-spread distribution ensemble and let $\tau, \ell \in \mathbb{Z}^+$ be $\mathtt{poly}(\lambda)$. Let $\mathcal{F}_{poly}$ be the ensemble of functions f_λ where f_λ is the set of non-constant, non-identity polynomials in $\mathbb{Z}_{p(\lambda)}[x]$ with degree at most τ.*

Let $P(x) = r_\rho x^\rho + r_{\rho-1}x^{\rho-1} + \ldots + r_1 x + r_0$ with $r_i \in \mathbb{Z}$ defined in some way by ad where any or all of the r_i may be 0. Suppose that Assumption 3.1 holds for $\ell = \tau(\rho+7)$. Define as obfuscation (with a, b, c uniformly distributed in $p(\lambda)$)

$$\mathsf{lockPoint}_P(\mathsf{val}, \mathsf{ad}; a, b, c) \stackrel{def}{=} \begin{bmatrix} a, & \left[a\mathsf{val} + \mathsf{val}^2 P(\mathsf{val}) + \sum_{i=\rho+3}^{\rho+7} \mathsf{val}^i\right]_g \\ b, & \left[b\mathsf{val} + \mathsf{val}^{\rho+8}\right]_g \\ c, & \left[c\mathsf{val} + \mathsf{val}^{\rho+9}\right]_g \end{bmatrix}.$$

Consider $\mathcal{F}_{poly}$ and distribution ensemble $\{\mathcal{X}_\lambda\}$. For any PPT algebraic *adversary $\mathcal{A}$ for nonmalleability in Definition 3.2, $\mathcal{A}$ outputs $f, \mathsf{unlockPoint}_{P'}, P'$, where $\mathsf{lockPoint}_{P'}$ obfuscates $f(\mathsf{val})$, $f \in \mathcal{F}_{poly}$, and P' is a polynomial of degree at most ρ, with negligible probability.*

We use the following lemma showing that Assumption B.1 implies Assumption B.2.

Lemma B.2. *Let $\mathcal{G}$ be defined as above. Suppose that Assumption B.1 holds for power ℓ and that one can properly prepare an instance of $\vec{\mathbf{R}}$ using linear operations on $\{k_i, [k_i x + x^i]\}_{i\in[2,\ldots,\ell]}$, then Assumption B.2 holds assuming* $\deg(\mathbf{T}) = \mathtt{poly}(\lambda)$.

Proof. The core of the proof is as in [8, Theorem 4.1]. We consider nonuniform distributions over x and because of this consider randomized polynomial families rather than a deterministic vector of polynomials.

Let $\mathcal{A}$ be an algebraic adversary that succeeds against distributions from $\vec{\mathbf{R}}$ with an advantage at least ϵ. We restrict our attention to the case when $\mathbf{T}$ is linearly independent of $\vec{\mathbf{R}}$. We now construct an adversary $\mathcal{B}$ that can break Assumption B.1.

Recall that adversary $\mathcal{B}$ takes input $\{k_i, z_i = [k_i x + x^i]_g\}_{i\in[2,..,\ell]})$. Then $\mathcal{B}$ forms the polynomials $\vec{\mathbf{R}}(x)$ and provides $\vec{\mathbf{R}}, [\vec{\mathbf{R}}(x)]_g$ to $\mathcal{A}$. Now $\mathcal{A}$ returns $\mathbf{S}, \mathbf{T}$ and g^* as above. Define the polynomial $\mathbf{P} = \mathbf{T} - \mathbf{S}\vec{\mathbf{R}}$, since $\mathbf{T}$ is linearly independent of $\vec{\mathbf{R}}$ this polynomial must be nonzero and polynomial degree, denoted as β. Factor the polynomial $\mathbf{P}$ to find its roots $\alpha_1, ..., \alpha_\beta$. If for some root α_j it is true that $\forall i, z_i = [k_i\alpha_j + \alpha^i]_{i\in[2,\ldots,\ell]}$ output α_j else abort. Then $\mathcal{B}$ succeeds whenever $\mathcal{A}$ which completes the proof. This completes the proof of Lemma B.2. □

Lemma B.2 suffices to prove Lemma B.1 with the following observations:

1. A good obfuscation can be linearly formed from the tuple $\{k_i, z_i = [k_i x + x^i]_g\}_{i\in[2,..,\ell]})$ as long as $\ell \geq \rho + 9$. This was shown in the proof of Theorem 4.2.

2. From the tampering function f and values a', b', c', ad' one can compute the polynomials in the exponent. Namely, the polynomials in the exponent are:

$$a'f(\mathsf{val}) + \mathsf{ad} * f(\mathsf{val})^2 + \sum_{i=\rho+3}^{\rho+7} f(\mathsf{val})^i$$
$$b'f(\mathsf{val}) + f(\mathsf{val})^{\rho+8}$$
$$c'f(\mathsf{val}) + f(\mathsf{val})^{\rho+9}$$

3. That for all $f \in \mathcal{F}_{poly}$, the resulting polynomials in the exponent are linearly independent of the provided polynomials. Shown in proof of Lemma 4.3.