Constrained PRFs for Inner-Product Predicates from Weaker Assumptions

Sacha Servan-Schreiber⋆

MIT

Abstract. In this paper, we provide a novel framework for constructing Constrained Pseudorandom Functions (CPRFs) with inner-product constraint predicates, using ideas from subtractive secret sharing and related-key-attack security.

Our framework can be instantiated using a random oracle or any suitable Related-Key-Attack (RKA) secure pseudorandom function. This results in three new CPRF constructions:

1. an adaptively-secure construction in the random oracle model;
2. a selectively-secure construction under the DDH assumption; and
3. a selectively-secure construction with a polynomial domain under the assumption that one-way functions exist.

All three instantiations are constraint-hiding and support inner-product predicates, leading to the first constructions of such expressive CPRFs under each corresponding assumption. Moreover, while the OWF-based construction is primarily of theoretical interest, the random oracle and DDH-based constructions are concretely efficient, which we show via an implementation.

⋆ This work was done in part while at Microsoft Research, New England.

Table of Contents

1 Introduction

Constrained pseudorandom functions (CPRFs) [10, 16, 48] are pseudorandom functions (PRFs) with a "default mode" associated with a master key msk, and a "constrained mode" associated with a constrained key csk defined over a predicate C. The constrained key csk can be used to compute the same "default mode" value of the PRF for all inputs x where $C(x) = 0$. However, for all inputs x where $C(x) \neq 0$, the constrained key csk can only be used to compute a pseudorandom value that is computationally independent of the PRF value under msk.

In the basic definition of CPRFs, the constrained key csk can reveal the predicate C (i.e., all inputs x where $C(x) = 0$). For example, the GGM PRF [41], admits *puncturing* constraints [10, 16, 48], where the constraint C is a point function that outputs 0 on all-but-one input. In the GGM PRF, csk reveals the punctured point to the constraint key holder. An enhanced definition of CPRFs, first[1] formalized by Boneh, Lewi, and Wu [14] (PKC 2017), requires csk to hide C, and is much more challenging to achieve, even for simple constraints [14, 26, 34].

Constructing CPRFs for expressive constraint classes under standard assumptions has proven to be a challenging task. Several constructions exist for simple constraint classes, such as prefix-matching, bit-fixing, and constraints expressible by t-CNF formulas (with constant t) under various assumptions, including the minimal assumption that one-way functions exist (see the excellent survey of related works in [34, Appendix A]). However, even slightly more expressive constraints, such as constraints represented by inner products, constant-degree polynomials, or circuits in NC^1 (the class of functions

[1] Alternative notions of constraint PRFs were discovered concurrently in [16, 48].

computable by logarithmic-depth circuits), appear to be much more challenging to construct from standard assumptions [3, 26, 28, 30].

In a recent work, Couteau, Meyer, Passelègue, and Riahinia [30] (Eurocrypt 2023) were able to realize CPRFs for NC^1 from DCR (but without the constraint-hiding property), as well as *constraint-hiding* CPRF with inner-product constraint predicates, through an elegant connection to homomorphic secret sharing [18, 19, 21, 54]. In contrast, *constraint-hiding* CPRFs for NC^1 are only known under LWE [26, 28, 55] (or indistinguishability obfuscation [14, 27]) and can even *imply* indistinguishability obfuscation in certain cases [26]. Therefore, the result of Couteau et al. significantly pushes the constraint expressivity of CPRFs under the Decisional Composite Residuosity (DCR) assumption. Prior to their result, the only known constructions for constraint-hiding CPRFs with sufficiently powerful constraint predicates to evaluate inner-product constraints required either the learning with errors (LWE) assumption or non-standard assumptions [14, 26, 55]. However, in contrast to other constraint predicates that can be realized from one-way functions [10, 16, 34, 48], there is still a significant gap in our understanding of which assumptions are necessary for realizing CPRFs for more expressive constraint classes, such as inner-product and NC^1 predicates.

Motivation. In this paper, we revisit the assumptions required to construct constraint-hiding CPRFs for inner-product constraint classes. This is motivated by the existence of CPRFs for NC^1 from Diffie-Hellman-style assumptions [3], as well as constraint-hiding CPRFs for bit-fixing and (constant sized) t-CNF formulas from the minimal assumption that one-way functions exist [34]. Understanding what assumptions are required to realize sufficiently expressive CPRFs can shed light on realizing closely related "high-end" cryptographic primitives such as functional encryption [26, 38], searchable symmetric encryption [14], attribute-based encryption [3], and even obfuscation [26]. Specifically, in this paper, we ask:

Under what assumptions do constrained PRFs with inner-product predicates exist?

The motivation for studying inner-product constraints is that they can be used to construct CPRFs with constraint predicates represented by constant-degree polynomials and extensions thereof (see Appendix B for details), and are of interest both as a theoretical object and as a practical tool.

From a theoretical lens, the fact that inner-product predicates lie somewhere in between t-CNF and NC^1 predicates in terms of expressivity, motivates the study of CPRFs for inner-product predicates under weaker assumptions, with the goal of potentially finding new techniques that could lead to more expressive constraints under weaker assumptions. This was also the motivation behind Attrapadung et al. [3] and other works examining the assumptions required to build CPRFs. Indeed, Davidson et al. [34] prove that CPRFs for inner-product predicates imply CPRFs for constant t-CNFs predicates (see [34, Appendix C] and Appendix B), which in turn imply CPRFs for bit-fixing predicates.

From a practical perspective, the current lack of any *concretely efficient* CPRF constructions for inner-product predicates,[2] motivates the quest of finding assumptions under which efficient constructions can be realized. This is especially motivated by the hope that concretely efficient constructions of CPRFs for inner-product predicates will lead to interesting real-world applications, as has been the case for the concretely efficient constructions of CPRFs admitting puncturing constraints (e.g., [5, 6, 15, 22, 37, 44, 50, 52, 58, 59, 60]).

Contributions. In this paper, we make the following three contributions:

New constructions from new assumptions. We construct the first CPRFs for inner-product predicates with (1) adaptive security in the random oracle model, (2) selective security under the Decisional Diffie-Hellman (DDH) assumption, and (3) selective security with a polynomial input domain under the minimal assumption that One-way Functions (OWFs) exist. All three of our results push the frontier of what was previously known theoretically on CPRFs. Moreover, our constructions are all constraint-hiding by default.

A simple framework. We provide a simple framework that exploits the properties of subtractive secret sharing to construct CPRFs for inner-product predicates. Our framework makes explicit several ideas that have been used implicitly in many prior works on CPRFs (e.g., [3, 24, 25, 55]), and may prove useful in obtaining more results in the future.

[2] To the best of our knowledge, no constraint-hiding CPRF constructions have been implemented to date.

An implementation. Due to the simplicity of our building blocks, we show that our constructions result in the first *practical* constraint-hiding CPRFs under standard assumptions. We implement and benchmark our constructions, proving that they are concretely efficient. (All prior constructions of CPRFs for inner-product predicates, including the DCR-based construction of Couteau et al., require computationally expensive machinery, making them impractical.)

Extensions and Applications. Our framework has the following applications and extensions.

1. *More complex predicates.* From inner-product constraints, we can build CPRFs for more complex predicates via generic transformations, including constraints represented by *constant degree polynomials* and CPRFs for the "AND" of d distinct inner-product predicates. In particular, the latter allows us to construct matrix-product constraint predicates, where the constraint is satisfied if and only if $\mathbf{Ax} = \mathbf{0}$, for a constraint matrix $\mathbf{A}$.
2. *Lower-bounds in learning theory.* In learning theory, Membership Query (MQ) learning provides a model for quantifying the "learnability" or complexity of a certain class of functions [61]. Informally, in the MQ learning framework, a learner gets oracle access to a function and must approximate the function after making a sufficient number of queries. Cohen, Goldwasser, and Vaikuntanathan [29] introduce a model they call MQ *with Restriction Access* ($\mathrm{MQ}_{\mathsf{RA}}$), where in addition to black-box membership queries, the learner obtains non-black-box access to a restricted subset of the function. Obtaining (negative) results on the learnability of a particular class in the $\mathrm{MQ}_{\mathsf{RA}}$ model can be done using a connection to constrained PRFs; see Appendix B.

Followup Work. In a followup work, Couteau, Devadas, Devadas, Koch, and Servan-Schreiber [31] extend our CPRF construction to realize a *shiftable* CPRF, which allows the master key holder to emulate the PRF evaluation on the constrained key for different potential constraints. They then show how to use this extended CPRF to realize an efficient OT extension protocol with precomputability and a non-interactive public-key setup, which heavily exploits the concrete efficiency of our CPRF construction.

1.1 Related Work

In Table 1, we summarize known constructions of CPRFs for inner-product predicates (including existing constructions for more general predicates such as NC^1 and $\mathsf{P/poly}$) and highlight our results.

CPRFs for inner-product predicates. Attrapadung et al. [3] construct constrained PRFs for NC^1 (which includes inner-product predicates) from the L-decisional Diffie-Hellman inversion (L-DDHI) in combination with DDH over the quadratic residue subgroup QR_p (they can make their construction adaptively-secure by using a random oracle instead of DDH in QR_p), but their construction is not constraint-hiding. Similarly, Couteau et al. [30] also show how to construct CPRFs for NC^1 predicates from the DCR assumption through homomorphic secret sharing (but also fail to achieve constraint privacy). Couteau et al. [30] additionally show that their techniques can be used to construct a CPRF from DDH *with a polynomially-bounded input domain.* CPRFs for more general predicates are known from multi-linear maps [10, 13], indistinguishability obfuscation [4, 11, 14, 34, 45, 46], and LWE [24, 25, 26, 28, 55], and can be used to instantiate CPRFs with inner-product constraints under those assumptions.

Constraint-hiding CPRFs for inner-product predicates. Davidson et al. [34] (Crypto 2020) construct (weakly) constraint hiding CPRFs for inner-product predicates from the LWE assumption. Specifically, their construction satisfies a weaker privacy definition, in which the adversary does not get access to an evaluation oracle. Constraint-hiding CPRFs for more general predicates (that include inner-product predicates) are known from the LWE assumption [25, 26, 28, 55] and indistinguishability obfuscation [14]. To the best of our knowledge, Couteau et al. [30] are the first realize constraint-hiding CPRFs for inner-product predicates from a non-lattice assumption, specifically from DCR.

One-one CPRFs. Our framework (as well as some prior constructions of CPRFs [3, 30, 34]) shares some conceptual similarities to the construction of one-one constrained PRFs [56]—an information-theoretic primitive that can be viewed as a CPRF in the "no-evaluation security" model [3], with applications to conditional disclosure of secrets. However, their constructions cannot be used to realize the standard notion of CPRFs from standard assumptions.

	Assumption	Security	Hiding	Predicate	Practical	Comments
[24, 25, 26, 28, 55]	LWE	Selective	✓/ ✗	$\supseteq \mathsf{NC}^1$	✗	[24] is not constraint hiding
AMNYY18 [3]	L-DDHI	Selective	✗	NC^1	✗	L-DDHI in $\mathsf{QR}_p \land$ DDH in $\mathbb{G}$
AMNYY18 [3]	L-DDHI	Adaptive	✗	NC^1	✗	L-DDHI in $\mathsf{QR}_p \land$ ROM
DKNYY20 [34]	LWE	Adaptive	✗	IP	✗	Is *weakly* constraint hiding
CMPR23 [30]	DCR	Selective	✓	IP	✗	
CMPR23 [30]	DDH	Selective	✓	IP	✗	Polynomial input domain
Theorem 1	ROM	Adaptive	✓	IP	✓	
Theorem 3	DDH	Selective	✓	IP	✓	
Theorem 5	VDLPN	Selective	✓	IP	✗	Only for *weak* CPRFs
Theorem 8	OWF	Selective	✓	IP	✗	Polynomial input domain

Table 1: Related work on CPRFs for Inner-Product (IP) predicates from standard assumptions.
ROM = Random Oracle Model.
DDH = Decisional Diffie-Hellman assumption.
DCR = Decisional Composite Residuosity assumption.
L-DDHI = L-decisional Diffie-Hellman Inversion assumption.
VDLPN = Variable-density Learning Parity with Noise assumption [22].

1.2 Organization

In Section 2, we provide a technical overview highlighting the main ideas behind our framework and constructions. In Section 3, we cover the necessary preliminaries on CPRFs and RKA-secure PRFs. In Section 4, we present our framework and provide an adaptively secure CPRF construction for inner-product predicates in the random oracle model. In Section 5, we show that we can instantiate our framework from RKA-secure PRFs, without the need for a random oracle. In Section 6, we show how to instantiate our framework from one-way functions. In Section 7, we discuss the practical efficiency of our constructions. In Appendix B, we discuss extensions and applications.

2 Technical Overview

In this section, we provide an overview of our framework and constructions.

Background on CPRFs. Following prior works [25, 30], for PRF domain $\mathcal{X}$ and a constraint $C\colon \mathcal{X} \to \{0,1\}$, we write $C(x) = 0$ for "true" (authorized), and $C(x) \neq 0$ for "false" (unauthorized). CPRFs consist of a master secret key msk, which can be used to evaluate the PRF on *all* inputs in the domain. From msk, it must then be possible to efficiently sample a constrained key csk for a given constraint C, which can be used to evaluate the PRF on all inputs x in the domain where $C(x) = 0$. Constraint hiding CPRFs have the added property that C remains hidden given csk. See Section 3 for formal definitions.

2.1 Our Approach

We now explain the main technical ideas that underpin our framework for constructing CPRFs for inner-product predicates. We start by explaining how we can use the idea of subtractive secret sharing to construct a constraint predicate C for inner-product predicates, inspired by Couteau et al.

The power of subtractive secret sharing. Subtractive secret shares of a value s, which we denote by s_0 and s_1, have the property that $s_0 - s_1 = s$ (over $\mathbb{Z}$). By splitting s into two random shares s_0 and s_1, individually each share is independent of the secret s. To use subtractive secret sharing to construct CPRFs, the main idea is to exploit the *symmetry* between the two shares. Specifically, consider what happens when the secret s is zero. Because we have that $s_0 - s_1 = 0$, it follows that $s_0 = s_1$. This symmetry present in subtractive secret shares has enabled many efficient techniques for distributed computations [17, 18, 19, 20, 21, 23, 39, 54], and surprisingly, also applies to CPRFs [30]. Specifically, consider the inner-product constraint $C_{\mathbf{z}}$ parameterized by a vector $\mathbf{z}$ and defined as

$C_{\mathbf{z}}(\mathbf{x}) = \langle \mathbf{z}, \mathbf{x} \rangle$. Next, denote subtractive secret shares of the constraint vector $\mathbf{z}$ by $\mathbf{z_0}$ and $\mathbf{z_1}$, such that $\mathbf{z_0} - \mathbf{z_1} = \mathbf{z}$. Thanks to the aforementioned symmetry property, for all input vectors $\mathbf{x}$:

– If $\langle \mathbf{z}, \mathbf{x} \rangle = 0$ (i.e., $C_{\mathbf{z}}(\mathbf{x}) = 0$, authorized), then $\langle \mathbf{z_0}, \mathbf{x} \rangle = \langle \mathbf{z_1}, \mathbf{x} \rangle$, and
– If $\langle \mathbf{z}, \mathbf{x} \rangle \neq 0$ (i.e., $C_{\mathbf{z}}(\mathbf{x}) \neq 0$, unauthorized), then $\langle \mathbf{z_0}, \mathbf{x} \rangle \neq \langle \mathbf{z_1}, \mathbf{x} \rangle$.

In words, the constraint is satisfied if and only if both shares of the inner product are equal. Moreover, note that $\mathbf{z_1}$ can be sampled *after* $\mathbf{z_0}$, because $\mathbf{z_0}$ is a random value independent of the "secret" constraint $\mathbf{z}$. We now describe how we can use these properties of subtractive secret sharing to construct a CPRF.

Initial attempt (not secure). Our first idea, which unfortunately turns out to be not secure, is to let the master secret key $\mathsf{msk} = \mathbf{z_0}$, for a random $\mathbf{z_0}$. Then, for a given constraint vector $\mathbf{z}$, the constrained key is computed (on-the-fly) as $\mathsf{csk} = \mathbf{z_1}$, where $\mathbf{z_1} = \mathbf{z_0} - \mathbf{z}$. The intuition is that for all $\mathbf{x}$ where $\langle \mathbf{z}, \mathbf{x} \rangle = 0$ (i.e., for all authorized $\mathbf{x}$), both the master secret key and the constrained key can be used to derive *the same* key k. Specifically, we can simply let $k = \langle \mathbf{z_0}, \mathbf{x} \rangle = \langle \mathbf{z_1}, \mathbf{x} \rangle$. Using the key k, in conjunction with any PRF F, we can define the output of the evaluation on the input $\mathbf{x}$ to be $F_k(\mathbf{x})$. Additionally, for all $\mathbf{x}$ where $\langle \mathbf{z}, \mathbf{x} \rangle \neq 0$ (i.e., for all unauthorized $\mathbf{x}$), the master key and constrained key derive *different* PRF keys, which results in the constrained key outputting a pseudorandom value.

Unfortunately, while this initial attempt provides the necessary correctness properties, it is not secure for the following two reasons:

1. the CPRF adversary, knowing the constraint $\mathbf{z}$ and given $\mathbf{z_1}$ can trivially recover $\mathbf{z_0}$ (the master secret key) simply by computing $\mathbf{z_0} = \mathbf{z_1} + \mathbf{z}$, and
2. in the case where $\langle \mathbf{z}, \mathbf{x} \rangle \neq 0$, the derived key is still *related* to the master key msk, in that $\langle \mathbf{z_1}, \mathbf{x} \rangle = \langle \mathbf{z_0}, \mathbf{x} \rangle - \langle \mathbf{z}, \mathbf{x} \rangle$.

Couteau et al. [30] resolves these two issues by resorting to HSS. In particular, they only use the value of $\langle \mathbf{z}, \mathbf{x} \rangle$ (which each party can compute a share of given $\mathbf{z_0}$ and $\mathbf{z_1}$, respectively) as a conditional mask in a HSS computation that computes a PRF. As such, they require evaluating a PRF inside of HSS which makes their construction impractical. This is where our approach diverges from the one of Couteau et al., which we explain next.

Second attempt (secure). To fix our initial attempt, we must first prevent the adversary from recovering $\mathbf{z_0}$ (the master secret key) from the constrained key $\mathbf{z_1}$, while still guaranteeing the necessary property that $\langle \mathbf{z_0}, \mathbf{x} \rangle = \langle \mathbf{z_1}, \mathbf{x} \rangle$ whenever $\langle \mathbf{z}, \mathbf{x} \rangle = 0$. To achieve this, we exploit the linearity of inner products. Specifically, let $\mathbb{F}$ be a finite field of order at least 2^λ, for a security parameter λ. As before, we let $\mathsf{msk} := \mathbf{z_0}$, for a random $\mathbf{z_0} \in \mathbb{F}^\ell$. However, now we let $\mathsf{csk} := \mathbf{z_1}$, where $\mathbf{z_1} := \mathbf{z_0} - \Delta\mathbf{z}$, for a random scalar "shift" $\Delta \in \mathbb{F}$. Notice that when $\langle \mathbf{z}, \mathbf{x} \rangle = 0$,

$$\langle \mathbf{z_0}, \mathbf{x} \rangle = \langle \mathbf{z_0}, \mathbf{x} \rangle - \Delta \langle \mathbf{z}, \mathbf{x} \rangle = \langle \mathbf{z_0}, \mathbf{x} \rangle - \underbrace{\langle \Delta\mathbf{z}, \mathbf{x} \rangle}_{\substack{\text{By linearity of} \\ \text{inner products}}} = \langle \mathbf{z_1}, \mathbf{x} \rangle ,$$

which still guarantees that the master secret key and constrained key can be used to derive the same PRF key k, whenever $C(\mathbf{x}) = 0$. Moreover, because Δ is uniformly random over $\mathbb{F}$ (which has order at least 2^λ), $\mathbf{z_1}$ cannot be used to recover $\mathbf{z_0}$, even with knowledge of the constraint $\mathbf{z}$, thereby preventing the CPRF adversary from recovering the master secret key msk from the constrained key csk.

Now, with the random shift Δ, we ensure that the constrained key csk does not leak the master secret key, and forms the basis for our framework described in Section 4. However, we are still left with the second problem we identified in our initial attempt: The derived PRF keys are still *related* to the master secret key, which does *not* guarantee that the resulting PRF evaluation is pseudorandom to the adversary. To deal with this, we can use the random oracle model.

Construction in the random oracle model. One simple way to instantiate the CPRF with correlated keys is to instantiate the PRF with a random oracle H. This forms the basis for our first instantiation, which we describe in Section 4.1. In a nutshell, we show that, if we use the derived key $k = \langle \mathbf{z_1}, \mathbf{x} \rangle$ with a random oracle H as the PRF, then the construction $F_k(\mathbf{x}) := H(k, \mathbf{x})$ is a secure CPRF. Specifically, the random oracle ensures that each evaluation is uniformly random, while still guaranteeing both the master secret key and the constrained key derive the same k when the constraint is satisfied.

Removing the random oracle with an RKA-secure PRF. To remove the random oracle requirement, we show that we can use a "special" PRF that remains provably secure when evaluated with different *related* keys. Such PRFs are known as Related-Key-Attack (RKA) secure PRFs [8] and have been studied extensively [1, 2, 7, 8, 12, 22, 32, 40, 42, 51], yielding several constructions to choose from. This result is rather surprising, since prior works that require notions of correlation-robustness (e.g., [47, 49, 57]) could only be constructed from more powerful assumptions. In contrast, we show that constructing CPRFs with inner-product constraints requires a much weaker flavor of correlation-robustness satisfied by RKA-secure PRFs with affine key-derivation functions. In particular, this weaker notion of correlation-robustness can be instantiated unconditionally leading to our one-way function based CPRF construction in Section 6.

Suitable RKA-secure PRFs. As we have informally shown above, a fully "RKA-secure" PRF can be realized with a random oracle to remove correlations in the keys. However, constructions of RKA-secure PRFs exist from several standard assumptions. These constructions achieve security against adversaries that can adaptively query the PRF when keyed on arbitrary functions of the secret key. In particular, we require RKA-security against *affine* functions of the key (see Section 3 for definitions), which is a stronger notion compared to standard RKA-security against *additive* functions that is often considered in the literature. The affine function requirement eliminates many RKA-secure PRF constructions (e.g., [2, 7, 8, 12, 32, 40, 51]), leaving us only with the DDH-based RKA-secure PRF for affine functions of Abdalla et al. [1].

The DDH-based RKA-secure PRF forms the basis for our first instantiation in the standard model. However, we also show that we can use any (weak) PRF[3] that is RKA-secure against additive functions to instantiate our framework and obtain a (weak) CPRF for inner-product predicates. In particular, this allows us to use the VDLPN-based RKA-secure (weak) PRF of Boyle et al. [22].

Additionally, we show that we can adapt the one-way function based RKA-secure PRF of Applebaum and Widder [2] to instantiate our framework (under certain restrictions). Specifically, the PRF of Applebaum and Widder [2] is only secure against additive functions and requires the number of related keys that the adversary queries to be apriori bounded by some polynomial t (in the security parameter). While these restriction makes their RKA-secure PRF construction have limited applications elsewhere, we find that it is just sufficiently powerful to apply to our framework provided that we bound the magnitude of the input vectors to be polynomial in t and limit CPRF to a polynomially-sized domain. However, a problem is that their construction is only proven RKA-secure for *additive* functions of the key, which is not suitable to instantiate our framework. Fortunately, however, we can easily adapt their result to the case of *affine* functions, making it compatible with our framework and leading to the first Minicrypt CPRF construction for inner-product predicates. Prior to this, the only CPRF for inner-product predicates with a polynomial domain was based on DDH [30].

3 Preliminaries

3.1 Notation

We let λ denote the security parameter. We let $\mathbb{F}$ denote a finite field (e.g., integers mod p), $\mathbb{Z}$ denote the set of integers, and $\mathbb{N}$ denote the set of natural numbers. We let $\mathbb{F}^\times$ denote the set $\mathbb{F} \setminus \{0\}$. A vector $\mathbf{v} = (v_1, \ldots, v_n)$ is denoted using bold lowercase letters. Scalar multiplication with a vector is denoted $a\mathbf{v} = (av_1, \ldots, av_n)$ and the inner product between two vectors $\mathbf{a}$ and $\mathbf{b}$ is denoted $\langle \mathbf{a}, \mathbf{b} \rangle$. We let $\mathsf{poly}(\cdot)$ denote any polynomial and $\mathsf{negl}(\cdot)$ denote a negligible function. We say an algorithm $\mathcal{A}$ is *efficient* if it runs in probabilistic polynomial time. For a finite set S, we let $x \xleftarrow{R} S$ denote a uniformly random sample from S. Assignment from a possibly randomized algorithm $\mathcal{A}$ on input x is denoted $y \leftarrow \mathcal{A}(x)$ and intialization of y to the value x is denoted as $y := x$.

3.2 Constrained Pseudorandom Functions

We start by recalling the syntax and properties of constrained pseudorandom functions (CPRFs). For simplicity, we restrict the definition to 1-key, constraint-hiding CPRFs, which is the definition satisfied

[3] A weak PRF is secure if the adversary only queries it on random inputs.

by our constructions. We point to Boneh et al. [14] for a more general definition of constraint-hiding CPRFs (i.e., with polynomial-key security).

Definition 1 (Constrained Pseudorandom Functions; adapted from [14, 30]). *Let $\lambda \in \mathbb{N}$ be a security parameter. A Constrained Pseudorandom Function (CPRF) with key space $\mathcal{K} = \mathcal{K}_\lambda$, domain $\mathcal{X} = \mathcal{X}_\lambda$, and range $\mathcal{Y}$, that supports constraints represented by the class of circuits $\mathcal{C} = \{\mathcal{C}_\lambda\}_{\lambda \in \mathbb{N}}$, where $C_\lambda \colon \mathcal{X} \to \{0,1\}$, consists of the following four algorithms.*

- $\mathsf{KeyGen}(1^\lambda) \to \mathsf{msk}$. *Takes as input a security parameter λ. Outputs a master secret key $\mathsf{msk} \in \mathcal{K}$.*
- $\mathsf{Eval}(\mathsf{msk}, x) \to y$. *Takes as input the master secret key msk and input $x \in \mathcal{X}$. Outputs $y \in \mathcal{Y}$.*
- $\mathsf{Constrain}(\mathsf{msk}, C) \to \mathsf{csk}$. *Takes as input the master secret key msk and a constraint circuit $C \in \mathcal{C}$. Outputs a constrained key csk.*
- $\mathsf{CEval}(\mathsf{csk}, x) \to y$. *Takes as input the constrained key csk and an input $x \in \mathcal{X}$. Outputs $y \in \mathcal{Y}$.*

We let any auxiliary public parameters pp be an implicit input to all algorithms. A CPRF must satisfy the following correctness and security properties.

Correctness. *For all security parameters λ, all constraints $C \in \mathcal{C}$, and all inputs $x \in \mathcal{X}$ such that $C(x) = 0$ (authorized), it holds that:*

$$\Pr\left[\mathsf{Eval}(\mathsf{msk}, x) = \mathsf{CEval}(\mathsf{csk}, x) \;\middle|\; \begin{array}{l} \mathsf{msk} \leftarrow \mathsf{KeyGen}(1^\lambda), \\ \mathsf{csk} \leftarrow \mathsf{Constrain}(\mathsf{msk}, C) \end{array}\right] = 1 - \mathsf{negl}(\lambda).$$

(1-key, adaptive) Security. *A CPRF is (1-key, adaptively)-secure if for all efficient adversaries $\mathcal{A}$, the advantage of $\mathcal{A}$ in the following security experiment $\mathsf{Exp}^{\mathsf{sec}}_{\mathcal{A},b}(\lambda)$ is negligible in λ. Here, b denotes the challenge bit.*

1. ***Setup:*** *On input 1^λ, the challenger runs $\mathsf{msk} \leftarrow \mathsf{KeyGen}(1^\lambda)$, initializes the set $Q := \emptyset$, and runs $\mathcal{A}(1^\lambda)$.*
2. ***Pre-challenge queries:*** *$\mathcal{A}$ adaptively sends arbitrary inputs $x \in \mathcal{X}$ to the challenger. For each x, the challenger computes $y \leftarrow \mathsf{Eval}(\mathsf{msk}, x)$, sends y to $\mathcal{A}$, and proceeds to update $Q \leftarrow Q \cup \{x\}$.*
3. ***Constrain query:*** *$\mathcal{A}$ sends* one *constraint $C \in \mathcal{C}$ to the challenger. The challenger computes $\mathsf{csk} \leftarrow \mathsf{Constrain}(\mathsf{msk}, C)$, and sends csk to $\mathcal{A}$.*
4. ***Challenge query:*** *For the single challenge query, $\mathcal{A}$ sends input $x^* \in \mathcal{X}$ as its challenge query, subject to the restriction that $x^* \notin Q$ and $C(x^*) \neq 0$. If $b = 0$, the challenger computes $y^* \leftarrow \mathsf{Eval}(\mathsf{msk}, x^*)$. Else, if $b = 1$, the challenger picks $y^* \xleftarrow{R} \mathcal{Y}$. The challenger sends y^* to $\mathcal{A}$.*
5. ***Post-challenge queries:*** *$\mathcal{A}$ continues to adaptively query the challenger on inputs $x \in \mathcal{X}$, subject to the restriction that $x \neq x^*$. For each x, the challenger computes $y \leftarrow \mathsf{Eval}(\mathsf{msk}, x)$ and sends y to $\mathcal{A}$.*
6. ***Guess:*** *$\mathcal{A}$ outputs its guess b', which is the output of the experiment.*

$\mathcal{A}$ wins if $b' = b$, and its advantage $\mathsf{Adv}^{\mathsf{sec}}_{\mathcal{A}}(\lambda)$ is defined as

$$\mathsf{Adv}^{\mathsf{sec}}_{\mathcal{A}}(\lambda) := \left|\Pr\left[\mathsf{Exp}^{\mathsf{sec}}_{\mathcal{A},0}(\lambda) = 1\right] - \Pr\left[\mathsf{Exp}^{\mathsf{sec}}_{\mathcal{A},1}(\lambda) = 1\right]\right|,$$

where the probability is over the randomness of $\mathcal{A}$ and KeyGen.

Definition 2 (Constraint Privacy; adapted from [14, 30]). *A CPRF is (1-key, adaptive)-constraint-hiding if for all efficient adversaries $\mathcal{A}$, the advantage of $\mathcal{A}$ in the following security experiment $\mathsf{Exp}^{\mathsf{priv}}_{\mathcal{A},b}(\lambda)$ is negligible in λ. Here, b denotes the challenge bit.*

1. ***Setup:*** *On input 1^λ, the challenger runs $\mathsf{msk} \leftarrow \mathsf{KeyGen}(1^\lambda)$, initializes the set $Q := \emptyset$, and runs $\mathcal{A}(1^\lambda)$.*
2. ***Pre-challenge queries:*** *$\mathcal{A}$ adaptively sends arbitrary input values $x \in \mathcal{X}$ to the challenger. For each x, the challenger computes $y \leftarrow \mathsf{Eval}(\mathsf{msk}, x)$, sends y to $\mathcal{A}$, and proceeds to update $Q \leftarrow Q \cup \{x\}$.*
3. ***Constrain query:*** *$\mathcal{A}$ sends a* pair *of constraints $(C_0, C_1) \in \mathcal{C}^2$ to the challenger,* subject to the restriction that *$C_0(x) = C_1(x)$, for all $x \in Q$. The challenger computes $\mathsf{csk}^* \leftarrow \mathsf{Constrain}(\mathsf{msk}, C_b)$, and sends csk^* to $\mathcal{A}$.*

4. ***Post-challenge queries:*** *$\mathcal{A}$ adaptively sends arbitrary input values $x \in \mathcal{X}$ to the challenger, subject to the restriction that $C_0(x) = C_1(x)$. For each x, the challenger computes $y \leftarrow \mathsf{Eval}(\mathsf{msk}, x)$, and sends y to $\mathcal{A}$.*
5. ***Guess:*** *$\mathcal{A}$ outputs its guess b', which is the output of the experiment.*

$\mathcal{A}$ wins if $b' = b$ and its advantage $\mathsf{Adv}^{\mathsf{priv}}_{\mathcal{A}}(\lambda)$ is defined as

$$\mathsf{Adv}^{\mathsf{priv}}_{\mathcal{A}}(\lambda) := \left| \Pr\left[\mathsf{Exp}^{\mathsf{priv}}_{\mathcal{A},0}(\lambda) = 1\right] - \Pr\left[\mathsf{Exp}^{\mathsf{priv}}_{\mathcal{A},1}(\lambda) = 1\right] \right|,$$

where the probability is over the randomness of $\mathcal{A}$ and KeyGen.

Definition 3 ((1-key, selective) Security). *A CPRF as defined in Definition 1 is said to be (1-key,* selectively*)-secure if the adversary commits to the constraint C* before *querying the challenger [14]. That is, $\mathcal{A}$ sends the constraint C to the challenger before issuing any pre-challenge queries. The same applies to the constraint-privacy definition (Definition 2).*

Remark 1 (Unique evaluation queries). Without loss of generality, we can restrict the PRF adversary $\mathcal{A}$ to issuing only *unique* evaluation queries (as was also done in prior PRF formalizations [2, 3]). Note that the adversary is already restricted to a unique challenge query in the above definition.

3.3 RKA-secure PRFs

Here, we formalize the notion of related-key attack (RKA)-secure PRFs.

Remark 2 (Find-then-Guess Security). We slightly modify the standard defintion of RKA-secure PRFs (e.g., [8]) to better align with the syntax of constrained PRFs. In the basic definition, the adversary does not obtain evaluation queries from what is guaranteed to be the output of the PRF F on *some* key. However, we note that this extra evaluation oracle is without loss of generality, and is only added to syntactically simplify our proofs. This definition is known as the find-then-guess PRF security game [30, Definition 10] and implies the real-or-random PRF security game, albeit with a polynomial loss in security.

Definition 4 (Φ-restricted Adversaries). *An efficient RKA-PRF adversary $\mathcal{A}$ is said to be Φ-restricted if its oracle queries have a related-key derivation function ϕ chosen arbitrarily from a set of valid key derivation functions Φ.*

Definition 5 (Related-Key-Attack Secure PRFs [8]). *Let $\lambda \in \mathbb{N}$ be a security parameter and $\ell = \ell(\lambda) \in \mathsf{poly}(\lambda)$. Let $\mathcal{F} = \{F_k : \mathcal{X}_\lambda \to \mathcal{Y}\}_{k \in \mathcal{K}_\lambda}$ be a family of functions and $\Phi\colon \mathcal{K}_\lambda \to \mathcal{K}_\lambda$ be a family of related-key derivation functions. $\mathcal{F}$ is said to be an RKA-secure PRF family if for all efficient Φ-restricted adversaries $\mathcal{A}$, the advantage of $\mathcal{A}$ in the following security experiment $\mathsf{Exp}^{\mathsf{rka}}_{\mathcal{A},b}(\lambda)$ is negligible in λ. Here, b denotes the challenge bit.*

- ***Setup:*** *On input 1^λ, the challenger samples $k \xleftarrow{R} \mathcal{K}_\lambda$, initializes the set $Q := \emptyset$, and runs $\mathcal{A}(1^\lambda)$.*
- ***Pre-challenge queries:*** *For each query (ϕ, x), the challenger computes $y \leftarrow F_{\phi(k)}(x)$, sends y to $\mathcal{A}$, and proceeds to update $Q \leftarrow Q \cup \{(\phi, x)\}$.*
- ***Challenge query:*** *For the single challenge query (ϕ^*, x^*),* subject to the restriction that $(\phi^*, x^*) \notin Q$*, the challenger proceeds based on the bit b as follows. If $b = 0$, the challenger computes $y \leftarrow F_{\phi^*(k)}(x^*)$. If $b = 1$, the challenger samples $y \xleftarrow{R} \mathcal{Y}$. The challenger then sends y to $\mathcal{A}$.*
- ***Post-challenge queries:*** *For each query (ϕ, x),* subject to the restriction that $(\phi, x) \neq (\phi^*, x^*)$*, the challenger computes $y \leftarrow F_{\phi^*(k)}(x)$, and sends y to $\mathcal{A}$.*
- ***Guess:*** *$\mathcal{A}$ outputs its guess b', which is the output of the experiment.*

$\mathcal{A}$ wins if $b' = b$ and its advantage $\mathsf{Adv}^{\mathsf{rka}}_{\mathcal{A}}(\lambda)$ is defined as

$$\mathsf{Adv}^{\mathsf{rka}}_{\mathcal{A}}(\lambda) := \left| \Pr\left[\mathsf{Exp}^{\mathsf{rka}}_{\mathcal{A},0}(\lambda) = 1\right] - \Pr\left[\mathsf{Exp}^{\mathsf{rka}}_{\mathcal{A},1}(\lambda) = 1\right] \right|,$$

where the probability is over the randomness of $\mathcal{A}$ and choice of k.

Definition 6 (Affine Related-Key Derivation Functions [1]). *Let $\mathbb{F}$ be a finite field and let $n \geq 1$ be an integer, let the class Φ_{aff} (aff for affine) denote the class of functions from $\mathbb{F}^n$ to $\mathbb{F}^n$ that can be separated into n component functions consisting of degree-1 univariate polynomials. That is,*

$$\Phi_{\mathsf{aff}} := \left\{ \boldsymbol{\phi} : \mathbb{F}^n \to \mathbb{F}^n \mid \begin{array}{l} \boldsymbol{\phi} = (\phi_1, \ldots, \phi_n); \\ \forall i \in [n],\ \phi_i(k_i) = \gamma_i k_i + \delta_i,\ \gamma_i \neq 0 \end{array} \right\}.$$

Note that $\gamma_i \neq 0$ is necessary to make the derivation function non-trivial.

Remark 3. Note that Φ_{aff} captures additive and multiplicative relations, which we denote by $\Phi_+ \subset \Phi_{\mathsf{aff}}$ and $\Phi_\times \subset \Phi_{\mathsf{aff}}$, respectively.

4 The Basic Framework and Construction

In Construction 1, we present our basic framework for constructing CPRFs for inner-product predicates, and present an instantiation of it in the random oracle model in Section 4.1. We extend this framework and use it in conjunction with RKA-secure PRFs in Section 5 to realize CPRFs for inner-product predicates under DDH, VDLPN, and OWFs.

Construction 1 (The basic framework)**.**
Let λ be a security parameter, $\ell \geq 1$ be an integer, and $\mathbb{F}$ be a finite field of order at least 2^λ. For a key space $\mathcal{K}$ and range $\mathcal{Y}$, a suitable choice of efficiently computable deterministic function $\mathsf{map}\colon \mathbb{F} \to \mathcal{K}$, and a PRF family $\mathcal{F} = \left\{F_k \colon \mathbb{F}^\ell \to \mathcal{Y}\right\}_{k \in \mathcal{K}}$, the CPRF algorithms are defined as:

$\mathsf{KeyGen}(1^\lambda, \ell)$:

1 : $k_0 \xleftarrow{R} \mathbb{F}$
2 : $\mathbf{z_0} \xleftarrow{R} \mathbb{F}^\ell$
3 : $\mathsf{msk} := (k_0, \mathbf{z_0})$

$\mathsf{Eval}(\mathsf{msk}, \mathbf{x})$:

1 : **parse** $\mathsf{msk} = (k_0, \mathbf{z_0})$
2 : $\delta_{\mathbf{x}} := \langle \mathbf{z_0}, \mathbf{x} \rangle$
3 : $k \leftarrow \mathsf{map}(k_0 + \delta_{\mathbf{x}})$
4 : **return** $F_k(\mathbf{x})$

$\mathsf{Constrain}(\mathsf{msk}, \mathbf{z})$:

1 : **parse** $\mathsf{msk} = (k_0, \mathbf{z_0})$
2 : $\Delta \xleftarrow{R} \mathbb{F}^\times$
3 : $\mathbf{z_1} := \mathbf{z_0} - \Delta \mathbf{z}$
4 : **return** $\mathsf{csk} := (k_0, \mathbf{z_1})$

$\mathsf{CEval}(\mathsf{csk}, \mathbf{x})$:

1 : **parse** $\mathsf{csk} = (k_0, \mathbf{z_1})$
2 : $\delta_{\mathbf{x}} := \langle \mathbf{z_1}, \mathbf{x} \rangle$
3 : $k \leftarrow \mathsf{map}(k_0 + \delta_{\mathbf{x}})$
4 : **return** $F_k(\mathbf{x})$

4.1 Instantiation via a Random Oracle

The simplest instantiation of Construction 1 is to let $F_k(\mathbf{x}) := H(k, \mathbf{x})$ where $H\colon \mathcal{K} \times \mathbb{F}^\ell \to \mathcal{Y}$ is a random oracle. Doing so ensures that when $\langle \mathbf{z}, \mathbf{x} \rangle \neq 0$, the output is uniformly random and independent of the constrained key csk, which guarantees that the evaluation under msk is independent of csk. We prove the following theorem.

Theorem 1. *Let λ be a security parameter, $\ell \geq 1$ be any integer, $\mathbb{F}$ be a finite field of order at least 2^λ, and map be any entropy-preserving map. Construction 1 is a (1-key, adaptively-secure, constraint-hiding) CPRF in the random oracle model when $\mathcal{F} = \left\{F_k \colon \mathbb{F}^\ell \to \mathcal{Y}\right\}_{k \in \mathcal{K}}$ is a PRF family, where $F_k(\mathbf{x}) := H(k, \mathbf{x})$ for all $k \in \mathcal{K}$ and $\mathbf{x} \in \mathbb{F}^\ell$, and where $H\colon \mathcal{K} \times \mathbb{F}^\ell \to \mathcal{Y}$ is a random oracle.*

Proof. We prove each required property in turn.

Correctness. Correctness follows from the intuition presented in Section 2. For all constraints $\mathbf{z}$ and inputs $\mathbf{x}$, whenever $\langle \mathbf{z}, \mathbf{x} \rangle = 0$, we have that

$$\delta_{\mathbf{x}} = \langle \mathbf{z_0}, \mathbf{x} \rangle = \langle \mathbf{z_0}, \mathbf{x} \rangle + \langle \mathbf{z}, \mathbf{x} \rangle = \langle \mathbf{z_0}, \mathbf{x} \rangle + \langle \Delta \mathbf{z}, \mathbf{x} \rangle = \langle \mathbf{z_1}, \mathbf{x} \rangle .$$

Therefore, Eval and CEval (of Construction 1) compute the same key k, because both Eval and CEval add the same shift $\delta_{\mathbf{x}}$ to the starting key k_0. It then follows that the evaluation is identical under the master key and the constrained key given that F_k is deterministic.

(1-key, adaptive) Security. Our proof consists of a sequence of hybrid games. First, we begin by noting that $H(k_0, \mathbf{x})$ trivially satisfies the definition of a pseudorandom function when H is a random oracle and k_0 has sufficient entropy to prevent guessing.

Hybrid $\mathcal{H}_0$. This hybrid consists of the (1-key, adaptive) CPRF security game.[4] We note that here, the challenger provides an oracle $\mathcal{O}_H$ via which the adversary $\mathcal{A}$ queries the random oracle H. We place the following restrictions on the adversary: (1) each query issued by $\mathcal{A}$ to the challenger (including queries to $\mathcal{O}_H$) is unique and (2) the adversary queries the pre- and post-challenge oracles *only* on constrained queries after issuing the single constraint query. These restrictions are without loss of generality in the 1-key setting, and have been used in prior work (e.g., [3, 34]).

Hybrid $\mathcal{H}_1$. In this hybrid game, the challenger starts by sampling responses to the pre-challenge evaluation queries until the constraint query is issued. That is, for a bound q_0 on the number of pre-challenge queries issued by $\mathcal{A}$, the challenger samples $v_1, \ldots, v_{q_0} \xleftarrow{\text{R}} \mathcal{Y}$. Then, the challenger responds to the i-th pre-challenge query $\mathbf{x_i}$ with v_i, and programs the random oracle $\mathcal{O}_H$ to output v_i on all future queries r when it holds that $r = k_0 + \langle \mathbf{z_0}, \mathbf{x_i} \rangle$, for some $i \in [q_0]$. After the constraint query is issued, the challenger responds to pre-challenge queries as in $\mathcal{H}_0$.

Claim. $\mathcal{A}$'s advantage in $\mathcal{H}_1$ is at most $\mathsf{negl}(\lambda)$ larger compared to $\mathcal{H}_0$.

Proof. Let q_H be the total number of random oracle queries issued by the adversary prior to the constrain query, and let r_i for $i \in [q_0]$ be the queries to $\mathcal{O}_H$. We define the event bad_0 as:

$$\exists (i, j) \in [q_H] \times [q_0] \text{ such that } r_i = (k_0 + \langle \mathbf{z_0}, \mathbf{x_j} \rangle, \mathbf{x_j}) \wedge H(r_i) \neq v_j.$$

The event bad_0 corresponds to the case where the adversary happens to query the random oracle $\mathcal{O}_H$ on a "bad" input r_i *prior* to the challenger programming $\mathcal{O}_H$ to output v_i, causing the response to be inconsistent with respect to $\mathcal{H}_0$. For any given pre-challenge query $\mathbf{x_j}$ issued before the constrain query, the probability that $\mathcal{A}$ issues an r_i query to $\mathcal{O}_H$ such that $r_i = (k_0 + \langle \mathbf{z_0}, \mathbf{x_j} \rangle, \mathbf{x_j})$ is equivalent to the probability of guessing k_0, which is bounded by $\frac{1}{|\mathbb{F}|}$. Hence, by a union bound, we get that

$$\Pr_{k_0 \xleftarrow{\text{R}} \mathbb{F}} [\mathsf{bad}_0] \leq \frac{q_H \cdot q_0}{|\mathbb{F}|} \leq \frac{q_H \cdot q_0}{2^\lambda} \leq \mathsf{negl}(\lambda),$$

which bounds $\mathcal{A}$'s advantage in $\mathcal{H}_1$ to a negligible function in λ.

Hybrid $\mathcal{H}_2$. In this hybrid game, we swap the definition of the constrained key and master key. Specifically, in this game, the challenger responds to $\mathcal{A}$'s constrain query $\mathbf{z}$ by sampling $\mathbf{z_1} \xleftarrow{\text{R}} \mathbb{F}^\ell$ and sending back $\mathsf{csk} = \mathbf{z_1}$. The challenger then samples $\Delta \xleftarrow{\text{R}} \mathbb{F}$, computes $\mathbf{z_0} = \mathbf{z_1} + \Delta \mathbf{z}$, and responds to future evaluation queries using $\mathbf{z_0}$ as the master key.

Claim. $\mathcal{A}$'s advantage in $\mathcal{H}_2$ is equivalent to its advantage in $\mathcal{H}_1$.

Proof. This change is purely syntactic and therefore does not affect the distribution of the keys. In particular, note that in $\mathcal{H}_1$, all evaluation queries prior to the constrain query are sampled independently of the master key. As such, it can be sampled at the time of the constraint query.

Hybrid $\mathcal{H}_3$. In this hybrid game, the challenger samples $w_1, \ldots, w_{q_1} \xleftarrow{\text{R}} \mathcal{Y}$ as the responses to the q_1 pre- and post-challenge evaluation queries issued following the constrain query. Then, the challenger responds to $\mathcal{A}$'s i-th evaluation query $\mathbf{x_i}$, where $\langle \mathbf{z}, \mathbf{x_i} \rangle \neq 0$ (recall the restriction in $\mathcal{H}_0$), with w_i.

[4] An alternative proof strategy is to use the framework of Attrapadung et al. [4] and show that $H(k_0 + \langle \mathbf{z_0}, \mathbf{x} \rangle, \mathbf{x})$ is a *no-evaluation* secure CPRF (similar to the CPRF game but the adversary does not get access to an evaluation oracle). They prove that any no-evaluation secure and "collision-resistant" CPRF becomes adaptively secure in the ROM when the output is passed through a random oracle. However, this then necessitate making the construction of the form $H'(H(k_0 + \langle \mathbf{z_0}, \mathbf{x} \rangle, \mathbf{x}))$ or arguing why $H'(H(\cdot))$ is equivalent to $H(\cdot)$ in the ROM. We opt here to prove adaptive security directly for completeness.

Claim. $\mathcal{A}$'s advantage in $\mathcal{H}_3$ is at most $\mathsf{negl}(\lambda)$ larger compared to $\mathcal{H}_2$.

Proof. Here, we let q_H be a bound on the total number of random oracle queries issued by the adversary throughout the game and let q_1 be a bound on the number of pre- and post-challenge evaluation queries issued following the constrain query. We then define the event bad_1 as:

$$\exists (i,j) \in [q_H] \times [q_1] \text{ such that } r_i = (k_0 + \langle \mathbf{z_0}, \mathbf{x_j} \rangle, \mathbf{x_j}) \wedge H(r_i) \neq w_j,$$

where r_i is a query to $\mathcal{O}_H$ issued by $\mathcal{A}$ and each $\mathbf{x_j}$ is constrained by assumption. The event bad_1 corresponds to the case where the adversary happens to query $\mathcal{O}_H$ on an input corresponding to a constrained evaluation under the master key msk, causing the response to be inconsistent with respect to the distribution in $\mathcal{H}_2$. For all post-constraint evaluation queries $\mathbf{x_j}$, where $j \in [q_1]$, define $y_j = H(k_0 + \langle \mathbf{z_0}, \mathbf{x_j} \rangle, \mathbf{x_j})$, which is computed identically to a post-constraint evaluation response in hybrid $\mathcal{H}_2$. We claim that y_j is computed independently of the constrained key $\mathsf{csk} = \mathbf{z_1}$. To see this, note that we can equivalently express y_j in terms of $\mathbf{z_1}$ as $y_j = H(k_0 + \langle \mathbf{z_1}, \mathbf{x_j} \rangle + \Delta \langle \mathbf{z}, \mathbf{x_j} \rangle, \mathbf{x_j})$, where and $\langle \mathbf{z}, \mathbf{x_j} \rangle \neq 0$ by assumption. Then, because Δ is uniformly random and independent of $\mathbf{z_1}$ in $\mathcal{H}_2$, each y_j is computed using a random oracle H that is "seeded" by $\Delta \langle \mathbf{z}, \mathbf{x_j} \rangle$, which makes the response independent of $\mathbf{z_1}$. Then, to compute the probability of the event bad_1, over the choice of $\Delta \in \mathbb{F}$, we can apply a union bound over all q_1 post-constraint evaluation queries issued by $\mathcal{A}$ to get

$$\Pr_{\Delta \xleftarrow{R} \mathbb{F}} [\mathsf{bad}_1] \leq \frac{q_H \cdot q_1}{|\mathbb{F}|} \leq \frac{q_H \cdot q_1}{2^\lambda} = \mathsf{negl}(\lambda),$$

which bounds the adversary's advantage in $\mathcal{H}_3$ to a negligible function in λ.

Hybrid $\mathcal{H}_4$. This hybrid consists of the find-then-guess PRF security game with PRF $F_k(\mathbf{x}) := H(k, \mathbf{x})$. Specifically, the challenger samples a random bit $b \in \{0,1\}$. If $b = 0$, the challenger samples a random $k \xleftarrow{R} \mathbb{F}$, then computes $y^* \leftarrow F_k(\mathbf{x}^*)$. Else, if $b = 1$, the challenger picks $y^* \xleftarrow{R} \mathcal{Y}$. In both cases, the challenger sends y^* to $\mathcal{A}$.

Claim. $\mathcal{A}$'s advantage in $\mathcal{H}_4$ is equivalent to its advantage in $\mathcal{H}_3$.

Proof. The claim follows by a straightforward reduction. In particular, note that all evaluation queries in $\mathcal{H}_3$ are sampled independently of Δ. Therefore, Δ is only used by the challenger in $\mathcal{H}_3$ to respond to the challenge query, which is equivalent to sampling a uniformly random and independent key k to answer the challenge query $\mathbf{x}^*$ in $\mathcal{H}_3$ given that $k_0 + \langle \mathbf{z_0}, \mathbf{x}^* \rangle = k_0 + \langle \mathbf{z_1}, \mathbf{x}^* \rangle + \Delta \langle \mathbf{z}, \mathbf{x}^* \rangle$ is a uniformly random value in $\mathbb{F}$ when Δ is sampled uniformly and independently of $\mathbf{z_1}$. ■

Constraint Privacy. We must prove that for all $\mathbf{z}$ and $\mathbf{z}'$ provided by the adversary $\mathcal{A}$, the constrained key, and all evaluation and challenge queries, do not reveal whether the constraint $\mathbf{z}$ or $\mathbf{z}'$ is used by the challenger.

First, we begin by noting that, even given $(\mathbf{z}, \mathbf{z}', \Delta)$, $\mathbf{z_0} + \Delta\mathbf{z}$ is distributed identically to $\mathbf{z_0} + \Delta\mathbf{z}'$ because $\mathbf{z_0}$ is uniformly random and independent of $\mathbf{z}$ and $\mathbf{z}'$. Therefore, the constrained key, absent the evaluation queries, is efficiently simulatable regardless of the constraint chosen by the challenger.

Now, we must show that this remains the case even when the adversary is given access to the evaluation and challenge oracles. Observe that we can proceed via the same sequence of hybrids used in the security proof above. Note that in the game defined by Hybrid $\mathcal{H}_1$, each constrained query is answered using a uniformly random key $k_i \in \mathcal{K}$. As such, the evaluation queries on constrained inputs are independent of the constraint, which guarantees that $\mathcal{A}$ cannot distinguish between $\mathbf{z}$ and $\mathbf{z}'$ with better than negligible advantage. ■

Remark 4 (Replacing the random oracle with a correlated-input secure hash). As noted by several prior works (e.g., [35, 42, 47]), the random oracle model is an overkill when all that is required is a notion of "correlation-robustness." Specifically, in our case, all we require is that H removes specific types of correlations present in its inputs. With this in mind, we can replace the random oracle H with a correlated-input secure hash (CIH) function [3, 35, 42, 47]. At a high level, a CIH is a publicly parameterized function H whose outputs "look random and independent" to a computationally-bounded adversary, even when the inputs are correlated. Specifically, we require the CIH to be secure against *affine* correlations between the inputs. The proof of security for Theorem 1 then follows the same blueprint, but instead hinges on the correlated-input security of H to ensure that the outputs are computationally indistinguishable from uniform. Unfortunately, we are not aware of an adaptively-secure CIH function construction (to the best of our knowledge, all existing constructions are in the selective-security regime). However, we note that there exist strong connections between CIH functions and RKA-PRFs, as discussed in-depth by Goyal, O'Neill, and Rao [42]. RKA-PRFs form the basis of our next instantiation of Construction 1.

5 Generalized Framework and Constructions

In this section, we instantiate our framework via RKA-secure PRFs. In Section 5.1, we start by extending the basic framework from Section 4 to make it more amenable with RKA-secure PRF constructions. We then prove that this framework yields constraint-hiding CPRFs from any RKA-secure PRF supporting Φ_{aff} key derivation functions. In Sections 5.2 and 5.3, we plug in the DDH-based and VDLPN-based RKA-secure PRF constructions into the framework. We defer instantiating the framework with our OWF-based RKA-secure PRF to Section 6, as there we must first construct a Φ_{aff}-RKA-secure PRF from OWFs.

5.1 Extended Framework

Existing constructions of RKA-secure PRFs (e.g., [1, 2, 7, 22]) have a key that is a *vector* of n field elements. As such, we cannot directly instantiate Construction 1 because the inner products are performed in $\mathbb{F}$ but the keys live in the vector space $\mathbb{F}^n$ (or subfield thereof). We therefore provide an *extended* version of our framework in Construction 2, that can be instantiated with the parameters of existing RKA-secure PRFs. At a high level, to accommodate keys that consist of vectors of n elements, we apply Construction 1 independently n times to derive a key for each coordinate. Formally, we capture this in Construction 2.

Construction 2 (The extended framework)**.**
Let λ be a security parameter, $n, \ell \geq 1$ be integers, and $\mathbb{F}$ be a finite field. For a key space $\mathcal{K}$ and range $\mathcal{Y}$, a suitable choice of efficiently computable deterministic function $\mathsf{map}\colon \mathbb{F}^n \to \mathcal{K}$, and a PRF family $\mathcal{F} = \left\{F_k \colon \mathbb{F}^\ell \to \mathcal{Y}\right\}_{k \in \mathcal{K}}$, the CPRF algorithms are defined as:

$\mathsf{KeyGen}(1^\lambda, \ell)$:

- $\mathbf{k_0} \xleftarrow{R} \mathbb{F}^n$
- **foreach** $i \in [n]$:
  - $\mathbf{z_0}_i \xleftarrow{R} \mathbb{F}^\ell$
- $\mathsf{msk} := (\mathbf{k_0}, \mathbf{z_0}_1, \ldots, \mathbf{z_0}_n)$

$\mathsf{Eval}(\mathsf{msk}, \mathbf{x})$:

- **parse** $\mathsf{msk} = (\mathbf{k_0}, \mathbf{z_0}_i, \ldots, \mathbf{z_0}_n)$
- **foreach** $i \in [n]$:
  - $\delta_{\mathbf{x}i} := \langle \mathbf{z_0}_i, \mathbf{x} \rangle$
- $\boldsymbol{\delta}_\mathbf{x} := (\delta_{\mathbf{x}1}, \ldots, \delta_{\mathbf{x}n})$
- $k \leftarrow \mathsf{map}(\mathbf{k_0} + \boldsymbol{\delta}_\mathbf{x})$
- **return** $F_k(\mathbf{x})$

$\mathsf{Constrain}(\mathsf{msk}, \mathbf{z})$:

- **parse** $\mathsf{msk} = (\mathbf{k_0}, \mathbf{z_0}_i, \ldots, \mathbf{z_0}_n)$
- **foreach** $i \in [n]$:
  - $\Delta_i \xleftarrow{R} \mathbb{F}$
  - $\mathbf{z_1}_i := \mathbf{z_0}_i - \Delta_i \mathbf{z}$
- **return** $\mathsf{csk} := (\mathbf{k_0}, \mathbf{z_1}_1, \ldots, \mathbf{z_1}_n)$

$\mathsf{CEval}(\mathsf{csk}, \mathbf{x})$:

- **parse** $\mathsf{csk} := (\mathbf{k_0}, \mathbf{z_1}_1, \ldots, \mathbf{z_1}_n)$
- **foreach** $i \in [n]$:
  - $\delta_{\mathbf{x}i} := \langle \mathbf{z_1}_i, \mathbf{x} \rangle$
- $\boldsymbol{\delta}_\mathbf{x} := (\delta_{\mathbf{x}1}, \ldots, \delta_{\mathbf{x}n})$
- $k \leftarrow \mathsf{map}(\mathbf{k_0} + \boldsymbol{\delta}_\mathbf{x})$
- **return** $F_k(\mathbf{x})$

Theorem 2. *Let $\mathbb{K}$ be a subfield of $\mathbb{F}$ and let the PRF key space $\mathcal{K} = \mathbb{K}^n$. Fix* map *to be any non-trivial ring homomorphism applied component-wise. If $\mathcal{F}$ is a family of RKA-secure pseudorandom functions with respect to affine related key derivation functions Φ_{aff}, as defined in Definition 6, then Construction 2 instantiated with $\mathcal{F}$ is a (1-key, selectively-secure, constraint-hiding) CPRF.*

Proof. We prove the required properties in turn.

Correctness. For all constraints $\mathbf{z}$ and inputs $\mathbf{x}$, whenever $\langle \mathbf{z}, \mathbf{x} \rangle = 0$, we have that $\delta_{\mathbf{x}i} = \langle \mathbf{z_0}_i, \mathbf{x} \rangle = \langle \mathbf{z_0}_i, \mathbf{x} \rangle + \Delta_i \langle \mathbf{z}, \mathbf{x} \rangle = \langle \mathbf{z_0}_i, \mathbf{x} \rangle + \langle \Delta_i \mathbf{z}, \mathbf{x} \rangle = \langle \mathbf{z_1}_i, \mathbf{x} \rangle \in \mathbb{F}$. Therefore, the resulting $\boldsymbol{\delta}_{\mathbf{x}}$ (as computed in Eval and CEval of Construction 1) is the same. Moreover, this holds for all $i \in [n]$, and because map is a ring homomorphism to a subfield of $\mathbb{F}$, the resulting keys are also identical when $\langle \mathbf{z}, \mathbf{x} \rangle = 0$. It then follows that the PRF evaluation is identical under the master key and the constrained key, because both Eval and CEval add the same $\boldsymbol{\delta}_{\mathbf{x}}$.

(1-key, selective) Security. We prove security by a reduction to the RKA-security of $\mathcal{F}$. Our proof consists of a sequence of hybrid games.

Hybrid $\mathcal{H}_0$. This hybrid consists of the (1-key, selective) CPRF security game.

Hybrid $\mathcal{H}_1$. In this hybrid, the challenger first samples the constrained key and *then* samples the master key. Specifically, at the start of the game, given the constraint $\mathbf{z}$ (we're in the selective security regime), the challenger first samples the constrained key $\mathsf{csk} := (\mathbf{k_0}, \mathbf{z_1}_1, \ldots, \mathbf{z_1}_n)$, where $\mathbf{k_0} \xleftarrow{R} \mathbb{F}^n$ and $\mathbf{z_1}_i \xleftarrow{R} \mathbb{F}^\ell$, for all $i \in [n]$. Then, the challenger computes the master secret key as $\mathsf{msk} := (\mathbf{k_0}, \mathbf{z_0}_1, \ldots, \mathbf{z_0}_n)$, where $\mathbf{z_0}_i := \mathbf{z_1}_i + \Delta_i \mathbf{z}$ and $\Delta_i \xleftarrow{R} \mathbb{F}$, for all $i \in [n]$.

Claim. $\mathcal{A}$'s advantage in $\mathcal{H}_1$ is identical to $\mathcal{A}$'s advantage in $\mathcal{H}_0$.

Proof. The claim follows immediately by observing that the distribution of msk and csk in $\mathcal{H}_1$ is identical to $\mathcal{H}_0$, because the change is merely syntactic.

Hybrid $\mathcal{H}_2$. In this hybrid game, the challenger does not sample Δ anymore. Instead, it is given access to the following stateful oracle $\mathcal{O}_{\mathsf{rka}}$:

Oracle $\mathcal{O}_{\mathsf{rka}}$

Initialize. Sample $\Delta \xleftarrow{R} \mathbb{K}^n$.

Evaluation. On input a affine function $\phi \in \Phi_{\mathsf{aff}}$ and $\mathbf{x} \in \mathbb{F}^\ell$, return $F_{\phi(\Delta)}(\mathbf{x})$.

The challenger is then defined as follows.

1. **Setup:** On input $(1^\lambda, \mathbf{z})$, $\mathcal{B}$ initializes $Q := \emptyset$, samples csk according to $\mathcal{H}_1$ by sampling $\mathbf{k_0} \xleftarrow{R} \mathbb{F}^n$, and $\mathbf{z_1}_i \xleftarrow{R} \mathbb{F}^\ell$, for all $i \in [n]$, and runs $\mathcal{A}$ on input $\mathsf{csk} := (\mathbf{k_0}, \mathbf{z_1}_1, \ldots, \mathbf{z_1}_n)$.
2. **Pre-challenge queries:** For each query $\mathbf{x}$ issued by $\mathcal{A}$, the challenger updates $Q \leftarrow Q \cup \{\mathbf{x}\}$, then does the following to compute y:
   - Compute $a_i := \mathsf{map}(\langle \mathbf{z}, \mathbf{x} \rangle)$ and $b_i := \mathsf{map}(\mathbf{k_0}_i + \langle \mathbf{z_1}_i, \mathbf{x} \rangle)$, for all $i \in [n]$.
   - Set $\phi \colon \mathbf{u} \mapsto \mathbf{a} \circ \mathbf{u} + \mathbf{b}$ where $\mathbf{a} := (a_1, \ldots, a_n)$ and $\mathbf{b} := (b_1, \ldots, b_n)$, where $\circ$ denotes the component wise (i.e., Hadamard) product.
   - Query $\mathcal{O}_{\mathsf{rka}}$ on input $(\phi, \mathbf{x})$, and forward the response y to $\mathcal{A}$.
     ▷ Note that y is computed by $\mathcal{O}_{\mathsf{rka}}$ as $F_{\mathbf{k}'}(\mathbf{x})$ where
     ▷ $\mathbf{k}' = \mathbf{a} \circ \Delta + \mathbf{b} \in \mathbb{K}^n = \phi(\Delta)$, for $\phi \in \Phi_{\mathsf{aff}}$.
3. **Challenge:** For the single challenge query $\mathbf{x}^*$, subject to $\langle \mathbf{z}, \mathbf{x}^* \rangle \neq 0$ and $\mathbf{x}^* \notin Q$, the challenger does the following. Sample $b \in \{0, 1\}$.
   - If $b = 0$, then
     - Compute $a_i := \mathsf{map}(\langle \mathbf{z}, \mathbf{x} \rangle)$ and $b_i := \mathsf{map}(\mathbf{k_0}_i + \langle \mathbf{z_1}_i, \mathbf{x}^* \rangle)$, for all $i \in [n]$.
     - Set $\phi^* \colon \mathbf{u} \mapsto \mathbf{a} \circ \mathbf{u} + \mathbf{b}$ where $\mathbf{a} := (a_1, \ldots, a_n)$ and $\mathbf{b} := (b_1, \ldots, b_n)$, where $\circ$ denotes the component wise product.
     - Query $\mathcal{O}_{\mathsf{rka}}$ on input $(\phi^*, \mathbf{x}^*)$, and forward the response y^* to $\mathcal{A}$.
   - Else if $b = 1$, then
     - Sample $y^* \xleftarrow{R} \mathcal{Y}$ and send y^* to $\mathcal{A}$.

4. **Post-challenge queries:** Answered identically to pre-challenge queries.

Claim. $\mathcal{A}$'s advantage in $\mathcal{H}_2$ is identical to $\mathcal{A}$'s advantage in $\mathcal{H}_1$.

Proof. The difference between $\mathcal{H}_2$ and $\mathcal{H}_1$ is again purely syntactic since each output is computed identically in both games, with the only difference being that the challenger now only has access to Δ via the oracle $\mathcal{O}_{\mathsf{rka}}$.

Hybrid $\mathcal{H}_3$. This hybrid consists of the RKA security game for $\mathcal{F}$ with respect to affine related key derivation functions Φ_{aff}.

Claim. If there exits an efficient adversary $\mathcal{A}$ for $\mathcal{H}_2$ that wins with non-negligible advantage, then there exists an efficient Φ_{aff}-restricted adversary $\mathcal{B}$ that wins the $\mathcal{H}_3$ game (RKA security game) with the same advantage as $\mathcal{A}$.

Proof. The challenger in $\mathcal{H}_2$ is already playing the role of a Φ_{aff}-restricted adversary when querying the oracle $\mathcal{O}_{\mathsf{rka}}$ to answer the pre- and post-challenge queries. The reduction to RKA security of $\mathcal{F}$ is therefore straightforward.

Constraint Privacy. For constraint privacy, we must show that if $\mathcal{F}$ is an RKA-secure PRF family, then all evaluation and challenge queries remain pseudorandom, regardless of whether constraint $\mathbf{z}$ or $\mathbf{z}'$ is used by the challenger.[5]

Again, note that $\mathbf{z_0}_i + \Delta_i\mathbf{z}$ is distributed identically to $\mathbf{z_0}_i + \Delta_i\mathbf{z}'$, thereby making the constraint key, absent the evaluation queries, efficiently simulatable regardless of the constraint chosen by the challenger. Now, we must show that this remains the case even when the adversary is given access to the evaluation oracles. We prove this via the following lemma. Roughly speaking, the lemma states that if the underlying PRF is RKA-secure, then distinguishing between evaluations under two different related-key derivation functions of the PRF key contradicts the RKA security of the PRF.

Lemma 1. *Let λ be a security parameter and $\mathcal{F} = \{F_k \colon \mathcal{X} \to \mathcal{Y}\}_{k\in\mathcal{K}}$ be an RKA-secure PRF. Then, for all efficient Φ-restricted adversaries $\mathcal{A}$, the advantage in the following game is negligible in λ.*

- **Setup:** On input 1^λ, the challenger samples $k \xleftarrow{\text{R}} \mathcal{K}$, samples a random bit $b \in \{0,1\}$, initializes the set $Q := \emptyset$, and runs $\mathcal{A}(1^\lambda)$.
- **Pre-challenge queries:** For each query (ϕ, x), the challenger computes $y \leftarrow F_{\phi(k)}(x)$, sends y to $\mathcal{A}$, and proceeds to update $Q \leftarrow Q \cup \{(\phi, x)\}$.
- **Challenge query:** $\mathcal{A}$ sends challenge query $(\phi_0^*, \phi_1^*, x^*)$, *subject to the restriction that* $(\phi_c^*, x^*) \notin Q$, $\forall c \in \{0,1\}$. The challenger computes $y^* \leftarrow F_{\phi_b^*(k)}(x^*)$ and sends y^* to $\mathcal{A}$.
- **Post-challenge queries:** For each query (ϕ, x) *subject to the restriction that* $(\phi, x) \neq (\phi_c^*, x^*), \forall c \in \{0,1\}$, the challenger computes $y \leftarrow F_{\phi(k)}(x)$, and sends y to $\mathcal{A}$.
- **Guess:** $\mathcal{A}$ outputs its guess b'.

$\mathcal{A}$ wins if $b' = b$ and its advantage is defined as $|\Pr[\mathcal{A}$ wins$] - \frac{1}{2}|$, where the probability is over the internal coins of $\mathcal{A}$ and choice of k.

The lemma follows immediately from a standard hybrid argument. By RKA-security of the PRF F we have that $F_{\phi_0(k)}(x) \approx_c R(x) \approx_c F_{\phi_1(k)}(x)$, where R is a random function. Therefore, a distinguisher would directly contradict the security of the RKA-PRF.

■

5.2 DDH-based Construction

In this section, we describe the DDH-based RKA-secure PRF construction of Bellare and Cash [7] (later extended by Abdalla et al. [1]) and describe how it fits into Construction 2 to realize a DDH-based CPRF for inner-product predicates.

RKA-secure PRF from DDH. The multiplicative variant [1, 7] of the Naor-Reingold PRF [53] is parameterized by an integer $n \geq 1$ and a multiplicative group $\mathbb{G}$ of prime order p with generator g. The

[5] Recall that the adversary provides two constraints $\mathbf{z}$ and $\mathbf{z}'$.

PRF key $\mathbf{k} = (a_1, \ldots, a_n) \in \mathbb{Z}_p^n$ consists of n random elements in $\mathbb{Z}_p^n$ and the input $x \in \{0,1\}^n \setminus \{0^n\}$ is chosen from the set of all non-zero n-bit strings. The PRF NR^* is then defined as:

$$\mathsf{NR}^*((a_1, \ldots, a_n), x) = g^{\prod_{i=1}^n a_i^{x_i}}. \tag{1}$$

The RKA-secure version of the multiplicative Naor-Reingold PRF is parameterized by a collision-resistant hash function $h\colon \{0,1\}^n \times \mathbb{G}^n \to \{0,1\}^{n-2}$ and is defined as:[6]

$$\mathsf{NR}^*((a_1, \ldots, a_n), 11 \| h(x, g^{a_1}, \ldots, g^{a_n})). \tag{2}$$

Abdalla et al. [1, Section 4] show that Equation (2) is an RKA-secure PRF for Φ_{aff}-restricted adversaries. We provide an informal merger of the main theorems from Abdalla et al. [1] pertaining to this construction here, for completeness.

Proposition 1 (Merge of [1, Theorems 4.5, 5.1, & A1]). *Let $\mathbb{G}$ be a multiplicative group of prime order p and let NR^* be defined as in Equation (1). Let $h\colon \{0,1\}^n \times \mathbb{G}^n \to \{0,1\}^{n-2}$ be a collision-resistant hash function. Define the PRF family $\mathcal{F} = \{F_{\mathbf{k}}\colon \{0,1\}^n \to \mathbb{G}\}_{\mathbf{k} \in \mathbb{Z}_p^n}$ to be as in Equation (2). Then, if the DDH assumption holds in $\mathbb{G}$, $\mathcal{F}$ is RKA-secure against all efficient, Φ_{aff}-restricted adversaries $\mathcal{A}$.*

Remark 5 (RKA security under DDH). Abdalla et al. [1] prove the RKA security of their construction for Φ_{aff}-restricted adversaries under the 1-DDHI assumption (which is known to be equivalent to the Square DDH assumption [9]). However, they explicitly note that, by combining Theorems 4.5, 5.1, & A1 (found in the full version of their paper), they obtain the same result under the DDH assumption.

Remark 6 (Supporting vector inputs). NR^* takes as input a *binary* string $x \in \{0,1\}^n$ as opposed to a vector $\mathbf{x} \in \mathbb{F}^\ell$ as is assumed by our framework. However, we can easily map any $\mathbf{x} \in \mathbb{F}^\ell$ to a binary string of required length via any collision-resistant hash function (CRHF), which are known from the discrete logarithm assumption [33] (implied by DDH, see Appendix C), making vector inputs $\mathbf{x} \in \mathbb{F}^\ell$ syntactically cleaner and without any loss of generality. In particular, for a CRHF h, the binary string input x can be computed as $h(\mathbf{x})$. Moreover, since the RKA-secure variant of NR^* already requires hashing the input using a CRHF, this does not introduce additional computational complexity.

Construction from DDH-based RKA-secure PRF. With the RKA-secure PRF construction of Proposition 1, we can instantiate Construction 2. To satisfy the key space and related-key derivation requirements, we must instantiate our extended framework with the following parameters. Let p be the order of the DDH-hard group $\mathbb{G}$. We set $\mathbb{F}$ to be a field extension of $\mathbb{F}_p$, and let $n = n(\lambda) \in \mathsf{poly}(\lambda)$, following Equation (2). Applying Theorem 2 in conjunction with Proposition 1 yields:

Theorem 3. *Assume that the DDH assumption holds in a group $\mathbb{G}$ of order p. Then, there exists a (1-key, selectively-secure, constraint-hiding) CPRF for inner-product constraint predicates with vectors in $\mathbb{F}_p^\ell$, for any $\ell \geq 1$.*

Remark 7 (Complexity of the DDH-based construction). The Naor-Reingold PRF from Equation (1) can be evaluated in NC^1. Interestingly, the same is true of the RKA-secure variant of Equation (2), provided that the collision resistant hash function can be evaluated in NC^1 (which is the case of the discrete log based construction [33]; see also Appendix C). We will use this later in Appendix B when applying our construction to lower bounds in learning theory.

5.3 VDLPN-based Construction

In this section, we show that we can instantiate Construction 2 from any RKA-secure PRF supporting only additive key derivation functions $\Phi_+ \subset \Phi_{\mathsf{aff}}$ over the field $\mathbb{F}_2$. In particular, this allows us to instantiate our framework using the weak PRF candidate of Boyle et al. [22] based on the Variable-density Learning Parity with Noise (VDLPN) assumption. This yields the first construction of a (weak) CPRF for inner-product predicates under a code-based assumption.

[6] Note that the prefix "11" ensures that the input is never 0^n, and therefore always in the domain of NR^* [1, 7].

RKA-secure weak PRF candidate from VDLPN. For a security parameter λ, the VDLPN-based weak PRF candidate of Boyle et al. [22] is parameterized by integers $D = D(\lambda)$, $w = w(\lambda)$, input space $\{0,1\}^n$ and key space $\{0,1\}^n$, where $n := w \cdot D(D-1)/2$. The PRF F_K is defined as:

$$F_K(x) = \bigoplus_{i=1}^{D} \bigoplus_{j=1}^{w} \bigwedge_{k=1}^{i} (K_{i,j,k} \oplus x_{i,j,k}). \tag{3}$$

Theorem 4 (Informal; adapted from [22, Theorem 6.9]). *Let λ be a security parameter and suppose that the VDLPN assumption holds with parameters $w(\lambda)$ and $D(\lambda)$. Then, the PRF in Equation (3) is an RKA-secure weak PRF with respect to additive key derivation functions Φ_+.*

We will use the following lemma which proves that for the case of $\mathbb{F}_2$, additive and affine RKA security are in fact equivalent in our context:

Lemma 2. *Let F be a PRF with key space $\mathbb{F}_2^n$ that is secure against Φ_+-restricted adversaries. Then, Construction 2 instantiated with F is a secure CPRF.*

Proof. Consider the proof of Theorem 2. We look at the queries issued by the CPRF challenger to the RKA oracle $\mathcal{O}_{\mathsf{rka}}$ in Hybrid $\mathcal{H}_2$ of the proof. For each query $\mathbf{x}$ issued by the adversary to the CPRF challenger, the induced affine function $\phi \in \Phi_{\mathsf{aff}}$ is parameterized by vectors $\mathbf{a}, \mathbf{b} \in \mathbb{F}_2^n$. Note that $\mathbf{a} = (a_1, \ldots, a_n)$, where $a_i \leftarrow \langle \mathbf{z}, \mathbf{x} \rangle$. Moreover, $a_i \neq 0$ for all queries that do not satisfy the constraint, which implies that $a_i = 1 \in \mathbb{F}_2$. As such, each (constrained) query issued to the RKA oracle $\mathcal{O}_{\mathsf{rka}}$ by the challenger is an affine function $\phi \in \Phi_{\mathsf{aff}}$ parameterized by $(\mathbf{1}, \mathbf{b})$ and the oracle $\mathcal{O}_{\mathsf{rka}}$ responds with the PRF evaluated using key $\mathbf{k} := \mathbf{1} \circ \Delta + \mathbf{b}$. This is equivalent to an *additive* function $\phi' \in \Phi_+$ simply parameterized by $\mathbf{b}$. The reduction in Theorem 2 therefore goes through as before, concluding the lemma. ∎

Construction from VDLPN-based RKA-secure weak PRF. With the RKA-secure weak PRF construction of Equation (3), we can instantiate Construction 2. To satisfy the key space and related-key derivation requirements, we must instantiate our extended framework with the following parameters. We set $\mathbb{F}$ to be a field extension of $\mathbb{F}_{2^n}$, $n = n(\lambda) \in \mathsf{poly}(\lambda)$, map maps from $\mathbb{F}$ to $\mathbb{F}_2^n$, and $\ell \geq n$ (inputs of length ℓ can be truncated to n before being fed into the PRF, without loss of generality). Applying Theorem 2 in conjunction with Theorem 4 and Lemma 2 yields:

Theorem 5. *Assume that the VDLPN assumption holds. Then, there exists a (1-key, selectively-secure, constraint-hiding) weak CPRF for inner-product constraint predicates computed over vectors in $\mathbb{F}_2^\ell$, where $\ell \geq n$.*

6 CPRFs for Inner-Product Predicates from OWFs

In this section, we instantiate our extended framework from Section 5.1 under the minimal assumption that one-way functions exist. Unlike our constructions in Section 5.1, here we will require that the set of possible related keys computed for evaluation queries is bounded by a fixed polynomial $t = t(\lambda)$, which forces us to restrict the input domain of the CPRF. Specifically, we show that we can satisfy this requirement *without* placing any restrictions on the CPRF adversary if the CPRF inputs are vectors in $[0, B)^\ell$ with $B \in O(1)$ and $\ell = \ell(\lambda) \in O(\log \lambda)$. These restrictions limit the L_∞-norm of each input vector and make the input domain of the CPRF polynomial in the security parameter. We note that this is the same class of inner-product constraints considered by Davidson et al. [34] (inner products over $\mathbb{Z}$) from the LWE assumption, albeit here we only obtain a polynomially-sized input domain.

Our construction builds off of a result by Applebaum and Widder [2], which constructs a restricted class of RKA-secure PRFs from any PRF and a m-wise independent hash function. Their construction is secure against *additive* relations over a group, provided that the RKA adversary uses at most $t = t(\lambda)$ different related-key derivation functions $\phi_1 \ldots, \phi_t \in \Phi_+$, where $t \ll m$. (We stress, however, that the adversary can query the PRF on an unbounded number of inputs using each of the t different RKA functions.) Because m-wise independent hash functions can be constructed unconditionally [62], the resulting RKA-secure PRF can be realized from any PRF, thus relying only on the assumption that one-way functions exist [2, 41]. More formally, they prove:

Theorem 6 (Adapted from [2]). *Let $\mathcal{K} = \{\mathbb{G}_\lambda\}_{\lambda \in \mathbb{N}}$ be a sequence of efficiently computable additive groups, and $t = t(\lambda)$ be an arbitrary fixed polynomial. Then, assuming the existence of a PRF $\mathcal{F} = \{F_k \colon \mathcal{X}_\lambda \to \mathcal{Y}\}_{k \in \mathbb{G}_\lambda}$, there exists an RKA-secure PRF with respect to addition over $\mathcal{K}$ provided that the total number of unique related-key derivation functions queried by the adversary is bounded by t. (The adversary is allowed to query each function on any number of inputs.)*

Unfortunately, we require the PRF to be RKA-secure with respect to *affine* relations Φ_{aff} and therefore cannot apply Theorem 6 directly. More concretely, the issue with affine (as opposed to additive) relations is that they are not "claw-free," meaning that there exist pairs of different functions $\phi_1, \phi_2 \in \Phi_{\mathsf{aff}}$ such that for a key $k \in \mathcal{K}$, $\phi_1(k) = \phi_2(k)$. The lack of claw-freeness poses problems in security proofs because, if an adversary is able to find two different $\phi_1, \phi_2 \in \Phi_{\mathsf{aff}}$ such that $\phi_1(k) = \phi_2(k)$, the adversary learns information about k and can then break the RKA-security of the PRF [1]. To address this, we strengthen Theorem 6 for the case of Φ_{aff}-restricted adversaries by showing that the number of collisions is bounded by a negligible factor in the security parameter, proving a stronger theorem via their approach. We describe this next.

6.1 Affine RKA-secure PRFs from OWFs

In this section, we show how to construct RKA-secure PRFs for affine related-key derviation functions from one-way functions. The framework and proof closely follows that of Applebaum and Widder [2] for constructing RKA-secure PRFs from m-wise independent hash functions.

Immunizing PRFs against RKA. The idea of Applebaum and Widder [2] is to to immunize any regular PRF family $\mathcal{F}$ with key space $\mathcal{K} = \mathcal{K}_\lambda$ against a *bounded* related-key attack, where the adversary makes at most t related key queries (but can make an unbounded number of PRF queries under each related key) for some apriori fixed $t = t(\lambda) \in \mathsf{poly}(\lambda)$. The high level idea is to use a long key s from a large key space $\mathcal{S}$ (larger than $\mathcal{K}^t$) and use a public hash function h to derive shorter key $h(s) \in \mathcal{K}$ for $\mathcal{F}$. Here, we generalize their approach to the case of affine functions.

Definition 7 (t-good hash function). *Let λ be a security parameter, $\mathbb{F}$ be finite field of order at least 2^λ, and $\mathcal{K} \subseteq \{0,1\}^\lambda$ be a set of strings. A hash function $h \colon \mathbb{F} \to \mathcal{K}$ is said to be t-good if for any t-tuple of distinct affine function $(\phi_1, \ldots, \phi_t) \in \Phi_{\mathsf{aff}}^t$, the joint distribution of $(h(s), h(\phi_1(s)), \ldots, h(\phi_t(s))$ induced by a random choice of $s \xleftarrow{R} \mathbb{F}$, is ε-close in statistical distance to the uniform distribution over $\mathcal{K}^{t+1}$, for some negligible $\varepsilon = \varepsilon(\lambda)$.*

Definition 8 (t-good hash family). *Let λ be a security parameter, $\mathbb{F}$ be a finite field of order at least 2^λ, and $\mathcal{Z}, \mathcal{K} \subseteq \{0,1\}^\lambda$. A family of hash functions $H = \{h_z \colon \mathbb{F} \to \mathcal{K}\}_{z \in \mathcal{Z}}$ is said to be t-good if with all-but-negligible probability, for a randomly selected $z \xleftarrow{R} \mathcal{Z}$, the hash function h_z is t-good.*

We now prove that if we have a t-good hash family, we can "immunize" any PRF against affine related key attacks. Later, in Lemma 3, we show how to construct a t-good hash family from m-wise independent hash functions.

Theorem 7 (Extended from [2, Lemma 7.1]). *Let λ be a security parameter, $t = t(\lambda) \in \mathsf{poly}(\lambda)$, $\mathbb{F}$ be a finite field of order at least 2^λ, and $\mathcal{Z}, \mathcal{K} \subseteq \{0,1\}^\lambda$. Let $\mathcal{F} = \{F_k \colon \mathcal{X} \to \mathcal{Y}\}_{k \in \mathcal{K}}$ be a PRF family and $H = \{h_z \colon \mathbb{F} \to \mathcal{K}\}_{z \in \mathcal{Z}}$ be a t-good hash family. The PRF family $\mathcal{G} = \{G_{s,z} \colon \mathcal{X} \to \mathcal{Y}\}_{s \in \mathbb{F}, z \in \mathcal{Z}}$, parameterized by a secret $s \xleftarrow{R} \mathbb{F}$ and public $z \xleftarrow{R} \mathcal{Z}$, and defined by the mapping $G_{s,z}(x) \mapsto F_k(x)$, where $k \leftarrow h_z(s)$, is an RKA-secure PRF family against t-bounded Φ_{aff}-restricted adversaries.*

Proof. Suppose, towards contradiction, there exists an efficient Φ_{aff}-restricted $\mathcal{A}$ that has non-negligible advantage in the RKA-security game for G. Then, there exists a non-negligible function ν such that,

$$\left| \Pr_{s \xleftarrow{R} \mathbb{F}, z \xleftarrow{R} \mathcal{Z}} [\mathcal{A}^{G_{s,z}}(1^\lambda, z)] - \Pr_{z \xleftarrow{R} \mathcal{Z}} [\mathcal{A}^{R}(1^\lambda, z)] \right| \geq \nu(\lambda),$$

where R is a truly random function.

Then, consider a vector of $t+1$ keys $\mathbf{k} := (k_0, k_1, \ldots, k_t) \in \mathcal{K}^{t+1}$, and define a stateful oracle $\mathcal{O}_{\mathbf{k}}$ as follows.

Oracle $\mathcal{O}_{\mathbf{k}}$

Initialize. Set $Q_\phi := \{\}$, define a dictionary $T := [\ ]$, and counter $j \leftarrow 1$.

Evaluation.

- For each non-RKA query x, output $F_{k_0}(x)$.
- For each RKA query (ϕ, x):
  - If $\phi \in Q_\phi$, retrieve $k_i \leftarrow T[\phi]$ and output $F_{k_i}(x)$.
  - If $\phi \notin Q_\phi$, set $T[\phi] \leftarrow k_j$, set $j \leftarrow j+1$, and output $F_{k_j}(x)$.

In words, $\mathcal{O}_{\mathbf{k}}$ outputs $F_{k_i}(x)$, and stores the association between ϕ and k_i to answer all future queries involving ϕ using PRF key k_i.

Now, because h_z is t-good, for a random vector $\mathbf{k}$ of $t+1$ keys, we have that

$$\left| \Pr_{\mathbf{k} \xleftarrow{R} \mathcal{K}^{t+1}, z \xleftarrow{R} \mathcal{Z}} [\mathcal{A}^{\mathcal{O}_{\mathbf{k}}}(1^\lambda, z)] - \Pr_{z \xleftarrow{R} \mathcal{Z}} [\mathcal{A}^{G_{s,z}}(1^\lambda, z)] \right| \geq \nu(\lambda) - \mathsf{negl}(\lambda).$$

By a straightforward hybrid argument, it follows that $\mathcal{A}$ has non-negligible advantage in winning the (standard) PRF game by distinguishing between $\mathcal{O}_{\mathbf{k}}$ and the truly random function R, contradicting that $\mathcal{F}$ is a PRF. This proves security against Φ_{aff}-restricted adversaries. ■

The following lemma shows that any $\Omega(\lambda \cdot t^2)$-wise independent hash function with a sufficiently large domain is t-good in the sense of Definition 7. Moreover, an m-wise independent hash function can be constructed unconditionally for any m (e.g., using a universal hash based on random polynomials [62]).

Lemma 3. *Let λ be a security parameter, $t = t(\lambda) \in \mathsf{poly}(\lambda)$, and H be a family of m-wise independent hash function with domain $S = \{S_\lambda\}$ and range $K = \{K_\lambda\}$ where $m \geq \lambda(3t+5)(t+1)$, $|K_\lambda| = 2^\lambda$, and $|S_\lambda| = 2^{\lambda(2t+6)}$. Then, H is a t-good family of hash function. In particular, for all but a $2^{-\lambda}$ fraction of the functions in H, the distribution of $h_z \xleftarrow{R} H$ is $2^{-0.99\lambda}$-close to uniform.*

Proof. The proof is deferred to Appendix D.1 as it closely follows the proof strategy of Applebaum and Widder [2, Proof of Lemma 7.2] for a similar lemma in the context of additive functions. ■

6.2 CPRF Construction from OWFs

Using the RKA-secure PRF construction from Theorem 7, we can instantiate Construction 2 with $\mathbb{F} = \mathbb{F}_p$, for sufficiently large $p \geq 2^{\lambda(2t+6)}$ as required by Lemma 3, and $n \geq 1$. However, we must set the input vector domain to $[0, B)^\ell \subset \mathbb{Z}^\ell$ with the vector length ℓ such that $B^\ell \leq t$. Specifically, this ensures that the total number of unique inputs to the t-good hash when deriving affine keys is bounded by $t = t(\lambda) \in \mathsf{poly}(\lambda)$. To see this, note that there are B^ℓ possible values for the inner product $\langle \mathbf{z_0}, \mathbf{x} \rangle + \Delta \langle \mathbf{z}, \mathbf{x} \rangle$ given that $\mathbf{z}$ and $\mathbf{z_0}$ are fixed while $\mathbf{x} \in [0, B)^\ell$ is chosen by the adversary. Hence, we can simply let map be defined by applying n different t-good hash functions component-wise to derive the PRF key in K^n. Then, applying Theorem 2 in conjunction with Theorem 7 yields:

Theorem 8. *Let λ be a security parameter and fix a polynomial $t = t(\lambda) \in \mathsf{poly}(\lambda)$. Assume that one-way functions exist. Then, there exists a (1-key, selectively-secure, constraint-hiding) CPRF for inner-product constraint predicates with $\ell = \ell(\lambda) \in O(\log \lambda)$ and input vectors in the range $[0, B)$ for any constant B such that $B^\ell \leq t$.*

Proof. We recall the proof of Theorem 2, and in particular Hybrid $\mathcal{H}_2$. In the game defined by $\mathcal{H}_2$, for each query $\mathbf{x}$ issued by the CPRF adversary, the challenger derives the affine function ϕ parameterized by vectors $\mathbf{a}, \mathbf{b} \in \mathbb{F}_p^n$ where:

- $\mathbf{a} := (a_1, \ldots, a_n)$ with $a_i = \langle \mathbf{z}, \mathbf{x} \rangle$ for all $i \in [n]$.

- $\mathbf{b} := (b_1, \dots, b_n)$ with $b_i = \langle \mathbf{z_0}_i, \mathbf{x} \rangle$ for all $i \in [n]$.

Note that $\mathbf{z}$ and $\mathbf{z_0}_i$, for all $i \in [n]$ are fixed at the start of the CPRF game. Therefore, $\mathbf{a}, \mathbf{b}$ are both entirely determined by the query vector $\mathbf{x}$. The RKA oracle $\mathcal{O}_{\mathsf{rka}}$ in $\mathcal{H}_2$ (when instantiated with the immunized RKA-PRF construction of Theorem 7) computes the RKA key as $h_i(a_i \Delta_i + b_i)$ for all $i \in [n]$, where h_i is an independent t-good hash function and Δ_i is an independent PRF key. We must show that, for all possible sets of queries $Q := \{\mathbf{x}_j \mid 1 \leq j \leq q_E\}$ issued by $\mathcal{A}$ (here q_E is an arbitrary upper bound on the total number of evaluation queries), the number of unique inputs to h_i never exceeds t. This follows from the fact that the number of possible values that $k_i := a_i \Delta_i + b_i$ can take on is bounded by the number of unique values of $\mathbf{x}$, which in turn is bounded by $B^\ell \leq t$, by construction. We remark that there are no restrictions placed on the adversary's queries—the adversary can adaptively query the CPRF challenger and issue any polynomial number of evaluation queries (independently of t). ∎

As a corollary, we obtain an analogous result to Theorem 8 but with an exponential input domain provided that the CPRF adversary makes at most t unique evaluation queries on constrained inputs.

Corollary 1. *Let λ be a security parameter and fix a polynomial $t = t(\lambda) \in \mathsf{poly}(\lambda)$. Assume that one-way functions exist. Then, there exists a (1-key, selectively-secure, constraint-hiding) CPRF for inner-product constraint predicates for any $\ell \geq 1$ provided that the adversary makes at most t constrained evaluation queries.*

7 Evaluation

In this section, we implement[7] and benchmark our CPRF constructions. For each construction, we first analyze the complexity (in terms of multiplication, additions, and invocations of other cryptographic primitives) and then report the concrete performance of our Go (v1.20) implementation benchmarked on an Apple M1 CPU. All benchmarks are performed on a single core.

7.1 Complexity and Benchmarks

Random oracle construction. The random oracle construction requires computing the inner product in $\mathbb{F}$ followed by a call to a random oracle. We heuristically instantiate the random oracle using the SHA256 hash function. We let the $\mathbb{F} = \mathbb{F}_p$ be a finite field where p is a 128-bit prime. The bottleneck of the construction is computing the inner product (modulo p), which requires a total of ℓ modular multiplications and additions. We report the concrete performance in Table 2. Overall, evaluation required a few microseconds of computation time, ranging from 2μs for small vectors ($\ell = 10$) and 200μs for large vectors ($\ell = 1000$).

DDH-based construction. In the DDH-based construction, the bulk of the required operations are performed modulo p, where p is the order of the DDH-hard group. For a security parameter λ and $n = n(\lambda)$, the CPRF construction requires computing (1) $n\ell$ multiplications and $n\ell$ additions (mod p) to compute the inner products between length-ℓ vectors, (2) one invocation of a collision-resistant hash function, and (3) n multiplications (mod p) and $n + 1$ group operations in $\mathbb{G}$ to compute the PRF evaluation. This results in a total complexity of $n(\ell + 1)$ multiplications (mod p), $n\ell$ additions, $n + 1$ group operations, and one invocation of a CRHF. Using the P256 elliptic curve, letting $n = 128$, and using the discrete logarithm based CRHF construction (see Appendix C), each CPRF evaluation requires a few ms to compute (note that in practice, the DL-based CRHF can be replaced with a fixed-key AES or SHA256 hash function for better performance). We report the concrete performance in Table 3. The concrete performance is worse for smaller vectors due to constant overheads of computing the CRHF and PRF relative to computing the inner product. For larger vectors, however, the inner product computation dominates the cost.

OWF-based construction. Our OWF-based construction requires computing the inner products over the integers, which requires ℓ multiplications and ℓ additions in $\mathbb{Z}$ to compute inner products. Then, we need to evaluate an m-wise independent hash function. This requires evaluating a random polynomial of degree $m = \lambda(3t+5)(t+1)$ with $\log_2(\lambda(2t+6))$-bit coefficients (recall Lemma 3). Here,

[7] The implementation is open source: `https://github.com/sachaservan/cprf`.

(ℓ)	10	50	100	500	1000
	2 μs	10 μs	19 μs	98 μs	200 μs

Table 2: Concrete evaluation time for our RO-based CPRF construction for vectors of length ℓ.

(ℓ)	10	50	100	500	1000
	8 ms	11 ms	16 ms	46 ms	85 ms

Table 3: Concrete evaluation time for our DDH-based CPRF construction for vectors of length ℓ.

we let $\lambda = 40$ as it is a statistical security parameter of the t-good hash function. For very small values of B and ℓ, we obtain reasonable concrete efficiency when evaluating the m-wise independent hash function (less than one second of computation for $B = 2$ and $\ell = 5$ and roughy 50MB public parameters). However, for larger parameters, the concrete efficiency quickly becomes impractical. This blowup is due to the quadratic overhead of Lemma 3. Additionally, the public parameters quickly become impractically large (e.g., petabytes) as ℓ increases due to the cubic factor in t. Furthermore, the concrete size of the public parameters required to store the description of the m-wise independent hash function (m coefficients of a random polynomial) is exceedingly large. This description already reaches terabytes in size with $B = 2$ and $\ell = 10$, barring any concretely practical instantiation.

7.2 Comparison to other CPRF constructions

Prior CPRF constructions for inner product (and NC^1) predicates [4, 30, 34] do not have implementations, and due to large parameters or heavy building blocks, are far too inefficient to be implemented. We briefly discuss the concrete efficiency roadblocks associated with these constructions.

- The LWE-based CPRF construction of Davidson et al. [34] is implementable but very inefficient due to the large parameters required for security and computationally expensive building blocks. Specifically, their construction requires computing a linear (in the input size) matrix-matrix products of LWE matrices, which poses a major efficiency roadblock. Similar roadblocks are faced with other LWE-based constructions, even if adapted to the simpler case of inner-product constraints.
- The constructions of Attrapadung et al. [3] is tailored to evaluating NC^1 boolean circuits and requires computing a linear number of group exponentiations in the degree of the universal NC^1 circuit computing the constraint predicate. While their construction can be theoretically applied to computing inner-product predicates, it does lend itself to a practical solution as it would require emulating field operations *inside* of the NC^1 universal circuit.
- The approach of Couteau et al. [30] based on DCR requires evaluating a PRF using HSS (where the PRF key is encoded as an HSS input share). This requires evaluating a linear (in the degree of the polynomial computing the PRF) number of HSS multiplications. Using a DCR-based variant of the Naor-Reingold PRF (the only DCR-based PRF in NC^1, as required for HSS evaluation) necessitates computing $g^{\prod_i^n a_i^{x_i}}$ in HSS, where the key $k = (a_1, \ldots, a_n)$ is the PRF key provided as input. The exceedingly high degree of this polynomial eliminates the possibility of a concretely practical instantiation, since even low-degree polynomials can already be concretely expensive to evaluate in HSS schemes [19].

7.3 Discussion

In light of the concrete performance of our constructions, is becomes clear that the OWF-based constructions is primarily of theoretical interest on realistic parameters, as it does not scale well with the length of the input vectors. In contrast, the random oracle and DDH-based constructions are both very efficient and require only a few microseconds or milliseconds to evaluate on long input vectors. To the best of our knowledge, these are the first *concretely efficient* constrained PRFs for inner-product predicates.

8 Conclusion and Future Work

In conclusion, this paper contributes a simple framework for constructing constraint-hiding CPRFs with inner-product constraint predicates through subtractive secret sharing and related-key-attack-secure PRFs. Through our framework, we constructed the first (1-key, selectively-secure, constraint-hiding) CPRFs with inner-product constraint predicates from DDH and from one-way functions, and the first (1-key, adaptively-secure, constraint-hiding) CPRFs in the random oracle model.

Future work. We identify several interesting avenues for future work. The first open problem is constructing (constraint-hiding) CPRFs for more expressive constraints from new assumptions, especially for NC^1 and puncturing constraints. Given the tight connection between our framework and RKA-secure PRFs, an additional avenue of exploration is constructing suitable RKA-secure PRFs from new assumptions (which will immediately enable instantiating our framework under those assumptions as well). Second, there are currently few practical applications of CPRFs with inner-product predicates that we are aware of, which we believe is due to the previous lack of concretely efficient constructions. Finding practical other use cases for CPRFs with inner-product predicates (constraint-hiding or not), is an interesting question and worth exploring in light of our efficient constructions.

Acknowledgements

I'd like to thank Geoffroy Couteau for insightful discussions, providing me with several pointers and references (especially when it came to pointing out the relevance of [23, 29]), and many invaluable suggestions. I'd also like to thank Michele Orrù for editorial advice and feedback. Finally, I'm grateful to Vinod Vaikuntanathan and Yael Kalai for helpful discussion on early ideas surrounding this work.

Bibliography

[1] Michel Abdalla, Fabrice Benhamouda, Alain Passelègue, and Kenneth G Paterson. Related-key security for pseudorandom functions beyond the linear barrier. In *Advances in Cryptology–CRYPTO 2014: 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part I 34*, pages 77–94. Springer, 2014.

[2] Benny Applebaum and Eyal Widder. Related-key secure pseudorandom functions: The case of additive attacks. *Cryptology ePrint Archive*, 2014.

[3] Nuttapong Attrapadung, Takahiro Matsuda, Ryo Nishimaki, Shota Yamada, and Takashi Yamakawa. Constrained PRFs for in traditional groups. In *Advances in Cryptology–CRYPTO 2018: 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19–23, 2018, Proceedings, Part II*, pages 543–574. Springer, 2018.

[4] Nuttapong Attrapadung, Takahiro Matsuda, Ryo Nishimaki, Shota Yamada, and Takashi Yamakawa. Adaptively single-key secure constrained PRFs for NC^1. In *IACR International Workshop on Public Key Cryptography*, pages 223–253. Springer, 2019.

[5] Carsten Baum, Lennart Braun, Alexander Munch-Hansen, and Peter Scholl. Moz$\mathbb{Z}_{2^k}$arella: efficient vector-OLE and zero-knowledge proofs over $\mathbb{Z}_{2^k}$. In *Annual International Cryptology Conference*, pages 329–358. Springer, 2022.

[6] Carsten Baum, Lennart Braun, Cyprien Delpech de Saint Guilhem, Michael Klooß, Emmanuela Orsini, Lawrence Roy, and Peter Scholl. Publicly verifiable zero-knowledge and post-quantum signatures from VOLE-in-the-head. In *Annual International Cryptology Conference*, pages 581–615. Springer, 2023.

[7] Mihir Bellare and David Cash. Pseudorandom functions and permutations provably secure against related-key attacks. In *Annual Cryptology Conference*, pages 666–684. Springer, 2010.

[8] Mihir Bellare and Tadayoshi Kohno. A theoretical treatment of related-key attacks: RKA-PRPs, RKA-PRFs, and applications. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 491–506. Springer, 2003.

[9] Olivier Blazy and David Pointcheval. Traceable signature with stepping capabilities. In *Cryptography and Security: From Theory to Applications: Essays Dedicated to Jean-Jacques Quisquater on the Occasion of His 65th Birthday*, pages 108–131. Springer, 2012.

[10] Dan Boneh and Brent Waters. Constrained pseudorandom functions and their applications. In *Advances in Cryptology-ASIACRYPT 2013: 19th International Conference on the Theory and Application of Cryptology and Information Security, Bengaluru, India, December 1-5, 2013, Proceedings, Part II 19*, pages 280–300. Springer, 2013.

[11] Dan Boneh and Mark Zhandry. Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation. *Algorithmica*, 79:1233–1285, 2017.

[12] Dan Boneh, Kevin Lewi, Hart Montgomery, and Ananth Raghunathan. Key homomorphic PRFs and their applications. In *Annual Cryptology Conference*, pages 410–428. Springer, 2013.

[13] Dan Boneh, Sam Kim, and Hart Montgomery. Private puncturable PRFs from standard lattice assumptions. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 415–445. Springer, 2017.

[14] Dan Boneh, Kevin Lewi, and David J Wu. Constraining pseudorandom functions privately. In *IACR International Workshop on Public Key Cryptography*, pages 494–524. Springer, 2017.

[15] Raphaël Bost, Brice Minaud, and Olga Ohrimenko. Forward and backward private searchable encryption from constrained cryptographic primitives. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1465–1482, 2017.

[16] Elette Boyle, Shafi Goldwasser, and Ioana Ivan. Functional signatures and pseudorandom functions. In *International workshop on public key cryptography*, pages 501–519. Springer, 2014.

[17] Elette Boyle, Niv Gilboa, and Yuval Ishai. Function secret sharing. In *Annual international conference on the theory and applications of cryptographic techniques*, pages 337–367. Springer, 2015.

[18] Elette Boyle, Niv Gilboa, and Yuval Ishai. Breaking the circuit size barrier for secure computation under DDH. In *Advances in Cryptology–CRYPTO 2016: 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part I*, pages 509–539. Springer, 2016.

[19] Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, and Michele Orrù. Homomorphic secret sharing: optimizations and applications. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 2105–2122, 2017.

[20] Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, and Peter Scholl. Efficient pseudorandom correlation generators: Silent OT extension and more. In *Advances in Cryptology–CRYPTO 2019: 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18–22, 2019, Proceedings, Part III 39*, pages 489–518. Springer, 2019.

[21] Elette Boyle, Lisa Kohl, and Peter Scholl. Homomorphic secret sharing from lattices without FHE. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 3–33. Springer, 2019.

[22] Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, and Peter Scholl. Correlated pseudorandom functions from variable-density LPN. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1069–1080. IEEE, 2020.

[23] Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, and Peter Scholl. Efficient pseudorandom correlation generators from ring-LPN. In *Advances in Cryptology–CRYPTO 2020: 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August 17–21, 2020, Proceedings, Part II 40*, pages 387–416. Springer, 2020.

[24] Zvika Brakerski and Vinod Vaikuntanathan. Constrained key-homomorphic PRFs from standard lattice assumptions: Or: How to secretly embed a circuit in your PRF. In *Theory of Cryptography: 12th Theory of Cryptography Conference, TCC 2015, Warsaw, Poland, March 23-25, 2015, Proceedings, Part II 12*, pages 1–30. Springer, 2015.

[25] Zvika Brakerski, Rotem Tsabary, Vinod Vaikuntanathan, and Hoeteck Wee. Private constrained PRFs (and more) from LWE. In *Theory of Cryptography Conference*, pages 264–302. Springer, 2017.

[26] Ran Canetti and Yilei Chen. Constraint-hiding constrained PRFs for NC from LWE. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 446–476. Springer, 2017.

[27] Nishanth Chandran, Srinivasan Raghuraman, and Dhinakaran Vinayagamurthy. Reducing depth in constrained PRFs: From bit-fixing to NC^1. In *Public-Key Cryptography–PKC 2016*, pages 359–385. Springer, 2016.

[28] Yilei Chen, Vinod Vaikuntanathan, and Hoeteck Wee. GGH15 beyond permutation branching programs: proofs, attacks, and candidates. In *Advances in Cryptology–CRYPTO 2018: 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19–23, 2018, Proceedings, Part II 38*, pages 577–607. Springer, 2018.

[29] Aloni Cohen, Shafi Goldwasser, and Vinod Vaikuntanathan. Aggregate pseudorandom functions and connections to learning. In *Theory of Cryptography: 12th Theory of Cryptography Conference, TCC 2015, Warsaw, Poland, March 23-25, 2015, Proceedings, Part II 12*, pages 61–89. Springer, 2015.

[30] Geoffroy Couteau, Pierre Meyer, Alain Passelègue, and Mahshid Riahinia. Constrained pseudorandom functions from homomorphic secret sharing. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 194–224. Springer, 2023.

[31] Geoffroy Couteau, Lalita Devadas, Srinivas Devadas, Alexander Koch, and Sacha Servan-Schreiber. Quietot: Lightweight oblivious transfer with a public-key setup. *Cryptology ePrint Archive*, 2024.

[32] Nan Cui, Shengli Liu, Yunhua Wen, and Dawu Gu. Pseudorandom functions from LWE: RKA security and application. In *Australasian Conference on Information Security and Privacy*, pages 229–250. Springer, 2019.

[33] Ivan Bjerre Damgård. Collision free hash functions and public key signature schemes. In *Workshop on the Theory and Application of of Cryptographic Techniques*, pages 203–216. Springer, 1987.

[34] Alex Davidson, Shuichi Katsumata, Ryo Nishimaki, Shota Yamada, and Takashi Yamakawa. Adaptively secure constrained pseudorandom functions in the standard model. In *Advances in Cryptology–CRYPTO 2020: 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August 17–21, 2020, Proceedings, Part I*, pages 559–589. Springer, 2020.

[35] Nico Döttling, Sanjam Garg, Yuval Ishai, Giulio Malavolta, Tamer Mour, and Rafail Ostrovsky. Trapdoor hash functions and their applications. In *Annual International Cryptology Conference*, pages 3–32. Springer, 2019.

[36] Zeev Dvir, Anup Rao, Avi Wigderson, and Amir Yehudayoff. Restriction access. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, pages 19–33, 2012.

[37] Thibauld Feneuil. *Post-Quantum Signatures from Secure Multiparty Computation*. PhD thesis, Sorbonne Université, 2023.

[38] Sanjam Garg, Craig Gentry, Shai Halevi, and Mark Zhandry. Fully secure functional encryption without obfuscation. *IACR Cryptol. ePrint Arch.*, 2014:666, 2014.

[39] Niv Gilboa and Yuval Ishai. Distributed point functions and their applications. In *Advances in Cryptology–EUROCRYPT 2014: 33rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Copenhagen, Denmark, May 11-15, 2014. Proceedings 33*, pages 640–658. Springer, 2014.

[40] David Goldenberg and Moses Liskov. On related-secret pseudorandomness. In *Theory of Cryptography Conference*, pages 255–272. Springer, 2010.

[41] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of the ACM (JACM)*, 33(4):792–807, 1986.

[42] Vipul Goyal, Adam O'Neill, and Vanishree Rao. Correlated-input secure hash functions. In *Theory of Cryptography: 8th Theory of Cryptography Conference, TCC 2011, Providence, RI, USA, March 28-30, 2011. Proceedings 8*, pages 182–200. Springer, 2011.

[43] Ronen Gradwohl and Amir Yehudayoff. t-wise independence with local dependencies. *Information processing letters*, 106(5):208–212, 2008.

[44] David Heath and Vladimir Kolesnikov. One hot garbling. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 574–593, 2021.

[45] Dennis Hofheinz, Akshay Kamath, Venkata Koppula, and Brent Waters. Adaptively secure constrained pseudorandom functions. In *International Conference on Financial Cryptography and Data Security*, pages 357–376. Springer, 2019.

[46] Susan Hohenberger, Venkata Koppula, and Brent Waters. Adaptively secure puncturable pseudorandom functions in the standard model. In *International conference on the theory and application of cryptology and information security*, pages 79–102. Springer, 2015.

[47] Yuval Ishai, Joe Kilian, Kobbi Nissim, and Erez Petrank. Extending oblivious transfers efficiently. In *Annual International Cryptology Conference*, pages 145–161. Springer, 2003.

[48] Aggelos Kiayias, Stavros Papadopoulos, Nikos Triandopoulos, and Thomas Zacharias. Delegatable pseudorandom functions and applications. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, pages 669–684, 2013.

[49] Vladimir Kolesnikov and Thomas Schneider. Improved garbled circuit: Free XOR gates and applications. In *Automata, Languages and Programming: 35th International Colloquium, ICALP 2008, Reykjavik, Iceland, July 7-11, 2008, Proceedings, Part II 35*, pages 486–498. Springer, 2008.

[50] Arthur Lazzaretti and Charalampos Papamanthou. Treepir: Sublinear-time and polylog-bandwidth private information retrieval from DDH. *Cryptology ePrint Archive*, 2023.

[51] Kevin Lewi, Hart Montgomery, and Ananth Raghunathan. Improved constructions of PRFs secure against related-key attacks. In *Applied Cryptography and Network Security: 12th International Conference, ACNS 2014, Lausanne, Switzerland, June 10-13, 2014. Proceedings 12*, pages 44–61. Springer, 2014.

[52] Yiping Ma, Ke Zhong, Tal Rabin, and Sebastian Angel. Incremental offline/online PIR. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 1741–1758, 2022.

[53] Moni Naor and Omer Reingold. Number-theoretic constructions of efficient pseudo-random functions. *Journal of the ACM (JACM)*, 51(2):231–262, 2004.

[54] Claudio Orlandi, Peter Scholl, and Sophia Yakoubov. The rise of Paillier: homomorphic secret sharing and public-key silent OT. In *Advances in Cryptology–EUROCRYPT 2021: 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17–21, 2021, Proceedings, Part I 40*, pages 678–708. Springer, 2021.

[55] Chris Peikert and Sina Shiehian. Privately constraining and programming PRFs, the LWE way. In *IACR International Workshop on Public Key Cryptography*, pages 675–701. Springer, 2018.

[56] Naty Peter, Rotem Tsabary, and Hoeteck Wee. One-one constrained pseudorandom functions. In *1st Conference on Information-Theoretic Cryptography (ITC 2020)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020.

[57] Benny Pinkas, Thomas Schneider, and Michael Zohner. Scalable private set intersection based on OT extension. *ACM Transactions on Privacy and Security (TOPS)*, 21(2):1–35, 2018.

[58] Kim Ramchen and Brent Waters. Fully secure and fast signing from obfuscation. In *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*, pages 659–673, 2014.

[59] Phillipp Schoppmann, Adrià Gascón, Leonie Reichert, and Mariana Raykova. Distributed vector-OLE: Improved constructions and implementation. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 1055–1072, 2019.

[60] Shi-Feng Sun, Xingliang Yuan, Joseph K Liu, Ron Steinfeld, Amin Sakzad, Viet Vo, and Surya Nepal. Practical backward-secure searchable encryption from symmetric puncturable encryption. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pages 763–780, 2018.

[61] Leslie G Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

[62] Mark N Wegman and J Lawrence Carter. New hash functions and their use in authentication and set equality. *Journal of computer and system sciences*, 22(3):265–279, 1981.

Supplementary Material

A Extensions

In this section, we describe extensions to CPRFs with inner-product constraints.

A.1 More General Constraint Predicates

It is known (in some cases folklore) that CPRFs for inner-product constraint predicates yield CPRFs with constraints described by constant-degree polynomials, t-CNF formulas (with constant t) [34], and the "AND" of an arbitrary set of constraint predicates. We explicitly describe these extensions here for completeness. We note that all the presented extensions preserve the constraint-hiding property.

CPRFs for constant-degree polynomials. A CPRF for inner-product constraint predicates can be converted to a CPRF for constraint predicates described by constant-degree polynomials P by associating each entry in the constraint vector $\mathbf{z}$ with a coefficient of P. Specifically, let $\mathbf{z} = (a_d, a_{d-1}, \ldots, a_1, a_0)$ be the coefficients describing the degree-d polynomial over $\mathbb{F}$. Then, for input vectors of the form $\mathbf{x} = (x^d, x^{d-1}, \ldots, x, 1)$, it holds that $P(x) = 0$ if and only if $\langle \mathbf{z}, \mathbf{x} \rangle = 0$.

CPRFs for t-CNF formulas. Any t-CNF formula can be defined as the AND of $d = \binom{m}{t} \cdot 2^t$ NC^0_t circuits, where NC^0_t is the class of NC^0 circuits that read at most t indices of the input bits [34]. More formally, a t-CNF circuit $C \colon \{0,1\}^m \to \{0,1\}$ can be defined as:

$$C(x) = \bigwedge_{i=1}^{d} C_i(x) \text{ where } C_i \in \mathsf{NC}^0_t. \tag{4}$$

Davidson et al. [34, Appendix C] provide a simple reduction from CPRFs for inner-product predicates to CPRFs for t-CNF formulas. The high level idea is to let $\mathbf{x} = (C_1(x), C_2(x), \ldots, C_d(x), -1)$, where the C_i's describe the t-CNF circuit C, as per Equation (4). The constraint vector is then defined as $\mathbf{z} = (z_1, \ldots, z_d, w)$, where $z_i = 1$ if the i-th circuit needs to be satisfied and $z_i = 0$ otherwise, and w is the hamming weight of $(z_1, \ldots, z_d)$. It then holds that $\langle \mathbf{z}, \mathbf{x} \rangle = 1$ if and only if $C(\mathbf{x}) = 0$. This reduction to t-CNF formulas implicitly uses the fact that we can describe constraints as the "AND" of many individual, simpler constraints. We describe this trick explicitly, and explain how it applied to constructing constraint predicates described by matrix-vector products.

Conjunction of constraints. Here, we show that if we have a CPRF for a constraint class $\mathcal{C}$, then we can construct a CPRF for the constraint class $\bigwedge_{i=1}^{d} C_i$ where $\forall i, C_i \in \mathcal{C}$. In a nutshell, we can define the CPRF for "AND constraints" as a vector of d CPRFs such that the output is defined to be the addition of all the individual CPRF outputs. It is not difficult to see that the sum of the d individual CPRF outputs will be consistent with the evaluation under the master secret key if and only if all the constraints are satisfied. To the best of our knowledge, we are the first to formalize this simple folklore extension to CPRFs.

Let $\mathsf{CPRF} = (\mathsf{CPRF.Gen}, \mathsf{CPRF.Eval}, \mathsf{CPRF.Constrain}, \mathsf{CPRF.CEval})$ be a CPRF for constraints in the class $\mathcal{C}$. We construct the CPRF $\widehat{\mathsf{CPRF}}$ for the AND of d constraints in $\mathcal{C}$ as follows. Let $\oplus$ denote the group operation over the range $\mathcal{Y}$.

- $\widehat{\mathsf{CPRF}}.\mathsf{Gen}(1^\lambda, d)$:
  1: Compute $\mathsf{msk}_i \leftarrow \mathsf{CPRF.Gen}(1^\lambda)$ for all $i \in [d]$.
  2: Output $\mathsf{msk} := (\mathsf{msk}_1, \ldots, \mathsf{msk}_d)$.
- $\widehat{\mathsf{CPRF}}.\mathsf{Eval}(\mathsf{msk}, x)$:
  1: Parse $\mathsf{msk} = (\mathsf{msk}_1, \ldots, \mathsf{msk}_d)$.
  2: Compute $y_i \leftarrow \mathsf{CPRF.Eval}(\mathsf{msk}_i, x)$ for all $i \in [d]$.
  3: Output $\bigoplus_{i=1}^{d} y_i$.
- $\widehat{\mathsf{CPRF}}.\mathsf{Constrain}(\mathsf{msk}, \widehat{C})$:

1: Parse $\mathsf{msk} = (\mathsf{msk}_1, \ldots, \mathsf{msk}_d)$ and $\widehat{C} = (C_1, \ldots, C_d) \in \mathcal{C}^d$.
2: Compute $\mathsf{csk}^{(i)} \leftarrow \mathsf{CPRF.Constrain}(\mathsf{msk}_i, C_i)$ for all $i \in [d]$.
3: Output $\mathsf{csk} := (\mathsf{csk}^{(1)}, \ldots, \mathsf{csk}^{(d)})$.

– $\widehat{\mathsf{CPRF}}.\mathsf{CEval}(\mathsf{csk}, x)$:
1: Parse $\mathsf{csk} = (\mathsf{csk}^{(1)}, \ldots, \mathsf{csk}^{(d)})$.
2: Compute $y_i \leftarrow \mathsf{CPRF.CEval}(\mathsf{csk}^{(i)}, x)$ for all $i \in [d]$.
3: Output $\bigoplus_{i=1}^{d} y_i$.

We prove the following proposition with regards to the above construction.

Proposition 2. *Let* $\mathsf{CPRF} = (\mathsf{CPRF.Gen}, \mathsf{CPRF.Eval}, \mathsf{CPRF.Constrain}, \mathsf{CPRF.CEval})$ *be a CPRF for constraints in the class* $\mathcal{C}$. *Then* $\widehat{\mathsf{CPRF}}$ *is a CPRF for constraint predicates described as* $\bigwedge_{i=1}^{d} C_i$, *where* $C_i \in \mathcal{C}$. *Moreover, if* CPRF *is constraint-hiding, then so is* $\widehat{\mathsf{CPRF}}$.

Proof sketch. We briefly sketch the proofs of correctness and security.

Correctness. Correctness holds because if all d constraints $C_1, \ldots, C_d$ are satisfied, then $\widehat{\mathsf{Eval}}$ and $\widehat{\mathsf{CEval}}$ agree on all y_i computed as $\mathsf{CPRF.Eval}(\mathsf{msk}_i, x)$ and $\mathsf{CPRF.CEval}(\mathsf{csk}^{(i)}, x)$, respectively. It then follows that the sum of the outputs is identical under both the master secret key and constrained key.

Security. If at least one $C_1, \ldots, C_d$ is *not* satisfied, then $\mathsf{CPRF.CEval}(\mathsf{csk}^{(i)}, x)$, for at least one $i \in [i]$ will output a pseudorandom value in $\mathcal{Y}$ (by the security of CPRF). By a straightforward hybrid argument, it then follows that $\widehat{\mathsf{CPRF}}.\mathsf{CEval}(\mathsf{csk}^{(i)}, x)$ outputs a pseudorandom value that is independent of the CPRF evaluation under the master key. Constraint hiding follows by a similar hybrid argument. ■

Matrix-vector product constrains. As a corollary of Proposition 2 and our constructions of CPRF for inner-product predicates, we can construct CPRFs for constraints where the constraint is satisfied if and only if $\mathbf{Ax} = \mathbf{0}$, for some constraint matrix $\mathbf{A}$. Specifically, for a matrix $\mathbf{A} \in \mathbb{F}^{d \times \ell}$ where $(\mathbf{a}_1, \ldots, \mathbf{a}_d) \in (\mathbb{F}^{\ell})^d$ is the vector of rows of $\mathbf{A}$, it holds that $\mathbf{Ax} = \mathbf{0} \iff \bigwedge_{i=1}^{d} \langle \mathbf{a}_i, \mathbf{x} \rangle = 0$.

B Application to Learning Theory

In this section, we highlight known connections between learning theory and CPRFs and provide a corollary that is implied by our CPRF construction from DDH.

Membership queries with restriction access. Motivated by the goal of providing stronger lower bound in learning theory, Cohen, Goldwasser, and Vaikuntanathan [29] introduce a learning model they call MQ *with Restriction Access* ($\mathrm{MQ}_{\mathsf{RA}}$) and show that CPRFs naturally define a concept class that is not learnable, even when the learner obtains non-black-box access to the function on a restricted subset of the domain. Informally, in the basic MQ learning framework [61] (without restriction access), a learner gets oracle access to a function and must approximate the function after a sufficient number of queries. *Restriction access* [36] is a different model in learning theory, where the learner obtains a non-black-box implementation of the function computing a restricted set of function evaluations. Cohen et al. merge the two model to introduce the model of MQ with Restriction Access ($\mathrm{MQ}_{\mathsf{RA}}$), where in addition to black-box membership queries, the learner obtains non-black-box access to a restricted "simplified" version of the function. We provide the informal definition here, and point the reader to Cohen et al. for details and further discussion.

Definition 9 (Membership queries with restriction access ($\mathrm{MQ}_{\mathsf{RA}}$) [29]). *Let* $\mathcal{C}\colon \mathcal{X} \to \{0,1\}$ *be a concept class, and* $\mathcal{S} = \{S \subseteq \mathcal{X}\}$ *be a collection of subsets of the domain* $\mathcal{X}$. $\mathcal{S}$ *is the set of allowable restrictions for concepts* $f \in \mathcal{C}$. *Let* Simp *be a "simplification rule" which, for a concept* f *and restriction* S *outputs a "simplification" of* f *restricted to* S. *An algorithm* $\mathcal{A}$ *is an* $(\epsilon, \delta, \alpha)$-$\mathrm{MQ}_{\mathsf{RA}}$ *learning algorithm for representation class* $\mathcal{C}$ *with respect to a restrictions in* $\mathcal{S}$ *and simplification rule* Simp *if, for every* $f \in \mathcal{C}$, $\Pr\left[\mathcal{A}^{\mathsf{Simp}(f,\cdot)} = h\right] \geq 1 - \delta$, *where* h *is an* ϵ*-approximation to* f, *and furthermore,* $\mathcal{A}$ *only requests restrictions for an* α*-fraction of the whole domain* $\mathcal{X}$.

Cohen et al. prove the following theorem (restated here in its informal version since the formal definitions require substantial notation):

Theorem 9 (Informal). *Suppose $\mathcal{F}$ is a family of constrained PRFs which can be constrained to sets in $\mathcal{S} = \{S \subseteq \mathcal{X}\}$. If $\mathcal{F}$ is computable in circuit complexity class $\mathcal{C}$, then $\mathcal{C}$ is hard to MQ_{RA}-learn with restrictions in $\mathcal{S}$.*

Let $\mathcal{IP} = \left\{\{\mathbf{x}_1, \ldots, \mathbf{x}_N, \mathbf{z}\} \mid \mathbf{x}_1, \ldots, \mathbf{x}_N, \mathbf{z} \in \mathbb{F}^\ell; \langle \mathbf{x}_i, \mathbf{z}\rangle = 0, \forall i \in [N]\right\}_{N \in \mathbb{N}}$ be the subsets of the input domain $\mathbb{F}^\ell$ that satisfy the inner-product relation with respect to a vector $\mathbf{z}$. Using our CPRFs for inner-product predicates, we immediately obtain the following two corollaries.

Corollary 2. *Assuming the DDH assumption holds in a cyclic group $\mathbb{G}$, there is a simplification rule such that NC^1 is hard to MQ_{RA}-learn with respect to restrictions in $\mathcal{IP}$.*

In particular, Corollary 2 uses the fact that our DDH-based CPRF construction can be evaluated in NC^1 (recall Remark 7).

C Collision-resistant Hashing from Discrete Logarithms

Here, we describe a construction of Collision-resistant Hash Function (CRHF) family from the Discrete Logarithm (DL) assumption that generalizes the construction of Damgård [33] in the natural way. Importantly, this construction is in the complexity class NC^1, which makes the CPRF construction from the DDH assumption (when instantiated with this DL-based CRHF family) have an evaluation function that is computable in the complexity class NC^1.

Construction. Fix a prime-order group $\mathbb{G}$ in which the discrete logarithm problem is hard and let $\mathsf{extract}\colon \mathbb{G} \to \{0,1\}^k$ be a randomness extractor with $\lambda \leq k \leq \log_2(|\mathbb{G}|)$ with public parameters pp_e. Let $p > 2^\lambda$ be the order of $\mathbb{G}$ and define the CRHF family $\mathcal{H} = \left\{h_{\mathbf{g}}\colon \mathbb{Z}_p^n \to \{0,1\}^k\right\}_{\mathbf{g} \in \mathbb{G}^n}$, parameterized by n random generators $\mathbf{g} = (g_1, \ldots, g_n)$ and public parameters pp consisting of the group description and pp_e, where the function $h_{\mathbf{g}}\colon \mathbb{Z}_p^n \to \{0,1\}^k$ is defined as

$$h_{\mathbf{g}}(\mathbf{x}) = \mathsf{extract}(\prod_{i=1}^{n} g_i^{\mathbf{x}_i}).$$

Claim. The function family $\mathcal{H} := \left\{h_{\mathbf{g}}\colon \mathbb{Z}_p^n \to \{0,1\}^k\right\}_{\mathbf{g} \in \mathbb{G}^n}$ is a CRHF family.

Proof. Consider the simpler hash function $\hat{h}_{\mathbf{g}}(\mathbf{x}) = \prod_{i=1}^n g_i^{\mathbf{x}_i}$ parameterized by $\mathbf{g} = (g_1, \ldots, g_n)$. Suppose, towards contradiction, that there exists an efficient $\mathcal{A}$ that finds a pair of colliding inputs to $\hat{h}_{\mathbf{g}}$ with non-negligible probability $\nu(\lambda)$. Then, on input $(1^\lambda, \mathbb{G}, \mathbf{g})$, $\mathcal{A}$ outputs $(\mathbf{x}, \mathbf{x}')$ such that $\mathbf{x} \neq \mathbf{x}'$ and $\hat{h}_{\mathbf{g}}(\mathbf{x}) = \hat{h}_{\mathbf{g}}(\mathbf{x}')$, with probability at least $\nu(\lambda)$. Therefore, when $\mathcal{A}$ succeeds, we have that $\prod_{i=1}^n g_i^{\mathbf{x}_i} = \prod_{i=1}^n g_i^{\mathbf{x}'_i}$. We can use $\mathcal{A}$ to solve the discrete logarithm problem as follows. On input a generator g for $\mathbb{G}$ and an element $y \in \mathbb{G}$,

1: Sample $i \xleftarrow{\text{R}} [n]$.
2: Sample $(a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_n) \xleftarrow{\text{R}} \mathbb{Z}_p^{n-1} \setminus \{0\}$.
3: Set $\mathbf{g} = (g^{a_1}, \ldots, g^{a_{i-1}}, y, g^{a_{i+1}}, \ldots, g^{a_n})$.
4: Run $\mathcal{A}$ on input $(1^\lambda, \mathbf{g})$ and obtain as output $(\mathbf{x}, \mathbf{x}')$.
5: Compute $z \leftarrow \sum_{j=1, j\neq i}^n a_j \mathbf{x}_j$ and $z' \leftarrow \sum_{j=1, j \neq i}^n a_j \mathbf{x}'_j$.
6: Output $a_i \leftarrow (z' - z)/(\mathbf{x}_i - \mathbf{x}'_i)$.

We now analyze the reduction. The probability that $\mathbf{x}_i \neq \mathbf{x}'_i$ is at least $\frac{1}{n}$ because i is chosen uniformly from the set $\{1, \ldots, n\}$. Second, observe that

$$\sum_{j=1}^{n} a_j \mathbf{x}_j - \sum_{j=1}^{n} a_j \mathbf{x}'_j = z - z' + a_i(\mathbf{x}_i - \mathbf{x}'_i) = 0,$$

which implies that $(z' - z)/(\mathbf{x}_i - \mathbf{x}'_i) = a_i$. As such, the reduction succeeds with probability $\frac{1}{n}\nu(\lambda)$, which is non-negligible, contradicting the discrete logarithm assumption in $\mathbb{G}$.

Finally, it follows that $h_{\mathbf{g}}$ is a CRHF if $\hat{h}_{\mathbf{g}}$ is a CRHF because $\mathsf{extract}$ is a randomness extractor and $k \geq \lambda$, making the advantage of $\mathcal{A}$ in the case where it is given outputs of the randomness extractor equivalent to the case where it is given the explicit description of group elements. Specifically, this follows from a random element of $\mathbb{G}$ having at least λ bits of min entropy.

■

D Deferred proofs

D.1 Proof of Lemma 3

Proof. The proof is almost identical (occasionally taken verbatim) to the related proof of Applebaum and Widder [2, Lemma 7.2] for the case of additive functions. However, it differs in several key places where we must consider *affine* functions and their impact on the corresponding distributions, which has sufficient repercussions to necessitate rewriting the proof in full.

Fix a sequence of t distinct affine functions $\boldsymbol{\phi} := (\phi_0, \phi_1, \ldots, \phi_t)$ where we define ϕ_0 to be the identity function for notational convenience. We say that $h_z \in H$ is ε-good for $\boldsymbol{\phi}$ if for a random s, the distribution $\{h_z(\phi_i(s))\}_{0 \leq i \leq t}$ is ε-close to the uniform distribution over $\mathcal{K}^{t+1}$. In order to bound the statistical distance, we must prove the following claim.

Claim. For all but a $2^{-2\lambda(t+1)-\lambda}$-fraction of the $h \in H$ the following holds. For every vector of (not necessarily distinct) keys $\mathbf{k} := (k_0, \ldots, k_t) \in \mathcal{K}^{t+1}$,

$$\Pr_{s \xleftarrow{R} S}\left[\bigwedge_{i=0}^{t} h(\phi_i(s)) = k_i\right] \in \left(\frac{1}{|\mathcal{K}|^{t+1}} \cdot (1 \pm 2^{-0.99\lambda})\right).$$

Proof. Fix a vector of keys $\mathbf{k} \in \mathcal{K}^{t+1}$. For every $s \in S$, define the indicator random variable χ_s which takes on the value 1 if $h(\phi_i(s)) = k_i$ for all $i \in \{0, 1, \ldots, t\}$ and a random choice of $h \in H$. Observe that the random variable $\bar{\chi}$ taking the value of $\Pr_s[\bigwedge_{i=0}^{t} h(\phi_i(s)) = k_i]$, and induced by a choice of h, can be written as $\bar{\chi} = \sum_{s \in S} \frac{\chi_s}{|S|}$. Next, we must prove the following bound:

$$\Pr_{h \xleftarrow{R} H}\left[\bar{\chi} \notin \left(\frac{1}{|\mathcal{K}|^{t+1}} \cdot (1 \pm 2^{-0.99\lambda})\right)\right] \leq 2^{-3\lambda(t+1)-\lambda}, \tag{5}$$

which we will later use to prove the claim via a simple union bound. To prove Equation (5), observe that since H is an m-wise independent hash family and $m > t+1$, we have that $\mathbb{E}[\chi_s] = 1/|\mathcal{K}|^{t+1}$, for every s. Then, by linearity of expectation, it is easy to see that $\mathbb{E}(\bar{\chi}) = 1/|\mathcal{K}|^{t+1}$. Next, we show that the average of χ_s is concentrated around its expectation. Following the proof of Applebaum and Widder [2, Claim 7.3], we can show that the χ_s's are r-wise independent, for $r \geq 3t + 5$, which yields a strong concentration bound despite local dependencies in the χ_s's (see the result of Gradwohl and Yehudayoff [43] for an overview of the deployed proof strategy).

To formally prove the bound, define a graph G over any pair of $s, s' \in S$ by placing an edge between s and s' if $\phi_i(s) = \phi_j(s')$ for some $i \neq j$. It then follows that the degree of each node in G is at most $d = (t+1)^2$. We claim that for every independent set I in the graph, the random variables $\{\chi_s : s \in I\}$ are r-wise independent (or using the terminology of Gradwohl and Yehudayoff [43], the random variable r-agree with G). To show this, consider any independent set $I \subseteq S$. For any r-sized subset $(s_1, \ldots, s_r) \subseteq I$, the value of each random variable χ_{s_j} for $s_j \in I$, solely depends on the value of h evaluated on the set of $t+1$ points $(\phi_0(s_j), \phi_1(s_j), \ldots, \phi_{t+1}(s_j))$. Moreover, observe that for all choices of $t+1$ distinct affine functions $(\phi_1, \ldots, \phi_{t+1})$, all elements of the set $\{\phi_0(s_j), \phi_1(s_j), \ldots, \phi_{t+1}(s_j)\}$ are also distinct with probability at least $1 - \frac{t+1}{2^\lambda}$, since the probability of a collision between any distinct ϕ_u and ϕ_v is exactly $1/|S| < 2^{-\lambda} \leq 1/|\mathcal{K}|$. It then follows (via a union bound and using the fact that I is an independent set) that the sets $\{\phi_0(s_j), \phi_1(s_j), \ldots, \phi_{t+1}(s_j)\}$ for all $j \in [r]$ are distinct with probability at least $1 - \frac{r(t+1)}{2^\lambda}$.

From the above, we conclude that with all but negligible probability in λ, the image of these sets under a randomly chosen h are statistically independent, since h is m-wise independent for $m \geq r(t+1)$. It then follows that $\chi_{s_1}, \ldots, \chi_{s_r}$ are statistically independent, or in other words, agree with G [43]. Applying the bound of [43, Corollary 3.2] and taking into account the negligible collision probability computed above, we get that:

$$\Pr_{h \xleftarrow{\mathrm{R}} H}\left[\bar{\chi} \notin \left(\frac{1}{|\mathcal{K}|^{t+1}} \cdot (1 \pm \delta)\right)\right] < 4\sqrt{\pi r}\left(\frac{|\mathcal{K}|^{t+1}\sqrt{(d+1)r}}{\delta\sqrt{|S|}}\right)^{r} + \frac{r(t+1)}{2^{\lambda}}. \tag{6}$$

Then, setting $\delta = 2^{-0.99\lambda}$, $|\mathcal{K}| = 2^{\lambda}$, $|S| = 2^{(2t+6)\lambda}$, and $r, t \in \mathsf{poly}(\lambda)$, Equation (6) is upper-bounded by $2^{-\lambda r} \leq 2^{-3\lambda(t+1)-\lambda}$ for all sufficiently large λ (recall that $d = (t+1)^2$ and $r \geq 3t+5$), and so Equation (5) follows. The claim then follows by applying a union bound over all $2^{\lambda(t+1)}$ possible $\mathbf{k} \in \mathcal{K}^{t+1}$, since $\lambda(t+1) - 3\lambda(t+1) - \lambda = -2\lambda(t+1) - \lambda$. ■

To complete the proof of the lemma, note that any h that satisfies the lemma is $2^{-0.99\lambda}$-good (as defined in the beginning of the proof) for the fixed sequence of affine functions $\boldsymbol{\phi}$. Specifically, $(h(\phi_0(s)), \ldots, h_t(\phi_{t+1}(s)))$ has a statistical distance of at most $2^{-0.99\lambda}$ from the uniform distribution. Moreover, as shown above, all but a $2^{-2\lambda(t+1)-\lambda}$-fraction of the $h \in H$ are t-good for the fixed vector $\boldsymbol{\phi}$. By applying a union bound over all possible $2^{2\lambda(t+1)}$ affine functions, we conclude that all but a $2^{-\lambda}$-fraction of the $h \in H$ are t-good, in the sense of Definition 7, and the lemma follows. ■