On Arithmetic Private Information Retrieval: Why Code-Based PIR (Usually) Fails *

Benny Applebaum
Tel-Aviv University
Tel-Aviv, Israel
benny.applebaum@gmail.com

Yuval Ishai
Technion and AWS†
yuval.ishai@gmail.com

Shahar Shechter
Tel-Aviv University
Tel-Aviv, Israel
shahar.shechter@gmail.com

June 10, 2026

Abstract

We initiate the study of *arithmetic* private information retrieval (APIR) schemes, in which the database is a vector of field elements and the scheme makes a black-box use of the field. We obtain the following results.

- Our main result is a negative one: We show that no single-server APIR scheme can achieve non-trivial download cost smaller than n field elements. We observe that recent proposals for code-based PIR (Holzbaur et al., ISIT'20; Verma and Hollanti, ISIT'24) are arithmetic, and show how to break them within a few minutes on a standard workstation for all suggested parameters.
- We complement the above by positive results in alternative models. Concretely, we show that with either two servers or a single server with secret-key preprocessing, it is possible to construct computationally secure APIR schemes based on well-studied coding assumptions. This is achieved by arithmetizing the distributed-point-function-based PIR of Boyle et al. (CCS'16), and by observing that the recent construction of secret-key single-server PIR by Chen et al. (STOC'26) also arithmetizes.
- Finally, we characterize the existence of information-theoretic two-server APIR schemes in linear-algebraic terms, and show that communication of $O(n^{1/3})$ can be achieved in this setting based on the original approach of Chor et al. (FOCS'95). The optimality of this result remains an interesting open question.

1 Introduction

Private information retrieval (PIR), introduced by Chor et al. [25], concerns the task of a client retrieving an item from a database while keeping the identity of the requested item private. In a PIR scheme, the

*B. A. and S. S. are supported by Israel Science Foundation (ISF) grant no. 2805/21 and by the European Union (ERC-2022-ADG) under grant agreement no.101097959 NFITSC. Y. I. supported by Israel Science Foundation (ISF) grant 3527/24 and BSF grant 2022370.

†This work is not associated with Amazon.

client interacts with one or more servers storing the data in such a way that the servers learn nothing about which item the client is accessing beyond what is implied by the database itself, while the client successfully obtains the desired information.

The existence of single-server PIR was first established by Kushilevitz and Ostrovsky [49] under a standard number-theoretic assumption. In the past decade, a large number of practical constructions have been proposed, mostly based on lattice assumptions (see [53, 6, 2, 4, 56, 54, 42, 30, 64, 55]). Following these developments, researchers from the coding theory community presented several constructions of code-based single-server PIR [44, 43]. As in the case of code-based cryptosystems (e.g., McEliece [52] or Alekhnovich [3]), the hope is that code-based PIR schemes will diversify the underlying cryptographic assumptions and, for certain parameter regimes, may even lead to better efficiency.

Code-based PIR. The first code-based construction was proposed by Holzbaur et al. [44] and subsequently broken by Bordage and Lavauzelle [19]. To cope with this attack, Bodur et al. [18] presented a ring-based variant of [44] that was recently broken by [48]. An alternative scheme, termed CB-cPIR, was later introduced by Verma and Hollanti [62] and partially broken by Lage and Bartz [50], though the complexity of their attack scales with the size of the underlying field and does not apply over sufficiently large fields. Moreover, in the full version [43] of [62], the authors further modified the construction to address all previously known attacks (see Remark 6 in [43]). To the best of our knowledge, this revised scheme is currently considered secure.

Interestingly, all the proposed schemes have the additional feature of being *arithmetic*.[1] Roughly speaking, this means that the database $x = (x_1, \ldots, x_n)$ is viewed as a vector over a finite field or a ring, and the PIR algorithms (query generation, answer computation, and result recovery) are implemented by an arithmetic circuit over this domain. The feature of being arithmetic has useful consequences. For example, arithmetic PIR schemes can be used to directly post-process numerical outputs (e.g., compute standard statistics) or distribute the client using arithmetic secure computation. However, this feature is currently unique to code-based PIR candidates. This raises the following questions:

> Under which assumptions or settings is it possible to construct arithmetic PIR schemes? Are current code-based candidates secure?

Similar questions about the arithmetic feasibility of other specific cryptographic primitives were studied (sometimes implicitly) in [31, 57, 27, 28, 32, 45, 11, 26, 9, 12, 15], and were studied more broadly by Applebaum et al. [7].

1.1 Our Contributions

We initiate a systematic study of arithmetic PIR. For this we adapt the general arithmetic-cryptography framework of [7] to the setting of PIR. This requires the PIR algorithms to be implemented by arithmetic circuits over a generic field $\mathbb{F}$, so that the PIR functionality should hold for an arbitrary adversarial choice of $\mathbb{F}$. (See Section 2 for formal definitions.) We derive both positive and negative results for this model.[2]

[1] This is sometimes referred to as being *field-agnostic* or making a *black-box use of the field*. See Section 1.3 for discussion of different variations of this model.

[2] In fact, as we will see, our negative results hold under a significantly weaker notion of arithmetic PIR.

1.1.1 Impossibility of single-server arithmetic PIR

Our main result is negative. We show that, in the arithmetic setting, there is no non-trivial PIR in which the server communicates less than n field elements. (See Section 3.)

Informal Theorem 1.1 (main)**.** *In every single-server arithmetic PIR, the server communicates at least n field elements where n is the length of the database $x \in \mathbb{F}^n$.*

Weakly arithmetic PIR. The theorem applies even to a very weak notion of arithmetic PIR. Roughly speaking, we place no restrictions on the query generation algorithm and impose only the following requirements.

1. The server's answering function $\text{ANS}_q(x)$ is arithmetic in the database $x \in \mathbb{F}^n$, but may depend arbitrarily on the query q.
2. The client's decoding function $\text{DEC}_{i,r}(\mathbf{ans})$ is arithmetic in the server's answer $\mathbf{ans} \in \mathbb{F}^\ell$, but may depend arbitrarily on the private randomness r and on the desired index i.

Here, the term *arithmetic* means that the functions can be computed by arithmetic circuits over $\mathbb{F}$. In fact, the theorem applies even to schemes tailored to a specific field $\mathbb{F}$, as long as the composition $\text{DEC}_{i,r}(\text{ANS}_q(\cdot))$ is a polynomial of degree strictly smaller than the size of the field. Moreover, the theorem generalizes to finite commutative rings. (See Appendix B.)

Distinguishing attack. To prove the theorem, we describe an attack that, given a query q generated by a client seeking to retrieve an index i, outputs a list of candidate indices that includes i. If the server's answer contains $\ell < n$ field elements (i.e., is non-trivial), then the resulting list has size strictly smaller than n. This enables a distinguishing attack that breaks the PIR guarantee, for example by distinguishing the true index i from a uniformly random index j. The attack can be implemented using $O(n\ell^2 + |\text{ANS}|)$ field operations, where $|\text{ANS}|$ denotes the number of operations required to compute the server's answers.

A bit more concretely, the client's query defines ℓ low-degree polynomials $p_1, \ldots, p_\ell$ for which there is another low-degree polynomial p, defined by the secret decoding function, such that $p(p_1, \ldots, p_\ell) = x_i$, where i is the client's index. Namely, x_i is *algebraically dependent* on $p_1, \ldots, p_\ell$. The server's candidate list contains *all* x_j that are algebraically dependent on $p_1, \ldots, p_\ell$. Algebraic dependence over sufficiently large fields can be tested efficiently using techniques from algebraic complexity [34, 46], which we simplify and make explicit for our specific use case. We further extend this attack beyond polynomials to cover more general PIR algorithms computed by arithmetic circuits that include zero-check and division gates.

Practical recovery attack. In Section 3.1 we give a practical version of the above attack that recovers the secret index i from the query q. The attack exploits the block structure of PIR schemes that support retrieval of entire blocks, and we further optimize it using sketching techniques to reduce the running time. We implemented the attack for all known code-based PIR schemes, including schemes which are still considered secure. As shown in Section 4, all parameter sets proposed in the original works are broken within a few minutes on a standard workstation (Intel Core i7-14700KF, 3.4 GHz, 64 GB RAM).

The code-based PIR schemes we empirically attacked turn out to have a very strong arithmetic structure, in which the query defines a compressive *linear function* of the database. A polynomial-time distinguishing attack for such linear PIR schemes was already pointed out in the context of doubly-efficient PIR [21, 22]. Indeed, writing the server's response as Qx for a matrix Q determined by the query, distinguishing between

the correct index i and a random index i' boils down to checking which of the corresponding unit vectors is in the row-span of Q. Our more general arithmetic distinguishing and recovery attacks rely heavily on *formal partial derivatives*. Similar techniques were extensively developed in the arithmetic cryptography framework of [7] and we adapt them here to the setting of arithmetic PIR.

The arithmetic perspective yields a simple and transparent explanation for the failure of code-based PIR constructions. It further extends to PIR over rings, thereby unifying and generalizing all existing attacks.[3] Moreover, it indicates that the underlying insecurity is fundamental and cannot be remedied by low-degree algebraic manipulations. In particular, security inherently requires "non-arithmetic" operations, whose degree scales with the underlying field, such as the ones employed in PIR schemes based on lattices or cryptographic groups.

1.1.2 Bypassing the impossibility: Adding a server or a secret-key

We complement the above negative results by showing that highly-efficient arithmetic PIR can be constructed in two relaxed settings.

Two-server PIR. First, we consider the case of two-server computationally-secure arithmetic PIR. In the non-arithmetic setting, one can achieve highly efficient PIR with communication that scales with $\log n$ based on *distributed point functions* [38]. In Section 6, we extend this notion to the arithmetic setting and show that the construction of [20] can be arithmetized. As a result, we derive the following theorem.

Informal Theorem 1.2. *Assuming the existence of an arithmetic pseudorandom generator (APRG), there exists a 2-server arithmetic PIR with computational security. For a length-n database, the communication complexity is $O(n^\epsilon)$ for an arbitrary constant $\epsilon > 0$.*[4]

As shown in [7], an APRG can be constructed based on the arithmetic version of the *Learning Parity with Noise* (LPN) assumption [17, 40], first proposed in [45]. See Appendix C for details.

Secret-key PIR. Another way to relax the single-server setting is via the use of a *secret-key*. In a secret-key PIR (sk-PIR) [21, 22] the client has a short secret key sk which is used to map the database x into an encoded database $\hat{x}$. In an offline preprocessing phase, the client uploads the encoded database $\hat{x}$ to the server and keeps only the secret-key sk. In the online phase, the client can repeatedly query the ith bit of x, by exchanging only $o(n)$ bits (or field elements) with the server.

Such an sk-PIR scheme was recently suggested by [24] based on arithmetic LPN, and a more efficient variant under the related *Learning Subspace with Noise* (LSN) assumption [33].

Informal Theorem 1.3. *Assuming arithmetic LPN, for every $\epsilon > 0$ there exists an arithmetic sk-PIR scheme with computational security and $O(n^\epsilon)$ communication for database length n.*

Furthermore, under the LSN assumption, the database encoding can be of length $(1 + \epsilon) \cdot n$.

1.1.3 Information-theoretic 2-server arithmetic PIR?

Finally, we revisit the case of two-server arithmetic PIR with information-theoretic security. Traditional constructions of 2-server PIR [25, 14, 63] can be arithmetized, yielding schemes with communication complexity $O(n^{1/3})$. However, we do not know how to arithmetize the best existing binary schemes [35, 5],

[3] Indeed, it is instructive to compare the simplicity of this extension with the attack of [48] on the ring-based PIR of [18].

[4] If the APRG is assumed to have sub-exponential security then the communication can be reduced to $\mathrm{polylog}(n)$.

which achieve sub-polynomial communication $n^{o(1)}$. Indeed, these schemes rely on the matching vectors framework, which does not appear to be field independent. In fact, this framework requires richer algebraic structures, such as rings with zero-divisors.

At present, it is unknown whether one can improve upon the $O(n^{1/3})$ communication bound, let alone achieve sub-polynomial communication, in the arithmetic setting. We view this as an intriguing open question. As a first step toward this goal, we show that any arithmetic PIR scheme can be transformed, with comparable communication complexity, into a PIR that is *linear* in the following sense: (1) The database is a vector of field elements; (2) the answer algorithm of each server is linear in the database; and (3) the decoding algorithm is linear in the servers' answers.[5] This transformation leads to the following equivalence between *arithmetic* 2-server PIR and *linear* 2-server PIR. In fact, this equivalence generalizes to any number of servers.

Informal Theorem 1.4 (See Thm. 5.3)**.** *The following are equivalent.*

1. *There exists a two-server PIR with information-theoretic security and arithmetic server and decoder and communication of $O(\ell(n))$ field elements for a database of size n.*
2. *For every sufficiently large field $\mathbb{F}$, there exist n pairs of distributions*

$$(\mathcal{A}_1, \mathcal{B}_1), \ldots, (\mathcal{A}_n, \mathcal{B}_n)$$

over $O(\ell(n)) \times n$ matrices such that:

   (a) *The rows of $(\mathcal{A}_i, \mathcal{B}_i)$ jointly span the unit vector e_i.*
   (b) *For every $i, j \in [n]$, the marginal distributions $\mathcal{A}_i$ and $\mathcal{A}_j$ (resp., $\mathcal{B}_i$ and $\mathcal{B}_j$) are identical.*
   (c) *The support of $(\mathcal{A}_i, \mathcal{B}_i)$ has size at most $|\mathbb{F}|^{O(\ell(n))}$.*
   (d) *The matrices $\mathcal{A}_i, \mathcal{B}_i$ can be computed by a universal arithmetic circuit that has black-box access to the field $\mathbb{F}$ and is otherwise field independent.*

Sketch. We postpone the proof of the more interesting (1) $\implies$ (2) direction to Section 5, and sketch here only the proof of the easy direction (2) $\implies$ (1). The client samples a pair of matrices $(A, B) \xleftarrow{R} (\mathcal{A}_i, \mathcal{B}_i)$ and sends to each server a description of A (resp., B) using $O(\ell)$ field elements. The servers return Ax (resp., Bx), from which the client recovers x_i as a linear combination of the responses, as guaranteed by item (a). □

Remark 1.1 (computational variant)**.** *The theorem extends to the setting of computationally private PIR by relaxing property (b) so that it requires only that the marginals $\mathcal{A}_i$ and $\mathcal{A}_j$ (resp., $\mathcal{B}_i$ and $\mathcal{B}_j$) are computationally indistinguishable. Indeed, the aforementioned DPF-based constructions give rise to such collections of matrices.*

Comparison to other restricted notions of PIR. Two-server linear PIR schemes were studied by [39, 36], and it remains open whether such schemes over the binary field can be ruled out with $O(\log n)$ communication.[6] Note that Theorem 1.4 gives rise to a 2-server linear PIR scheme that is *field-agnostic*, as captured by property (2d). This additional structure may make it more amenable to lower-bound techniques.

[5] Similarly to the case of linear secret sharing [13], when the database is a vector over $\mathbb{F}$, the existence of an $\mathbb{F}$-linear answering algorithm implies the existence of an $\mathbb{F}$-linear decoding algorithm, and vice versa. The converse direction follows by including only the linear part of the answer functions, as higher-degree monomials must cancel out.

[6] Strong $\Omega(n)$ lower bounds are known when the servers' answers are very short, even for non-linear PIR [47].

It should be emphasized that our definition of linear PIR assumes that the database is a vector of field elements and that linearity is taken over the same field. In contrast, if the database is restricted to a subset of the field (e.g., $\{0,1\}$), linearity can be obtained at a relatively low cost. In particular, the matching-vector-based PIR of [35] yields a 2-server PIR in which the answering algorithm is linear over a small field (such as $\mathbb{F}_3$), while the database $x \in \{0,1\}^n$ consists only of $0/1$ elements. We further note that in this setting decoding is performed via a non-linear operation. Indeed, the equivalence between linear answering and linear recovery (Footnote 5) does not hold in this case.

Finally, we note that a strong $\Omega(n^{1/3})$ lower bound is known for so-called *bilinear* 2-server PIR [58], a notion that appears incomparable to our definition of linearity.

1.2 Motivation

Our primary motivation for the study of arithmetic PIR is similar to the motivation for studying black-box reductions in cryptography: drawing a clear syntactic line between different kinds of techniques from the literature. The simple "litmus test" we provide applies not only to previous PIR candidates and attempts to fix them, but will also guide future efforts. For example, one may try to adapt techniques from the literature on secret-key PIR [21, 22, 24] to the plain model. Our main negative result gives a clear barrier that should be circumvented in such future attempts.

Similarly to black-box reductions, arithmetic PIR is also a useful goal on its own. The broad motivation is similar to recent lines of work on field-agnostic proof systems and MPC protocols: arithmetic PIR supports seamless integration with numerical computations or other computations over fields without needing any conversion steps that may be complex or expensive.

As representative examples, consider the following two specific motivating scenarios:

- **Secure statistics**: Here we use PIR to generate additive shares of a small secret sample of a large numerical database, and then perform a secure post-processing protocol to compute useful statistics (such as average and variance) of the sampled values. See [23] for using PIR in this context. Arithmetic PIR avoids expensive secure type conversion protocols and scales seamlessly to higher levels of precision. In contrast, non-arithmetic state-of-the-art PIR schemes such as SimplePIR [42] were carefully optimized to records that come from a specific (small) domain size, and do not scale as gracefully to natively support larger domains.

- **Distributing the client and/or server**: In MPC use cases, one may want to apply a single-server PIR protocol where the role of the client and/or server is distributed between two or more parties. With arithmetic PIR, this can be done in a simple and efficient way using either a standard "BGW-style" protocol (in the honest-majority case) or fast pseudorandom correlation generators for multiplication triples (in the dishonest-majority case). See Section 6 in [15] for further discussion. In contrast, the client in lattice-based protocols such as SimplePIR is much harder to distribute as it involves non-arithmetic operations such as noise sampling and rounding.

1.3 Other Related Works

Other code-based PIR schemes. Very recently, Abram et al. [1] constructed a single-server PIR with slightly sublinear communication, namely $n/2^{\log^{1-\beta} n}$ for some constant $\beta > 0$, based on the Learning Parity with Noise (LPN) assumption. Specifically, their construction relies on the pseudorandomness of a random noisy codeword in the low-noise regime $O(\log^{1+\beta} \lambda/\lambda)$, where λ denotes the dimension of the random binary linear code, whose rate can be taken to be an arbitrary inverse polynomial.

Not surprisingly, their result does not arithmetize. In particular, it relies on a non-arithmetic mapping of the database, viewed as a binary string, to sparse vectors. Similarly, the LPN-based hash function [10] underlying their construction does not arithmetize.

Arithmetic key-homomorphic encryption. We further note that an alternative form of additively homomorphic symmetric encryption can be constructed from LPN [8], and admits arithmetic realizations [7, Remark 10.6]. For example, given a ciphertext C encrypting an unknown message M under an unknown key K, one can generate a new ciphertext C' that encrypts the message $M' = aM + b_1$ under the key $K' = aK + b_2$, for any scalar a and shift vectors b_1, b_2. This was recently employed for LPN-based secure aggregation [16]. However, it is not known how to leverage this form of homomorphism to obtain a standard additively homomorphic encryption scheme, or a single-server PIR scheme. Indeed, such an approach appears unlikely to succeed in light of our negative result for arithmetic PIR, as well as the negative results of [7] for arithmetic encryption with stronger additive homomorphism.

Arithmetic cryptography. While many information-theoretic primitives admit arithmetic implementations, significantly less is known for computationally secure primitives. In the context of secure multiparty computation, Ishai, Prabhakaran, and Sahai (IPS) [45] proposed an arithmetic model in which the parties are allowed to access the bit representation of field elements, where both the field and its representation are chosen by the adversary. This allows the honest parties to learn an upper bound on the field size and to embed field elements into standard Boolean cryptographic primitives (e.g., encryption or oblivious transfer).[7]

A stronger (more restrictive) arithmetic model was proposed by Applebaum, Avron, and Brzuska [7] (AAB), in which honest algorithms manipulate field elements solely via black-box access to the field. In this model, which will be adopted for this work, the honest parties are agnostic to the underlying field in a stronger sense, for example, they do not even know an upper bound on the field size. In both models, the adversary selects the field and its bit representation and is allowed to mount non-arithmetic attacks.[8] Thus, arithmetic constructions impose additional efficiency restrictions on the honest algorithms without weakening the security requirements.

The feasibility of secure two-party computation in the IPS model was established in [45] and was extended to the AAB model in [7]. On the negative side, [7] ruled out arithmetic garbled circuits and secure two-party protocols with low online communication independent of the output length, as well as different variants of additively homomorphic encryption schemes. While these results are technically incomparable to ours, we rely on similar underlying techniques, most notably the use of partial derivatives.

Finally, it is instructive to compare the distinction between the AAB vs. IPS models for arithmetic cryptography to the distinction between Maurer's [51] and Shoup's [60] variants of generic groups: AAB, like Maurer's formulation, treats algebraic elements as opaque objects accessible only through algebraic operations, whereas IPS, like Shoup's formulation, uses transferable bit-string labels for those elements.

Other arithmetic primitives. We note that *Oblivious Transfer* (OT) [37], a primitive closely related to PIR, can be realized arithmetically under standard cryptographic assumptions [7]. Consequently, the results of the current paper rule out the existence of an arithmetic transformation from OT to PIR, yielding an "arithmetic separation" between the two primitives. Whether a non-arithmetic transformation from OT to

[7] Indeed, in this model one can realize a single-server arithmetic PIR from any "standard" single-server PIR by having the client retrieve each bit of the representation of the i-th field element. This increases the download cost by a factor proportional to the bit-length of the field representation.

[8] Nevertheless, all attacks considered in this work are purely arithmetic.

PIR exists remains an open question; partial black-box impossibility results for the standard model are implied by [41]. Finally, we note that the converse transformation, from PIR to OT [29], extends to the AAB arithmetic model.

Other arithmetic PIR variants. While our main negative result rules out standard arithmetic PIR with non-trivial communication, other APIR variants fall outside the scope of this lower bound. For example, the bound does not apply to PIR schemes in which the client maintains and updates a database-dependent secret state. In particular, the Piano system [64] admits an arithmetization and achieves non-trivial communication together with sublinear server computation.

Organization. Following the preliminaries in Section 2, we present our main attack against single-server arithmetic PIR in Section 3, and specialize it to coding-based schemes in Section 4. We then turn to two-server arithmetic PIR, devoting Section 5 to the information-theoretic setting and Section 6 to the computational setting. Secret-key arithmetic PIR is discussed in Section 7. Extensions of the main attack are presented in Appendix A, which considers more general arithmetic circuits, and in Appendix B, which treats the ring setting. Background on arithmetic PRGs and arithmetic Distributed Point Functions is provided in Appendices C and D, respectively.

2 Preliminaries

2.1 Arithmetic preliminaries

The following arithmetic conventions are mostly taken from [7].

2.1.1 Field implementations

Throughout the paper, we consider finite fields whose elements have an efficient representation and whose field operations are efficiently computable. Specifically, we will use the following definition.

Definition 2.1 (Circuit implementation of a field)**.** *Let F be a Boolean circuit which takes as an input, an operation* op *in*

$$\{\mathsf{add}, \mathsf{subtract}, \mathsf{multiply}, \mathsf{divide}, \mathsf{constant}, \mathsf{zerocheck}, \mathsf{sample}, \mathsf{bitsample}\}$$

(using an appropriate encoding) and up to two binary strings of length ℓ. F is said to be a valid implementation of the finite field $\mathbb{F}$*, if there is an injective mapping* $\mathsf{label} : \mathbb{F} \to \{0,1\}^\ell$ *such that the following holds:*

- *For every operation* $\mathsf{op} \in \{\mathsf{add}, \mathsf{subtract}, \mathsf{multiply}, \mathsf{divide}\}$ *and any $x, y \in \mathbb{F}$ it holds that*

  $$F(\mathsf{op}, \mathsf{label}(x), \mathsf{label}(y)) = \mathsf{label}(x *_{\mathbb{F}} y)$$

  where $*_{\mathbb{F}}$ is the corresponding field operation. A division by zero may lead to an arbitrary output.[9]
- $F(\mathsf{constant}, 0^\ell) = \mathsf{label}(0)$*, and* $F(\mathsf{constant}, 1^\ell) = \mathsf{label}(1)$.
- *If* $a = \mathsf{label}(0)$*, then* $F(\mathsf{zerocheck}, a) = \mathsf{label}(1)$.

[9] When zerocheck gates are allowed, we may assume that a zerocheck is always applied to the divisor.

- *If* $a = \mathsf{label}(x)$ *for* $x \in \mathbb{F}$, $x \neq 0$, *then* $F(\mathsf{zerocheck}, a) = \mathsf{label}(0)$.
- $F(\mathsf{sample})$ *implements the uniform distribution over* $\{\mathsf{label}(x) : x \in \mathbb{F}\}$.
- $F(\mathsf{bitsample})$ *implements the uniform distribution over* $\{\mathsf{label}(x) : x \in \{0,1\}\}$.

2.1.2 Arithmetic Circuits

An arithmetic circuit is a circuit whose wires carry field elements, and whose gates perform arithmetic operations. The circuit is black-box with respect to the field representation, i.e., it does not have access to the bit representation of the elements carried by its wires. In addition to the standard arithmetic operations, we allow "checking for zero". Although not an arithmetic operation, we find it hard to design meaningful schemes when even equality cannot be recognized and so we allow for this operation in our model.

Randomized arithmetic circuits also have two special, randomized gates: The sample gate draws elements from the underlying field $\mathbb{F}$ uniformly at random, and the bitsample gate draws a uniformly random element from the set $\{0,1\}$ which contains the zero and one elements of the field.

Definition 2.2 (Arithmetic Circuit)**.** *An arithmetic circuit is a directed acyclic graph. Each input gate (source) is labeled by an input variable* x_i*, a constant* 1 *or* 0*, or a randomized gate* sample *or* bitsample*. Internal gates are labeled by:*

$$\{\mathsf{add}, \mathsf{subtract}, \mathsf{multiply}, \mathsf{divide}, \mathsf{zerocheck}\}$$

For an arithmetic circuit $C(x)$ *with* n *input variables and* m *output variables any field* $\mathbb{F}$ *induces in the natural way a (randomized) mapping* $C^{\mathbb{F}}(x) : \mathbb{F}^n \to \mathbb{F}^m$*. Furthermore a field implementation* F *naturally induces a Boolean circuit* $C^F(x)$ *which implements the mapping* $C^{\mathbb{F}}(x)$ *with the representation* F*. We use* $C^F(x)$*,* $C^{\mathbb{F}}(x)$ *and* $C(x)$ *interchangeably, when* F *and* $\mathbb{F}$ *are clear from context.*

In some cases, it is convenient to treat certain inputs as *binary* inputs. This can be captured by the above definition by adopting the convention that such inputs are restricted to values in $\{0,1\}$. Note that Boolean logic over these inputs can be implemented using standard arithmetization, for example by translating $\neg x$ to $1 - x$ and $x \wedge y$ to $x \cdot y$.

Throughout this work when discussing a family of circuits $C = \{C_n\}$ we assume by default polynomial time uniformity. That is, there exists a PPT Turing machine that on input 1^n outputs a description of C_n.

We will occasionally consider restricted forms of arithmetic circuits that use only a subset of the types of gates. Most notably, we will consider *deterministic* arithmetic circuits (which do not use sample, bitsample gates), deterministic arithmetic circuits without zerocheck gates (which compute rational functions) and deterministic arithmetic circuits without zerocheck, divide gates (which compute polynomials). We refer to the latter (most restricted) type of circuits as *strictly arithmetic circuits*.

2.2 Arithmetic PIR

Definition 2.3 (k-server Arithmetic PIR Scheme)**.** *For an integer* $k \geq 1$*, a* k*-server arithmetic PIR scheme consists of the following three families of polynomial-sized arithmetic circuits, each parameterized by the database length* n*:*

- *A randomized query circuit* $\text{QUERY}_n^{\mathbb{F}}(i; r)$ *that given an index* $i \in [n]$ *in binary form and randomness* $r = (r_A, r_B) \xleftarrow{R} \mathbb{F}^{\rho_A(n)} \times \{0,1\}^{\rho_B(n)}$ *outputs a vector of queries* $q = (q_1, ..., q_k) \in (\mathbb{F}^{t(n)})^k$*. We*

refer to $t(n)$ as the query length *(or upload cost). We say that r is* chosen uniformly *if $(r_A, r_B) \xleftarrow{R} \mathbb{F}^{\rho_A(n)} \times \{0,1\}^{\rho_B(n)}$, i.e., the arithmetic part is chosen uniformly from the field and the binary part is chosen as a uniform string.*[10]

- *A deterministic answer circuit $\text{ANS}_n^{\mathbb{F}}(x, q)$ that given a query $q \in \mathbb{F}^{t(n)}$ and database $x \in \mathbb{F}^n$ outputs an answer $\mathbf{ans} \in \mathbb{F}^{\ell(n)}$. We refer to $\ell(n)$ as the* answer length *(or download cost).*

- *A deterministic decoding circuit $\text{DEC}_n^{\mathbb{F}}(i, r, \mathbf{ans}_1, ..., \mathbf{ans}_k)$ that given an index $i \in [n]$ in binary, randomness $r = (r_A, r_B) \in \mathbb{F}^{\rho_A(n)} \times \{0,1\}^{\rho_B(n)}$ and vector of answers $(\mathbf{ans}_1, ..., \mathbf{ans}_k)$ outputs a single field element y.*

We will typically write the index i as a subscript and write $\text{QUERY}_{i,n}^{\mathbb{F}}(r)$ and $\text{DEC}_{i,n}^{\mathbb{F}}(r, \mathbf{ans}_1, ..., \mathbf{ans}_k)$ to emphasize that i is binary.

The algorithms should satisfy the following properties:

- *Correctness: For every finite field $\mathbb{F}$, database $x \in \mathbb{F}^n$ and every index $i \in [n]$, consider the randomness r of the client and let $(q_1, \ldots, q_k) = \text{QUERY}_{i,n}^{\mathbb{F}}(r)$, then the error probability (over the randomness r)*

$$\Pr_r[\text{DEC}_{i,n}^{\mathbb{F}}(r, \text{ANS}_n^{\mathbb{F}}(x, q_1), \ldots, \text{ANS}_n^{\mathbb{F}}(x, q_k)) \neq x_i]$$

  is negligible in n. When the error is zero, we say that the PIR is perfectly correct.[11]

- *Privacy (against a single server): For every adversary $\mathcal{A} = \{\mathcal{A}_n\}$ (modeled as a $\text{poly}(n)$-size Boolean circuit) the winning probability in the following game (parameterized by n) is at most $\frac{1}{2} + \text{negl}(n)$:*

  - *The adversary $\mathcal{A}_n$ chooses an implementation of a field $\mathbb{F}$, a pair of indices $(i_0, i_1) \in [n]^2$ and a "corrupted" server $t \in [k]$ and sends to the challenger the tuple $(\mathbb{F}, i_0, i_1, t)$.*
  - *The challenger samples a bit $b \xleftarrow{R} \{0,1\}$ and computes $(q_1, ..., q_k) = \text{QUERY}_{i_b,n}^{\mathbb{F}}(r)$ where r is chosen uniformly and sends back to the adversary the t-th query q_t.*
  - *The adversary $\mathcal{A}$ outputs a bit b' and wins if $b' = b$.*

Privacy is statistical *if the above holds against computationally-unbounded adversary and* perfect *if, in addition, the winning probability is at most 0.5.*

We say that a single-server arithmetic PIR has degree *d if the composition of the decoding and answer circuits can be written as a degree d multivariate polynomial in the database $x \in \mathbb{F}^n$. Formally, the degree is d if for every i, n, r and query $q \in \mathbb{F}^t$, the mapping*

$$\text{DEC}_{i,n}^{\mathbb{F}}(r, \text{ANS}_n^{\mathbb{F}}(x, q))$$

is a multivariate polynomial whose total degree is at most d in the database $x \in \mathbb{F}^n$. If either the decoding or the answer circuit employs zero-test gates or division gates, the degree is undefined and is therefore referred to as infinite.

[10] In fact, our attacks also apply to the more general case where the randomness is chosen according to an arbitrary distribution $D(\mathbb{F}, n)$ that makes a non-black-box use of the underlying field.

[11] One can consider a relaxed version of the definition where correctness holds only over efficiently realizable fields $\mathbb{F} = \{\mathbb{F}_n\}$ (i.e., when $\{\mathbb{F}_n\}$ can be implemented by a family of $\text{poly}(n)$-size circuits). Our negative results apply to this setting as well, while our positive results achieve the stronger version of correctness for every field.

The above definition can be relaxed by requiring privacy and correctness to hold over a specific finite field $\mathbb{F}$. We refer to such a PIR as an $\mathbb{F}$-*arithmetic PIR* or simply *PIR over the field* $\mathbb{F}$. Note that the standard notion of PIR corresponds to $\mathbb{F}_2$-arithmetic PIR.

3 Attacking Arithmetic PIR

We prove the following key-lemma. We restrict our attention to the case of PIR with finite degree and extend the results to general arithmetic circuits with infinite degree (that make use of zero-check gates and division gates) in Section A.

Lemma 3.1 (key Lemma)**.** *Let* $(\text{QUERY}_n^{\mathbb{F}}, \text{ANS}_n^{\mathbb{F}}, \text{DEC}_n^{\mathbb{F}})$ *be a single-server* $\mathbb{F}$*-arithmetic PIR of degree* $d < |\mathbb{F}|$ *with answer length* $\ell(n)$ *perfect correctness. Then, there exists an efficient deterministic algorithm* $\mathcal{M}$ *that takes a query* q *and outputs a matrix* $M_q \in \mathbb{F}^{\ell(n)\times n}$ *such that for every index* $i \in [n]$ *and a query* q *in the support of* $\text{QUERY}_n^{\mathbb{F}}(i)$*, the rows of* $M_q = \mathcal{M}(q)$ *span the* i*th unit* e_i*. Specifically, the matrix* M_q *can be computed by using* $O(|\text{ANS}_n|)$ *arithmetic operations where* $|\text{ANS}_n|$ *is the size of the arithmetic circuit* ANS_n*.*

Proof. Given a query q the algorithm $\mathcal{M}$ computes the matrix M as follows. Define the circuit $A(x) = \text{ANS}_n^{\mathbb{F}}(x, q)$. Let $\mathcal{J}A(x) : \mathbb{F}^n \to \mathbb{F}^{\ell\times n}$ be the *formal Jacobian* matrix of $A(x)$, i.e., $\mathcal{J}A(x)$ is a matrix of multivariate polynomials whose (i, j)-th entry is the formal partial derivative of the i-th output of $A(x) : \mathbb{F}^n \to \mathbb{F}^\ell$ with respect to the j-th input (the formal variable x_j). Fix some database $\hat{x} \in \mathbb{F}^n$, e.g., the all-zero database, and compute the matrix of field elements $M = \mathcal{J}A(\hat{x}) \in \mathbb{F}^{\ell\times n}$.[12]

Analysis. Assume that $q = \text{QUERY}_n^{\mathbb{F}}(i, r)$ for some index i and randomness r. We show that e_i is spanned by M. Define the $\mathbb{F}$-arithmetic circuit $D(\mathbf{ans}) = \text{DEC}_{i,n}^{\mathbb{F}}(r, \mathbf{ans})$. By perfect correctness, for every $x \in \mathbb{F}^n$ it holds that

$$D(A(x)) = x_i. \tag{1}$$

Let us view both the RHS and LHS of the equation as a multivariate polynomial over n variables $x = (x_1, \ldots, x_n)$. Since both polynomials are of degree at most d and since the field is larger than d, the above equality implies that these polynomials are equivalent as formal multivariate polynomials over $\mathbb{F}$. This means that the corresponding partial derivatives are equivalent as polynomials. Let us compute the gradient (i.e., the row vector of all partial derivatives) of both sides of (1). For the LHS we use the chain rule, and for RHS we simply get the ith unit vector e_i,

$$\partial D(A(x)) \cdot \mathcal{J}A(x) = e_i \tag{2}$$

This equation holds for all input databases, and so by plugging in $\hat{x}$, we get that e_i is in the row-span of $M = \mathcal{J}A(\hat{x})$, as required. □

We record the following extensions of the lemma.

Remark 3.2 (multiple servers)**.** *Lemma 3.1 solely relies on the correctness property of the PIR and is independent of the privacy condition. Consequently, we can apply the lemma on a multi-server PIR by treating the answers of the all the servers as an answer of a single server (in a virtual non-private PIR scheme). This means that the matrix* M_q *now depends on the vector* $q = (q_1, \ldots, q_k)$ *of all the queries.*

[12] This can be done efficiently, since the circuit $\mathcal{J}A(x)$ can be computed efficiently based on $A(x)$ in a gate-by-gate manner via the Bauer-Strassen algorithm (see, e.g., [61]).

Remark 3.3 (extension to imperfect correctness)**.** *Lemma 3.1 can be extended to imperfect PIR schemes as follows. We say that a query q is* correct for i *(or i-correct) if there exists a randomness r for which $q = \text{QUERY}_n^{\mathbb{F}}(i; r)$ and $\text{DEC}_n^{\mathbb{F}}(i, r, \text{ANS}_n^{\mathbb{F}}(x, q)) = x[i]$ for at least $p = 0.99$ fraction of the x's. (The value p can be replaced by any constant or even by $1 - 1/\operatorname{poly}(n)$.) Then Lemma 3.1 holds for every i-correct query assuming that $d < p|\mathbb{F}|$. The proof is identical except that Eq. (1) holds over a fraction of at least p of the x's. Since both sides of the equation are polynomials of degree d, we can conclude by the Schwartz-Zippel Lemma that the equality holds for* all *inputs $x \in \mathbb{F}^n$, and the rest of the analysis remains the same. (Note that for perfectly-correct PIR every q in the support of $\text{QUERY}_n^{\mathbb{F}}(i)$ is i-correct.)*

Lemma 3.1 implies that a low-degree $\mathbb{F}$-arithmetic PIR can only have trivial answer length.

Theorem 3.4. *Every perfectly-correct (resp., statistically-correct) single server arithmetic PIR over a field $\mathbb{F}$ with degree $d < |\mathbb{F}|$ (resp., $d < 0.99|\mathbb{F}|$) has a trivial answer length of $\ell(n) \geq n$. Consequently, this also applies to every single-server PIR with finite degree.*

Put differently, such PIRs cannot beat the trivial solution in which the server simply sends the entire database to the client.

Proof. We begin with the first part of the theorem. Let n be the database size and let $(\text{QUERY}_n^{\mathbb{F}}, \text{ANS}_n^{\mathbb{F}}, \text{DEC}_n^{\mathbb{F}})$ be the arithmetic circuits of a single-server $\mathbb{F}$-arithmetic PIR with degree d. We either assume that (A) $d < |\mathbb{F}|$ and the scheme is perfectly-correct or (B) that the scheme has negligible correctness error and $d < 0.99|\mathbb{F}|$. Let ϵ denote the probability that a random query q that is sampled from $\text{QUERY}^{\mathbb{F}}(i)$ is *not correct* for i (as per Remark 3.3). Observe that if the scheme is perfectly correct then $\epsilon = 0$, and if the scheme is imperfect with negligible error, then ϵ is negligible in n. To see this, fix i and let $q_r := \text{QUERY}^{\mathbb{F}}(i; r)$ and call r good for x (over i) if

$$\text{DEC}_n^{\mathbb{F}}(i, r, \text{ANS}_n^{\mathbb{F}}(x, q_{i,r})) = x[i].$$

By correctness, for every x, a randomly-chosen r is correct except with negligible probability, and therefore by an averaging argument, except with negligible probability, a randomly-chosen r is good for all but negligible fraction of the x's.

Let $x \in \mathbb{F}^n$ be a database of size n, and let t be the query length and ℓ the answer length. Assume towards a contradiction that $\ell < n$. We will show how to break the security of the PIR via the following adversary $\mathcal{A}$:

1. Sample a pair of indices $(i_0, i_1) \in [n]^2$ and send to the challenger the tuple $(\mathbb{F}, i_0, i_1)$
2. Given a query q from the challenger, compute the matrix $M = \mathcal{M}(q)$ based on the algorithm from Lemma 3.1. If both or neither e_{i_0}, e_{i_1} are in the row-span of M output a random bit b'. Otherwise, output the bit b' for which $e_{i_{b'}}$ is spanned by M.

Fix the challenger b and the index $i = i_b$ and let $j = i_{1-b}$ be a random index. By Lemma 3.1 (or Remark 3.3 in the case of imperfect scheme), the matrix M is an $\ell \times n$ matrix and so its span contains at most ℓ different unit vectors (being a matrix of rank ℓ). In addition, $M = \mathcal{M}(q)$ is statistically independent of the random index j that was not chosen by the challenger. It follows that

$$\Pr_j[e_j \in \text{rowspan}(M)] \leq \ell/n.$$

The adversary $\mathcal{A}$ deterministically loses the game if q is not i-correct and j is in the span which happens with probability at most $\epsilon \cdot (\ell/n)$. Otherwise, the adversary is forced to guess and loses with probability

0.5 in two scenarios: the query q not i-correct and e_j is not in the span (probability at most $\epsilon(1-\ell/n)$), or, q is i-correct but e_j is in the span (probability at most $(1-\epsilon)\ell/n$). Summing these cases, the total loss probability is at most $0.5\epsilon + \ell/2n$. Consequently, since $\ell \leq n-1$, the winning probability is at least $\frac{1}{2} + \frac{1}{2n} - \frac{\epsilon}{2}$. Since ϵ is negligible, the adversary wins with a non-negligible advantage.

To prove the "Consequently" part, denote by d the degree of the given arithmetic PIR. The adversary chooses a finite field $\mathbb{F}$ such that $|\mathbb{F}| > d$. Since the degree is at most exponential in the circuit-size $s = \mathrm{poly}(n)$ of the PIR algorithms, this can be done in time $\mathrm{poly}(n)$ by selecting a sufficiently large prime with bit length of $\mathrm{poly}(n)$. Given such a field (and its natural implementation), and apply the same attack as before. □

Remark 3.5 (Thm. 3.4 is tight.)**.** *The above attack breaks any perfectly correct arithmetic scheme with download complexity ℓ, with winning probability at least $1 - \frac{\ell}{2n}$. This bound is tight, as demonstrated by the following trivial scheme: the client samples a random ℓ-subset $S \subset [n]$ that contains the requested index i, and sends it to the server, which responds with the ℓ field elements $(x_j)_{j \in S}$.*

3.1 A Recovery Attack

In the previous section we described a distinguishing attack against PIR. In this section we show that, in some cases, it is possible to efficiently extract the index i from the query q. This is especially effective in the case of *block-based PIR*.

A block-based PIR partitions the database x into b-size blocks $B_1, \ldots, B_m$, where $m = n/b$, such that a query for an entry in block B_i fetches the ith block. This feature is common in practical PIR constructions, including the code-based PIR schemes that will be attacked in the next section. In this setting, Lemma 3.1 implies that if the query q lies in the support of $\textsc{Query}_n^{\mathbb{F}}(i)$, then the rows of the matrix M_q span *all* the unit vectors in $V_i = \{e_j : j \in B_i\}$.

We can therefore try to recover the block index by checking, for each candidate i, whether M_q spans V_i, and keeping only the indices that pass the test. When the scheme has good download rate, i.e., not much larger than the block-size b, we can hope to obtain a single index i that passes the test. In that case, we know for sure that i is the correct index. Jumping ahead, our experiments show that this is the case for all parameter regimes of the code-based schemes.

This attack checks whether n "target" vectors $e_1, \ldots, e_n$ lie in the row-span of an $\ell \times n$ matrix M_q. A naive implementation requires $n \cdot O(\ell^2 n) = O(\ell^2 n^2)$ arithmetic operations. This can be reduced to $O(\ell n^2)$ operations by (1) computing the (reduced) column echelon form (CEF) of the matrix M_q, applying the same column operations to the target vectors $e_1, \ldots, e_n$, and (2) verifying that the transformed target vectors vanish on the zero columns. The first step costs $O(\ell^2 n + n^2 \ell) = O(n^2 \ell)$ arithmetic operations and the second step requires additional $O(n)$ operations per target vector.

In our implementations, we apply two additional optimizations. First, for each block B_i we test only whether the characteristic vector $1_{B_i} \in \mathbb{F}^n$ is spanned by M_q, rather than testing all vectors in V_i. Second, when the communication length ℓ is significantly smaller than the database size n, we reduce the ambient dimension from n to $n' = (1 + O(1))\ell$ using a standard sketching technique. Specifically, we multiply M_q by a random sparse $n \times n'$ matrix S, in which each row contains a constant number of non-zero entries. The same linear transformation is applied to the test vectors $1_{B_1}, \ldots, 1_{B_m}$. After applying the sketching step (whose cost is $O(n\ell + nm)$), we execute the CEF-based attack over the sketched matrix and vectors with an additional cost of $O(\ell(\ell + m)n')$. Assuming that $m < \ell$, the overall arithmetic complexity is reduced to $O(n\ell + \ell^3)$.

These optimizations lose some information and, although the error remains one-sided, they may introduce additional false positives. Nevertheless, when applied to the code-based schemes, the attack remains effective (we obtain a single candidate). Moreover, these optimizations yield a significant speed-up, especially when the matrices and the underlying field are large.[13] Indeed, over fields of size $\geq 2^{16}$ and matrices of dimensions $n \in [13K, 26K]$ and $\ell \in [500, 1200]$, we observed an approximately one-order-of-magnitude speed-up when using a random sketching matrix with a single non-zero entry per row and $n' = 1.2\ell$. (See Table 1 for more details.)

4 Attacking Existing Code-Based PIR Schemes

In this section, we apply the recovery attack (Section 3.1) to three constructions of code-based PIRs from [44, 43]. To maintain consistency with the original papers, the symbol n is utilized throughout this section to represent codeword length whereas the database length will be taken to be the product of two auxiliary parameters m, δ.

4.1 Overview

We review the PIR schemes of Holzbaur et al. (HHW) [44] and its follow-ups. While the original description is somewhat complicated, we note that all these schemes are not only arithmetic, but also *linear* in the following sense.

Let m, δ, ℓ be some integer-valued parameters whose value will be determined later. The database $X \in \mathbb{F}^{m\delta}$ is represented as a row vector with m blocks (aka files) $X_1, \ldots, X_m \in \mathbb{F}^{\delta}$ of size δ each. The query algorithm maps an index $i \in [m]$ and some private randomness r to a matrix $Q \in \mathbb{F}^{m\delta \times \ell}$ and the answer algorithm outputs the product $X \cdot Q \in \mathbb{F}^{\ell}$. Finally, the decoding algorithm extracts the desired block X_i from the vector XQ_i by applying some linear operation $L_{r,i}$ (that depends on the private randomness r).

Since both the decoding and recovery are linear, the degree of the system is 1 and it is vulnerable to our attacks with respect to *every* finite field $\mathbb{F}$. Indeed, we break this scheme and its successors over all the finite fields that were suggested by the authors. See Section 4.5 and Table 1.

The basic scheme has asymmetric communication with query length of $t = m\delta\ell$ (the entire matrix) and an answer length of ℓ which will be taken to be only slightly larger than the length of a single data item (roughly $\ell = \delta \cdot \lambda$ where λ is the security parameter). The rate of the scheme can be improved via a standard balancing technique (e.g., [25]). This optimization will be ignored here since it does not affect the attack.[14]

4.2 The HHW PIR

Set-up and parameters. Recall that the query generates a matrix $Q \in \mathbb{F}^{m\delta \times \ell}$. The dimension ℓ will be taken to be the product $\ell = sn$ of two integer-valued parameters s and n. Rows in $\mathbb{F}^{\ell}$ will be often parsed as

[13] The arithmetic complexity of the attack is independent of the field; however, the cost of individual arithmetic operations increases with the field size.

[14] Specifically, the database is assumed to contain m files $X_1, \ldots, X_m$ where each file has L rows and δ columns. Then, we can parse each row of the database as a single database of size $m\delta$ and apply the basic PIR with a single query for the ith file and L answers that are computed for the same query with respect to each database. This optimization is presented in [44] as part of the basic system via the use of an additional parameter L. Since the value of L does not affect the security (only the rate), we focus here on $L = 1$ for the sake of clarity. Indeed, this optimization does not change the query computation which is the part that will be attacked here.

n-long vectors over $\mathbb{F}_s$ — the extension field of degree s over $\mathbb{F}$. By abuse of notation, we sometimes view $\mathbb{F}_s$ as an $\mathbb{F}$-linear subspace. The scheme also employs several additional subspaces over $\mathbb{F}^{sn}$.

Let $v < s$ and $k < n$ be integer-valued parameters and take $\delta \leq (n-k)(s-v)$ to be the block-length of the database. (Roughly speaking, think of kv as the security parameter.) For a basis $\Gamma = \{\gamma_1, ..., \gamma_s\}$ for $\mathbb{F}_s$ over $\mathbb{F}$, let V denote the subspace of $\mathbb{F}_s$ that is spanned by the first v vectors of Γ i.e., $\{\gamma_1, ..., \gamma_v\}$, and let W denote the orthogonal subspace that is spanned by the last $s-v$ vectors of Γ i.e., $\{\gamma_{v+1}, ..., \gamma_s\}$.

The scheme also employs a random linear MDS code $\mathcal{C}$ of length n and dimension k over $\mathbb{F}_s$. The code can be made public although our attacks apply even if the code is kept private. Since this is an MDS code, for any k-subset (typically secret) $I \subset [n]$, we can decompose the coordinate set $[n]$ into an "information" set I and a "noise" set $\bar{I} = [n] \setminus I$. We will often generate a vector $b \in \mathbb{F}_s^n$ as a sum of a codeword $c \in \mathbb{F}_s^n$ and a noise vector χ whose support is on the set $\bar{I}$. Given the vector $q = c + \chi$ and the set I it is possible to recover the noise vector χ via a linear operation $\mathcal{R}_I(q)$. We will typically construct the noise vector $\chi \in \mathbb{F}_s^n$ by placing a short vector $u \in \mathbb{F}_s^{n-k}$ in the $\bar{I}$ coordinates, and denote this linear operation by $\phi_{\bar{I}}(u)$. We naturally extend the operation R_I (resp., $\phi_{\bar{I}}$) to matrices by applying it separately to each row.

Query: The query $\text{QUERY}_{i,m}^{\mathbb{F}}(r)$ algorithm interprets the randomness r as a tuple $(D, I, \Gamma, E, \Delta)$, where:

1. $D \in \mathbb{F}_s^{m\delta \times n}$ is a matrix whose rows are codewords sampled uniformly at random from $\mathcal{C}$.
2. $I \subset [n]$ is a random information set of size k and we denote its complement by $\bar{I} = [n] \setminus I$.
3. $\Gamma = \{\gamma_1, \ldots, \gamma_s\}$ is a basis of $\mathbb{F}_s$ over $\mathbb{F}$, let $V = \langle \gamma_1, \ldots \gamma_v \rangle_{\mathbb{F}}$ and $W = \langle \gamma_{v+1}, \ldots \gamma_s \rangle_{\mathbb{F}}$ denote the corresponding vector subspaces.
4. $E_0 \in \mathbb{F}_s^{m\delta \times (n-k)}$ is a matrix whose entries are sampled uniformly at random from V, and we set $E = \phi_{\bar{I}}(E_0) \in \mathbb{F}_s^{m\delta \times n}$.
5. $\Delta_0 \in \mathbb{F}_s^{\delta \times (n-k)}$ whose entries are sampled from W, is a full row rank matrix when expanded as a $\delta \times s(n-k)$ matrix over $\mathbb{F}$, and we set $\Delta = \phi_{\bar{I}}(\Delta_0) \in \mathbb{F}_s^{\delta \times n}$. Accordingly, Δ_0 has a right inverse $\Delta_0^{-1} \in \mathbb{F}^{(n-k)s \times \delta}$ over $\mathbb{F}$. (This inverse will be used later for decoding.)

Finally, for the desired file $i \in [m]$, compute the *index-encoding matrix* $M = e_i \otimes \Delta \in \mathbb{F}_s^{m\delta \times n}$ (i.e., the block matrix obtained by placing Δ in the i-th block position and zeros elsewhere) and output

$$Q = D + E + M \in \mathbb{F}^{m\delta \times ns}.$$

Answer: The answer algorithm $\text{ANS}_m^{\mathbb{F}}(X, Q)$ outputs $\mathbf{ans} = X \cdot Q \in \mathbb{F}^{ns}$.

Decode: The decoding algorithm $\text{DEC}_{i,m}^{\mathbb{F}}(r, \mathbf{ans})$ parses the randomness r as a tuple $(D, I, \Gamma, E, \Delta)$, then it recovers the "noisy" part by computing $B = \mathcal{R}_I(\mathbf{ans}) \in \mathbb{F}^{(n-k)s}$, projects each entry of B to the linear sub-space W and gets the row vector $B' \in \mathbb{F}^{(n-k)s}$. Finally, output the δ-long vector $B' \cdot \Delta_0^{-1}$.

Arithmetic structure and Linearity. We assume that all randomly dependent parameters, including matrices and their inverses, are precomputed given the randomness r, and so the query generation and decoding algorithms do not need to perform matrix inversions internally. Consequently, these schemes avoid division operations and are treated as strictly arithmetic circuits. Furthermore, for a fixed randomness r, both the answer algorithm and the decoding algorithm are linear.

Existing attacks. In [19], the scheme was shown to be insecure due to rank differences in submatrices of the query. Concretely, for a query Q^i corresponding to file i, let $j \in [n]$ and let $Q^i[j]$ denote the submatrix of Q^i obtained by deleting rows $\{(j-1)\delta + 1, j\delta\}$ from Q^i. The authors observed that one can easily distinguish between the cases $i = j$ and $i \neq j$. Specifically, for every $i \in [m]$, we have $\mathrm{rank}(Q^i[i]) \leq ns - \delta$, while with a high probability for $j \neq i$, we have $rank(Q^i[j]) > ns - \delta$.

Our attack. While the attack of [19] exploits a very specific rank difference in the query sub-matrices, this phenomenon is only a symptom of the arithmetic properties of the scheme. Indeed, by applying our recovery attack, we break the scheme and recover the index i for all parameter settings suggested in the original HHW paper. Furthermore, our experiments show that our recovery attack remains decisive (retaining only one candidate for i) whenever the download cost ns is smaller than the database size $m\delta$.

4.3 CB-cPIR

To address the attack demonstrated against the HHW scheme, a new PIR scheme was proposed in [43] focusing on fixing the sub-matrix rank differences. The idea is simple: Given an index i, additively share the unit vector $e_i \in \mathbb{F}^m$ into a pair of vectors $v_1, v_2 \in \mathbb{F}^m$ and apply a pair of HHW-queries one with respect to v_1 and one with v_2.

Specifically, the client samples two MDS codes $\mathcal{C}_1, \mathcal{C}_2$ and two independent random tapes r_1 and r_2 that specify two independent HHW tuples

$$(D_1, I_1, \Gamma_1, E_1, \Delta_1), \quad \text{and} \quad (D_2, I_2, \Gamma_2, E_2, \Delta_2).$$

The client then computes the index-encoding matrices $M_1 = v_1 \otimes \Delta_1$ and $M_2 = v_2 \otimes \Delta_2$ and generates two query matrices

$$Q_1 = D_1 + E_1 + M_1, \quad Q_2 = D_2 + E_2 + M_2.$$

The server answers these queries with $\mathbf{ans}_1 = X \cdot Q_1, \mathbf{ans}_2 = X \cdot Q_2$, and the client applies the HHW decoding algorithm to $\mathbf{ans}_1$ and $\mathbf{ans}_2$ and subtracts the resulting vectors.

Existing attacks. The CB-cPIR was designed specifically to eliminate the rank-based attack of [19]. However, a different attack was discovered by [50]. This attack takes advantage of the repeated use of the same matrices Δ_1 and Δ_2, by constructing a low rank auxiliary matrix and appending specially crafted rows while measuring rank differences ultimately recovers the client's index. Along the way, the attack iterates over all possible field elements, and it performs $\left\lceil \frac{|\mathbb{F}|}{\delta - 1} \right\rceil$ rank computations for $ns \times ns$ matrices over $\mathbb{F}$. The overall complexity is therefore linear in the field size, which makes it feasible only for small fields.

Our attack. From an arithmetic point of view, the above modification changes nothing and certainly does not address the main source of insecurity. Indeed, the scheme is still arithmetic and has degree 1, and so we can still apply our attack over any field. We apply our recovery attack and observe similar performance to its performance on the HHW scheme. The attack recovers the index whenever the download cost $2ns$ is smaller than the database size $m\delta$ (i.e., for every non-trivial rate). Moreover, in contrast to the attack of [50], the arithmetic complexity is independent of the field size, and it remains effective even for large fields that were previously assumed to be secure.

4.4 CB-cPIR version 2

In response to the attack of [50], the CB-cPIR authors added a new variant to the paper [43, Remark 6]. The scheme is identical to the previous one except that we inject more randomness to the index-encoding matrices M_1 and M_2.

In particular, in addition to the HHW tuples $(D_i, I_i, \Gamma_i, E_i, \Delta_i)$, $i \in \{1, 2\}$, sample random matrix $S \in \mathbb{F}^{m\delta \times \delta}$, parse it as m block matrices $S_1, \ldots, S_m \in \mathbb{F}^{\delta \times \delta}$, and set

$$M_1 = \begin{bmatrix} S_1\Delta_1 \\ \vdots \\ S_m\Delta_1 \end{bmatrix}, \quad M_2 = \begin{bmatrix} S_1\Delta_2 \\ \vdots \\ (S_i + I_{\delta\times\delta})\Delta_2 \\ \vdots \\ S_m\Delta_2 \end{bmatrix}.$$

The rest of the scheme is identical to the previous version. (That is, $Q_i = D_i + E_i + M_i$, $\mathbf{ans}_i = X \cdot Q_i$, $i \in [m]$, and the decoding algorithm applies the HHW decoder to $\mathbf{ans}_1$ and $\mathbf{ans}_2$ and subtracts the resulting vectors.)

Our attack. This countermeasure increases the complexity of the attack from [50] as it introduces additional variables that needs to be guessed. However, it has no effect on the arithmetic attack. Indeed, our recovery attack, behaves exactly as it does on the CB-cPIR scheme. In particular, the rank tests remain decisive, and the client's index is recovered whenever the download cost $2ns$ is smaller than the database size $m\delta$ (i.e., for every non-trivial rate). To the best of our knowledge, this is the first attack that breaks this scheme in any setting of parameters.

4.5 Attack parameters and runtime results

We implemented the attack for all schemes in SageMath and Python on a standard workstation (Intel Core i7-14700KF, 3.4GHz, 64GB RAM). As shown in Table 1, we broke the schemes for all the parameters suggested in the original works. (For which the original estimated security level ranged between 96 bits to 128 bits depending on the exact setting of parameters; see Table 2 in [43].)

While the number of arithmetic operations is independent of the underlying field, the actual cost of each such operation grows polylogarithmically with the field size. For small fields the attack completes within a few seconds, whereas for large fields it may require several minutes. The experiments were carried out using the smallest database size for which the attack succeeds. Naturally, the execution time increases as the file count m grows. To ensure the attack remains practical for larger fields, we additionally employed sketching techniques to reduce the dimension of the query matrix as explained in Section 3.1. The code for our attack can be found in https://github.com/arithmetic-pir/breaking-arithmetic-pir

5 Two-Server Arithmetic PIR

In this section we provide a linear-algebraic characterization of 2-server arithmetic PIR by showing that any arithmetic PIR can be converted into a fully-linear PIR (Section 5.1). We then present a construction of such PIR with information theoretic security and non-trivial communication of $O(n^{1/3})$ (Section 5.2). For simplicity, we restrict our attention in this section to perfectly correct PIR. (All the result can be extended to PIR with negligible correctness error.)

scheme	m	δ	$\lvert\mathbb{F}\rvert$	s	v	n	k	time [seconds]
HHW	65	50	2^4	32	31	100	50	2
HHW	65	200	2^4	32	16	100	50	14
HHW	65	50	2^4	32	31	100	50	5
HHW	65	200	2^5	32	24	100	50	13
HHW	65	50	2^5	32	31	100	50	2
HHW	65	200	2^5	32	21	100	50	24
HHW	65	200	2^6	32	21	100	50	24
CB-cPIRv1	129	100	2^5	32	30	100	50	23
CB-cPIRv1	129	50	2^5	32	31	100	50	11
CB-cPIRv1	129	100	2^{16}	12	10	100	50	86
CB-cPIRv1	129	120	$2^{32}-5$	6	4	120	60	263
CB-cPIRv1	129	100	2^{32}	5	3	100	50	135
CB-cPIRv1	129	200	$2^{61}-1$	6	2	100	50	167
CB-cPIRv2	129	50	2^5	32	31	100	50	8
CB-cPIRv2	129	100	2^5	32	30	100	50	25
CB-cPIRv2	129	100	2^{16}	12	10	100	50	83
CB-cPIRv2	129	120	$2^{32}-5$	6	4	120	60	267
CB-cPIRv2	129	100	2^{32}	5	3	100	50	134
CB-cPIRv2	129	200	$2^{61}-1$	6	2	100	50	176

Table 1: Parameter sets used in the attack. Recall that the block size is δ, the number of blocks in the database is m, and the download cost is $\ell = sn$.

5.1 From arithmetic PIR to linear-PIR

Theorem 5.1 (Arithmetic-PIR $\implies$ Linear-PIR)**.** *Suppose that $\mathcal{P} = (\text{QUERY}, \text{ANS}, \text{DEC})$ is a two-server $\mathbb{F}$-arithmetic PIR with perfect (resp., statistical, computational) security and degree $d < |\mathbb{F}|$.*

Then there exists a two-server $\mathbb{F}$-arithmetic PIR $\mathcal{P}' = (\text{QUERY}', \text{ANS}', \text{DEC}')$ with perfect (resp., statistical, computational) security that is linear *in the sense that: (1) the answer algorithm is linear in the database; and (2) the decoding algorithm is linear in the answers of the servers. Furthermore, $\mathcal{P}'$ preserves the lengths of the queries and the answers (aka upload and download cost) of the original scheme.*

The theorem and its proof extend to any number of servers.

Proof. Let $x \in \mathbb{F}^n$ be a database of size n, and let t be the query length and ℓ the answer length. Let $(\text{QUERY}_n^{\mathbb{F}}, \text{ANS}_n^{\mathbb{F}}, \text{DEC}_n^{\mathbb{F}})$ be the arithmetic circuits of a two-server arithmetic PIR with degree $d < |\mathbb{F}|$.

For a fixed query q denote the circuit $A_q(x) = \text{ANS}_n^{\mathbb{F}}(x, q)$ and let $M_q \in \mathbb{F}^{\ell \times n}$ denote the Jacobian $\mathcal{J}A_q(\mathbf{0})$ evaluated at the all-zero database $\mathbf{0} \in \mathbb{F}^n$. By the multi-server variant of Lemma 3.1 (see Remark 3.2), for every index $i \in [n]$ randomness r, and queries $(q_1, q_2) = \text{QUERY}_n^{\mathbb{F}}(i; r)$, the rows of the matrices M_{q_1} and M_{q_2} jointly span the i-th standard basis vector e_i. Let us denote by $v_1 = v_1(i, r), v_2 = v_2(i, r) \in \mathbb{F}^\ell$ the row vectors for which $v_1 \cdot M_{q_1} + v_2 \cdot M_{q_2} = e_i$.

We construct a new arithmetic PIR scheme with linear answer as follows:

1. The query circuit runs the original query outputs $(q_1, q_2) = \text{QUERY}_n^{\mathbb{F}}(i; r)$.
2. The jth server's answer circuit given the query q_j, computes $M_{q_j} \in \mathbb{F}^{\ell \times n}$, and outputs $\mathbf{ans}_j = M_{q_j} \cdot x$

where $x \in \mathbb{F}^n$ is viewed as a column vector. (Recall that M_{q_j} can be computed arithmetically by evaluating the Jacobian $\mathcal{J}A_{q_j}(\mathbf{0})$).)

3. The decoding circuit given the answers and the original queries, computes $v_1 = v_1(i,r), v_2 = v_2(i,r) \in \mathbb{F}^\ell$ and outputs $v_1 \cdot \mathbf{ans}_1 + v_2 \cdot \mathbf{ans}_2 \in \mathbb{F}$.

Note that the decoder is arithmetic since $v_1 = v_1(i,r), v_2 = v_2(i,r)$ can be computed by computing M_{q_1} and M_{q_2} and then solving a system of linear equations. Furthermore, for fixed randomness and fixed index i, the decoder computes a linear function in the answers.

The correctness of the scheme is immediate, and the privacy follows from the privacy of the original scheme (since the query algorithm has remained unchanged). □

Theorem 5.2 (extension to PIR with zero-check gates)**.** *Theorem 5.1 remains valid when the degree condition on $\mathcal{P}$ is replaced by the assumption that $\mathcal{P}$ has circuit size s and is $\mathbb{F}$-arithmetic over a prime-order field $\mathbb{F}$ with $|\mathbb{F}| > 10s2^{2s}$. Under this assumption, $\mathcal{P}'$ is a 2-server linear PIR scheme whose privacy error (or, alternatively, correctness error) is at most $\epsilon = 4s2^{2s}/|\mathbb{F}|$ larger than that of $\mathcal{P}$.*

Proof. The proof is similar to the proof of Theorem 5.1 except that we use the zero-check removal procedure $\mathcal{T}$ from Definition A.1 as follows.

Take the circuit $A_q(x)$ as defined above, sample a random database $\hat{x} \xleftarrow{R} \mathbb{F}^n$ and consider the zerocheck-free circuit $A'_{q,\hat{x}} := \mathcal{T}(A_q, \hat{x})$. Let $\mathcal{J}A'_{q,\hat{x}}(x) : \mathbb{F}^n \to \mathbb{F}^{\ell \times n}$ be the *formal Jacobian* matrix of $A'_{q,\hat{x}}(x)$, i.e., $\mathcal{J}A'_{q,\hat{x}}(x)$ is a matrix of rational functions whose (i,j)-th entry is the formal partial derivative of the i-th output of $A'_{q,\hat{x}}(x) : \mathbb{F}^n \to \mathbb{F}^\ell$ with respect to the j-th input (the formal variable x_j). Compute the matrix of field elements $M_{q,\hat{x}} = \mathcal{J}A'_{q,\hat{x}}(\hat{x}) \in \mathbb{F}^{\ell \times n}$. The construction of the linear PIR is now identical to the construction in Theorem 5.1, except that the randomness $\hat{x}$ is sampled by the client and sent to the servers together with q_1 and q_2.

The analysis is similar to Theorem 5.1. The crucial observation is that, by the multi-server variant of Lemma A.4 (see Remark 3.2), for every index $i \in [n]$, randomness r, and queries $(q_1, q_2) = \text{QUERY}_n^{\mathbb{F}}(i; r)$, except with probability ϵ over the choice of $\hat{x}$, the rows of the matrices $M_{q_1,\hat{x}}, M_{q_2,\hat{x}}$ jointly span the i-th standard basis vector e_i. Note that this variant increases the correctness error by ϵ and does not affect the privacy. Alternatively, we can let client sample random $\hat{x}$ conditioned on the event that $M_{q_1,\hat{x}}$ and $M_{q_2,\hat{x}}$ jointly span e_i. This eliminates the correctness error, but leaks some information. In particular, since $\hat{x}$ can be simulated up to a statistical distance at most ϵ (by sampling a random vector), the privacy error grows by at most ϵ. □

Theorems 5.1 and 5.2 apply to (field agnostic) arithmetic PIR $\mathcal{P}$. In this case, the modified PIR $\mathcal{P}'$ will be linear only for fields that are larger than the degree of $\mathcal{P}$ (resp., sufficiently larger than the circuit size of $\mathcal{P}$). This suffices to prove the following equivalence theorem (stated as Informal Theorem 1.4 in the introduction).

Theorem 5.3 (Arithmetic-PIR is Linear-PIR)**.** *The following are equivalent.*

1. *There exists a two-server arithmetic PIR with perfect security and finite degree that communicates $O(\ell(n))$ field elements for a database of size n.*
2. *For every sufficiently large field $\mathbb{F}$, there exist n pairs of distributions*

$$(\mathcal{A}_1, \mathcal{B}_1), \ldots, (\mathcal{A}_n, \mathcal{B}_n)$$

over $O(\ell(n)) \times n$ matrices such that:

(a) The rows of $(\mathcal{A}_i, \mathcal{B}_i)$ jointly span the unit vector e_i.

(b) For every $i, j \in [n]$, the marginal distributions $\mathcal{A}_i$ and $\mathcal{A}_j$ (resp., $\mathcal{B}_i$ and $\mathcal{B}_j$) are identical.

(c) The matrices $\mathcal{A}_i, \mathcal{B}_i$ can be sampled via a two step process: first a short description (q_1, q_2) of length $O(\ell)$ field elements is sampled via a circuit $S_i^{\mathbb{F}}$ and then the matrices are computed based on q_1 and q_2 via some deterministic extraction process $E^{\mathbb{F}}$ that is independent of i. Both circuits have black-box access to the field $\mathbb{F}$ (and do not use division gates/zero-check gates).[15]

Furthermore, the equivalence holds if in the first item we relax perfect privacy to statistical privacy (resp., perfect correctness to statistical correctness) and relax item 2b to statistical indistinguishability (resp., relax item 2a to hold except with negligible probability).

Proof. We begin by showing that (1) implies (2). Let $\mathcal{P} = (\text{QUERY}, \text{ANS}, \text{DEC})$ be a two-server arithmetic PIR scheme with perfect (resp., statistical) security with total communication cost $O(\ell(n))$ and finite degree d. For every fixed field $\mathbb{F}$ of cardinality larger than d, we can apply Theorem 5.1, and transform $\mathcal{P} = (\text{QUERY}, \text{ANS}, \text{DEC})$ into a linear arithmetic PIR $\mathcal{P}_L = (\text{QUERY}_L, \text{ANS}_L, \text{DEC}_L)$ that preserves the query and answer lengths. Recall that the $\text{QUERY}_L = \text{QUERY}$ and that $\text{ANS}_L(x, q) = M_q \cdot x$. Furthermore, note that the mapping from q to M is computed via an $\mathbb{F}$-arithmetic circuit $C^{\mathbb{F}}(q) = \mathcal{J}A_q(\mathbf{0})$ where $A_q(x) = \text{ANS}_n^{\mathbb{F}}(x, q)$.

For any index $i \in [n]$, we define the joint distribution $(\mathcal{A}_i, \mathcal{B}_i)$ by sampling the queries $(q_1, q_2) \xleftarrow{R} \text{QUERY}_{i,n}^{\mathbb{F}}(r)$ and output the matrix pair $(M_{q_1}, M_{q_2}) = C^{\mathbb{F}}(q_1), C^{\mathbb{F}}(q_2)$. The perfect correctness and perfect privacy of the linear-PIR imply that (2a) and (2b) hold.[16] Item 2c follows by noting that q consists of $O(\ell)$ field elements, and that the matrices can be sampled by composing the field agnostic circuits $\text{QUERY}^{\mathbb{F}}$ and $C^{\mathbb{F}}$.

We move on and show that (2) implies (1). Let $x \in \mathbb{F}^n$ be a database of size n. Given n pairs of distributions $\{(\mathcal{A}_j, \mathcal{B}_j)\}_{j \in [n]}$ over $O(\ell(n) \times n)$ satisfying the conditions we construct the following PIR scheme.

1. Query: The randomized $\text{QUERY}_{i,n}^{\mathbb{F}}(r)$ circuit uses its randomness r to sample a pair of matrix identifiers q_1, q_2 by invoking $S_i^{\mathbb{F}}(r)$.

2. Answer: The deterministic $\text{ANS}_n^{\mathbb{F}}(x, q)$ extracts a matrix $M_q = E^{\mathbb{F}}(q)$ and generates the output $\mathbf{ans} = M_q \cdot x$.

3. Decode: The deterministic circuit $\text{DEC}_{i,n}^{\mathbb{F}}(i, r, \mathbf{ans}_1, \mathbf{ans}_2)$ computes $(q_1, q_2) = \text{QUERY}_{i,n}^{\mathbb{F}}(r)$ and $M_1 = E^{\mathbb{F}}(q_1), M_2 = E^{\mathbb{F}}(q_2)$, recovers a vector $v = (v_1, v_2)$ so that $v_1 \cdot M_1 + v_2 \cdot M_2 = e_i$ and outputs $v_1 \cdot \mathbf{ans}_1 + v_2 \cdot \mathbf{ans}_2$. (This is well defined by item 2a.)

It is not hard to verify that the resulting PIR is perfectly-correct perfectly-private arithmetic linear 2-server PIR with communication of $O(\ell(n))$.[17] □

Remark 5.4 (extension)**.** *It is possible to partially extend Thm. 5.3 to the case where the PIR makes use of division gates/zero-check gates (i.e., does not have finite degree). For this, we follow the same proof except*

[15] Note that this implies that the support of $(\mathcal{A}_i, \mathcal{B}_i)$ has size at most $|\mathbb{F}|^{O(\ell(n))}$.

[16] More generally, imperfect correctness with an error ϵ and statistical privacy imply that (2a) holds w/p $1 - \epsilon$ and (2b) holds with negligible statistical distance.

[17] More generally, if (2a) holds w/p $1 - \epsilon$ we get correctness error of ϵ and if (2b) holds up to negligible statistical distance we get statistical privacy.

that Thm. 5.1 is replaced with Thm. 5.2. It can be verified that for every sufficiently large n and every field $\mathbb{F}$ that is sufficiently large (e.g., larger than $2^{p(n)}$ for some polynomial $p(\cdot)$), the resulting matrices satisfy items 2a except with negligible probability as well as items 2b and 2c.

5.2 PIR on the Line

The initial constructions of information-theoretic PIR schemes, including [25, 14, 63], can all be shown to arithmetize. We demonstrate this for the 2-server construction of Woodruff and Yekhanin [63]. This extends naturally to the t-private k-server variant of [63].

Let $\mathbb{F}$ be a finite field of size $|\mathbb{F}| > 2$.[18] Let $n \in \mathbb{N}$ be the size of the database and let $m = O(n^{\frac{1}{3}})$ such that $n \leq \binom{m}{3}$. Let $x \in \mathbb{F}^n$ denote a database of size n and such that each element $x_{i,j,k}$ corresponds to a specific index triplet (i, j, k). We define the following degree-3 polynomial

$$F_x(z_1, ..., z_m) = \sum_{\substack{i,j,k \in [m] \\ i<j<k}} x_{i,j,k} \cdot z_i z_j z_k$$

and let $\lambda_1, \lambda_2 \in \mathbb{F}_q$ be distinct and non-zero.

1. The query algorithm $\text{QUERY}^{\mathbb{F}}_{(i,j,k),m}(r)$ interprets the randomness r as a uniformly random vector $v \in \mathbb{F}^m$ and outputs the query tuple

   $$(q_1, q_2) = (e_{i,j,k} + \lambda_1 \cdot v, e_{i,j,k} + \lambda_2 \cdot v)$$

   where $e_{i,j,k}$ denotes the vector with 1 at the indices i, j, k and 0 elsewhere.

2. The answer algorithm $\text{ANS}^{\mathbb{F}}_n(x, q)$ given the query q and the database x, outputs

   $$(F_x(q), \partial_1 F_x(q), \ldots, \partial_m F_x(q))$$

   where $\partial_i F_x$ denotes the polynomial that is obtained by taking the partial derivative of F_x with respect to the ith variable z_i.

3. The decoding algorithm $\text{DEC}^{\mathbb{F}}_{(i,j,k),n}(r, \mathbf{ans}_1, \mathbf{ans}_2)$ parses the randomness r to recover the original queries (q_1, q_2). The partial derivatives (from $\mathbf{ans}_1, \mathbf{ans}_2$) are utilized to recover the value of the first derivative of the univariate polynomial of degree-3 $f(\lambda) = F(e_{i,j,k} + \lambda \cdot v)$ at the points λ_1 and λ_2. Finally, the recovered four values $f(\lambda_1), f'(\lambda_1), f(\lambda_2), f'(\lambda_2)$ uniquely determine the coefficients of $f(\lambda)$ by solving a system of linear equations and outputs $f(0) = x_{i,j,k}$. This is done by applying Hermite interpolation which is a linear operation in q_1 and q_2.

Analysis. For each server the query q_s is uniformly distributed over $\mathbb{F}^m$. Thus, the scheme is secure. The asymptotic communication complexity is $O(n^{\frac{1}{3}})$ as the query length is m and the answer length is $m + 1$. Finally, all the algorithms can be implemented by arithmetic circuits. Furthermore, the answer algorithm is linear in the database and the decoding algorithm is linear in the answers, so the resulting PIR is indeed fully-linear. (We also note that we only need the inverses of fixed points that are determined by the public evaluation points λ_1, λ_2. So for fixed randomness all the algorithms are strictly arithmetic.)

[18] There are several ways to handle the case of a field of size 2 (for example, by working over the extension field $\text{GF}(4)$). Moreover, one can obtain a field-agnostic arithmetic construction by first checking whether the underlying field has size 2, and then selecting either the binary scheme or the general scheme accordingly. This test can be implemented by sampling many random field elements and verifying that all sampled elements are equal to either 0 or 1.

6 Two-server Arithmetic PIR with Computational Security

6.1 Overview

A *distributed point function* (DPF) [38] enables splitting a "point function" $f_{\alpha,\beta}$, evaluating to β on input α and to 0 elsewhere, into two additive shares. Concretely, the two shares are described by compact keys k_0, k_1 such that (1) each k_i hides $f_{\alpha,\beta}$ and (2) each k_i specifies a function s_i satisfying

$$f_{\alpha,\beta} = s_0 - s_1.$$

In the arithmetic setting, we view $f_{\alpha,\beta}$, s_0, and s_1 as functions from $[n]$ to $\mathbb{F}$, and accordingly take $\alpha \in [n]$ and $\beta \in \mathbb{F}$. We can associate $f_{\alpha,\beta}$ with its truth-table representation, namely the vector $\beta \cdot \mathbf{1}_\alpha \in \mathbb{F}^n$, where $\mathbf{1}_\alpha$ denotes the 1-hot vector of length n. Thus, we may think of k_0 and k_1 as compact representations of vectors $s_0, s_1 \in \mathbb{F}^n$ that form a computationally private additive secret sharing of $\beta \cdot \mathbf{1}_\alpha \in \mathbb{F}^n$.

As in the binary case, arithmetic DPFs naturally give rise to a two-server computational PIR scheme. A client wishing to retrieve the ith element of a database shares the point function $f_{i,1} = \mathbf{1}_i$ into two short keys k_0 and k_1, and sends one key to each server. Server $b \in \{0, 1\}$ extracts the vector s_b from k_b and computes

$$y_b = \langle s_b, x \rangle$$

where $x \in \mathbb{F}^n$ is the database and $\langle \cdot, \cdot \rangle$ stands for inner-product. Given $y_0, y_1 \in \mathbb{F}$, the client computes

$$y_0 - y_1 = \langle s_0 - s_1, x \rangle = x[i].$$

The communication complexity is dominated by the length of the keys, and if the DPF is arithmetic, then so is the resulting PIR scheme. Therefore Theorem 1.2 follows from the existence of arithmetic DPFs.

In Section D we formally define the notion of arithmetic DPFs and prove the following theorem.

Theorem 6.1. *Assuming the existence of an arithmetic PRG, there exists an arithmetic DPF with key-length polynomial in the security parameter. Accordingly, there exists a 2-server arithmetic PIR with computational security with communication of n^ϵ for an arbitrary constant $\epsilon > 0$ where n is the length of the database.*[19]

Recall that arithmetic PRGs can be constructed based on standard assumptions, like the arithmetic variant of LPN. (See Appendix C.)

6.2 Proof Sketch

We sketch the construction here and refer the reader to Appendix D for full details. We show that the DPF construction of [20] can be arithmetized. To simplify notation, we assume that we wish to share a block-point function, namely a function $f_{\alpha,\beta} \in (\mathbb{F}^\lambda)^n$ whose αth block is equal to $\beta \in \mathbb{F}^\lambda$.

The construction can be described via the following recursive process. We decompose the index α into $y \in [n/2]$ and $z \in \{0, 1\}$ such that $\alpha = 2y + z$. Assume that we have already generated short keys k'_0 and k'_1 for the function $f_{y,\beta'} \in (\mathbb{F}^\lambda)^{n/2}$, where $\beta' \in \mathbb{F}^\lambda$ is pseudorandom. That is, k'_0 and k'_1 specify a pair of vectors $s'_0, s'_1 \in (\mathbb{F}^\lambda)^{n/2}$ that are composed of $n/2$ blocks of size λ each, such that

$$s'_0 = (s'_0[1], \ldots, s'_0[n/2]) \quad \text{and} \quad s'_1 = (s'_1[1], \ldots, s'_1[n/2])$$

[19] If the PRG is assumed to have sub-exponential security then the communication can be reduced to $\mathrm{polylog}(n)$.

agree on every block except the yth block. We further assume that the keys k'_0, k'_1 also specify a pseudorandom secret sharing $t'_0, t'_1 \in \mathbb{F}^{n/2}$ satisfying $t'_0 - t'_1 = 1_y$. That is, t'_0 and t'_1 (referred to as the *control vectors*) agree on every entry except the yth entry, where they differ by one.

To generate a sharing for $f_{\alpha,\beta}$, we append to k'_0 and k'_1 a single correction vector $\Delta \in \mathbb{F}^{2\lambda+2}$, whose value will be determined later.[20] Our goal is to expand s'_b and t'_b into longer vectors $s_b \in (\mathbb{F}^\lambda)^n$ and $t_b \in \mathbb{F}^n$ such that $s_0 - s_1 = \beta \cdot 1_\alpha$ and $t_0 - t_1 = 1_\alpha$. (The new control vectors will allow the recursion to continue.)

The expansion is performed in two steps. Let $G : \mathbb{F}^\lambda \to \mathbb{F}^{2\lambda+2}$ be an arithmetic pseudorandom generator. First, we apply the PRG to expand each block i of s'_0 (resp., s'_1) into two blocks and two scalars. Specifically, for each $i \in [n/2]$,

$$(s''_b[2i], t''_b[2i], s''_b[2i+1], t''_b[2i+1]) = G(s'_b[i]).$$

Collecting the outputs yields vectors $s''_b \in (\mathbb{F}^\lambda)^n$ and $t''_b \in \mathbb{F}^n$. Since $s'_0[i] = s'_1[i]$ for all $i \neq y$, it follows that

$$(*) \qquad s''_0 \text{ and } s''_1 \text{ (resp., } t''_0 \text{ and } t''_1\text{) agree on all blocks except } 2y \text{ and } 2y+1.$$

The second step is a correction step, whose purpose is to fix these special blocks. For each $i \in [n/2]$, we define

$$(s_b[2i], t_b[2i], s_b[2i+1], t_b[2i+1]) = (s''_b[2i], t''_b[2i], s''_b[2i+1], t''_b[2i+1]) + t'_b[i] \cdot \Delta.$$

Observe that, regardless of the choice of Δ, property (*) continues to hold: for every $i \neq y$, we have $s_0[i] = s_1[i]$ and $t_0[i] = t_1[i]$, since $t'_0[i] = t'_1[i]$. On the other hand, for $i = y$, we have $t'_0[i] - t'_1[i] = 1$, which allows us to arbitrarily "program" the difference between

$$(s_0[2i], t_0[2i], s_0[2i+1], t_0[2i+1]) \quad \text{and} \quad (s_1[2i], t_1[2i], s_1[2i+1], t_1[2i+1]).$$

In particular, we can ensure that $s_0 - s_1 = \beta \cdot 1_\alpha$ and $t_0 - t_1 = 1_\alpha$. Finally, it is not hard to show that Δ itself is pseudorandom given one of the keys k'_b, and therefore the augmented keys $k_b = (k'_b, \Delta)$ hide both α and β. The base case ($n = 2$) can be handled via a trivial information-theoretic secret-sharing with keys of length $2\lambda + 2$, and so the overall complexity of the construction is $O(\lambda \cdot \log n)$.

7 Secret-Key Arithmetic PIR

In this section we present an alternative one-server PIR model in the arithmetic setting that achieves a non-trivial answer length, in contrast to the lower bound established in Theorem 3.4. The following definition is adopted from [24] to the arithmetic setting.

Definition 7.1 (Single-server sk-APIR Scheme)**.** *A* single-server secret-key arithmetic PIR *(sk-APIR) scheme consists of the following four families of polynomial-sized arithmetic circuits, each parameterized by the database length n:*

- *A randomized key generation circuit* $\text{KGEN}_n^{\mathbb{F}}(1^n)$ *that samples a random secret key* sk.
- *A deterministic encoding circuit* $\text{ENC}_n^{\mathbb{F}}(\mathsf{sk}, x)$ *that, given the secret key* sk *and the database x, encodes the database that is later uploaded to the server. Let $\hat{x} \in \mathbb{F}^{m(n)}$ be the encoded database of size $m(n)$. We refer to $m(n)$ as the size of the encoded database.*

[20] As in [20], one can optimize the construction and reduce the length of Δ to $\lambda + 2$ field elements; however, this does not affect the asymptotic complexity.

- *A randomized query circuit* $\text{QUERY}_n^{\mathbb{F}}(\mathsf{sk}, i; r)$ *that given a secret key* sk*, an index* $i \in [n]$ *in binary form and randomness* $r = (r_A, r_B) \xleftarrow{R} \mathbb{F}^{\rho_A(n)} \times \{0,1\}^{\rho_B(n)}$ *that can contain both random bits and random field elements outputs a query* $q \in \mathbb{F}^{t(n)}$*. We refer to* $t(n)$ *as the query length.*

- *A deterministic answer circuit* $\text{ANS}_n^{\mathbb{F}}(\hat{x}, q)$ *that given a query* $q \in \mathbb{F}^{t(n)}$ *and an encoded database* $\hat{x} \in \mathbb{F}^{m(n)}$ *outputs an answer* $\mathbf{ans} \in \mathbb{F}^{\ell(n)}$*. We refer to* $\ell(n)$ *as the answer length.*

- *A deterministic decoding circuit* $\text{DEC}_n^{\mathbb{F}}(\mathsf{sk}, i, r, \mathbf{ans})$ *that given a secret key* sk*, an index* $i \in [n]$ *in binary, randomness* $r = (r_A, r_B) \in \mathbb{F}^{\rho_A(n)} \times \{0,1\}^{\rho_B(n)}$ *and an answer* $\mathbf{ans}$ *outputs a single field element* y*.*

The algorithms should satisfy the following properties:

- δ*-Correctness: For every field* $\mathbb{F}$*, database* $x \in \mathbb{F}^n$*, secret key* $\mathsf{sk} \leftarrow \text{KGEN}^{\mathbb{F}}(1^n)$ *such that* $\hat{x} = \text{ENC}_n^{\mathbb{F}}(\mathsf{sk}, x)$ *and every index* $i \in [n]$*, then the error probability (over the randomness* r*)*

$$\Pr_r[\text{DEC}_{i,n}^{\mathbb{F}}(\mathsf{sk}, r, \text{ANS}_n^{\mathbb{F}}(\hat{x}, \text{QUERY}_n^{\mathbb{F}}(\mathsf{sk}, i, r))) \neq x_i] \leq \delta.$$

  By default, δ *is negligible in* n*.*

- *Privacy (against a single server): For every adversary* $\mathcal{A} = \{\mathcal{A}_n\}$ *(modeled as a polynomial-size Boolean circuit) the winning probability in the following game (parameterized by* n*) is at most* $\frac{1}{2} + \text{negl}(n)$*:*

  - *The adversary* $\mathcal{A}_n$ *chooses an implementation of a field* $\mathbb{F}$*, a database* $x \in \mathbb{F}^n$*, and sends it to the challenger.*
  - *The challenger samples a bit* $b \xleftarrow{R} \{0,1\}$ *a secret key* $\mathsf{sk} \leftarrow \text{KGEN}(1^n)$*, computes* $\hat{x} \leftarrow \text{ENC}(\mathsf{sk}, x)$ *and gives the adversary the encoded database* $\hat{x}$ *together with an oracle access to the randomized function* $Q_b^{\mathbb{F}}(\mathsf{sk}, \cdot)$ *where* $Q_0^{\mathbb{F}}(\mathsf{sk}, \cdot)$ *executes the randomized query algorithms* $\text{QUERY}^{\mathbb{F}}(\mathsf{sk}, \cdot)$ *and* $Q_1^{\mathbb{F}}(\mathsf{sk}, \cdot)$ *is an oracle that ignores its input and outputs* $\text{QUERY}^{\mathbb{F}}(\mathsf{sk}, 1)$*. In both cases, the internal randomness is sampled independently for each call.*
  - *The adversary* $\mathcal{A}$ *outputs a bit* b' *and wins if* $b' = b$*.*

7.1 Construction

We rely on the following arithmetic variant of the LPN assumption from [45], which is sometimes referred to as the Random Linear Code (RLC) assumption. The following definition is based on [7].

Definition 7.2 (LPN Assumption - arithmetic variant)**.** *The* Arithmetic Learning Parity with Noise *assumption* $(k, m(k), \mu(k))$*-aLPN asserts that for every adversary* $\mathcal{A} = \{\mathcal{A}_k\}$ *(modeled as a polynomial-size Boolean circuit) the winning probability in the following game (parameterized by* k*) is at most* $\frac{1}{2} + \text{negl}(k)$*:*

- *The adversary* $\mathcal{A}_k$ *chooses an implementation of a field* $\mathbb{F}$ *and sends it to the challenger.*
- *The challenger samples a bit* $b \xleftarrow{R} \{0,1\}$*, if* $b = 0$ *it sends back* m *uniformly random row vectors*

$$(a_i, y_i) \xleftarrow{R} \mathbb{F}^k \times \mathbb{F}, \qquad \forall i \in [m] \tag{3}$$

and if $b = 1$ it sends back the vectors

$$(a_i, a_i \cdot s + e_i) \in \mathbb{F}^k \times \mathbb{F}, \quad \forall i \in [m], \tag{4}$$

where $a_1, \ldots, a_m \xleftarrow{R} \mathbb{F}^k$, $s \xleftarrow{R} \mathbb{F}^k$ is a random secret column vector and $e_1, \ldots, e_m \in \mathbb{F}$ are "noise" elements, i.e., each entry is iid and is taken to be 0 with probability $1 - \mu$ and random field element with probability μ.

- *The adversary returns the bit b' and wins if $b' = b$.*

We assume that aLPN holds for arbitrary polynomial $m(\cdot)$ and whenever $\mu > \frac{1}{k^{1-c}}$ for some constant $c > 0$.

Remark 7.3 (On the arithmetic LPN assumption)**.** *Let $\mathbb{F} = (\mathbb{F}_k)_{k \in \mathbb{N}}$ be a field family. The $(k, m(k), \mu(k), \mathbb{F})$-*LPN assumption *asserts that the probability ensemble (indexed by k)*

$$(a_i, a_i \cdot s + e_i) \in \mathbb{F}^k \times \mathbb{F}, \qquad \forall i \in [m(k)],$$

defined as in Eq. (4), is pseudorandom over $(\mathbb{F}_k^k \times \mathbb{F}_k)^{m(k)}$.

Under this terminology, the arithmetic LPN assumption stated above is equivalent to requiring that the $(k, t(k), \mu(k), \mathbb{F})$-LPN assumption holds for every efficiently realizable field family $\mathbb{F}$. This formulation was adopted in [45], extending the binary LPN assumption of [17, 40]. We further note that our field-agnostic construction is secure over every field family $\mathbb{F}$ for which the corresponding $\mathbb{F}$-LPN assumption holds.

Remark 7.4 (Matrix-LPN assumption)**.** *We will also consider a matrix variant of the LPN assumption, parameterized by an additional parameter $N = N(k)$. This variant is defined analogously to the standard formulation, with the following modifications. In Eq.* (3)*, we sample $y_1, \ldots, y_m \xleftarrow{R} \mathbb{F}^N$ as independent uniformly random vectors. In Eq.* (4)*, the secret is sampled as a random matrix $s \xleftarrow{R} \mathbb{F}^{k \times N}$, and each noise term $e_i \in \mathbb{F}^N$ is sampled independently coordinate-wise, where every entry equals 0 with probability $1 - \mu$ and a uniformly random field element with probability μ.*

Intuitively, this assumption captures the setting of simultaneously solving N LPN instances (one for each column of s) that share the same public samples $a_1, \ldots, a_m$. The standard LPN assumption is recovered as the special case $N = 1$. Moreover, it is not difficult to show (see, e.g., [8]) that the standard $(k, m(k), \mu(k))$-aLPN assumption implies the matrix $(k, m(k), \mu(k), N(k))$-aLPN assumption for every polynomially bounded function $N(k)$. The same proof extends verbatim to the arithmetic setting.

In Figure 1 we describe a simple construction of sk-APIR due to [24] based on the LPN assumption. Since this mainly serves as a feasibility result we describe the simplest variant of their construction. To simplify the description, we employ a long secret key sk that each of its entries is sampled uniformly from the underlying field. The secret key sk can be generically compressed into a short seed $\mathsf{sk}' \in \mathbb{F}^{n^\delta}$ for arbitrary constant $\delta > 0$, using an arithmetic PRG whose existence follows from the LPN as well. (See Section C)

In addition, the scheme has noticeable error and achieves only $O(n^{0.5+\delta})$ communication for an arbitrary constant $\delta > 0$. These can be amplified via the techniques of [24] that arithmetize naturally. (See Remark 7.6.)

For completeness, we sketch here a security proof for this basic PIR. (The proof in [24] applies to a somewhat more complicated variant.)

Theorem 7.5. *Suppose that the $(k, m(k), \mu)$-LPN assumption holds for dimension k, noise rate μ and arbitrary polynomial $m(\cdot)$. Then, the basic scheme is a secret-key arithmetic PIR with communication complexity of $O(k + N + L)$ and with the error of $N\mu = o(1)$.*

Basic-CIMR

Parameters: Let $\mathbb{F}$ be a field. We assume that the database size is $n \in \mathbb{N}$ and view the database $X \in \mathbb{F}^{N \times L}$ as an $N \times L$ matrix, where N, L are parameters satisfying $N \cdot L = n$. Let $k = k(n)$ and $\mu = \mu(n) \in (0,1)$ denote the LPN parameters (dimension and noise rate, respectively). To obtain security against $\mathrm{poly}(n)$-time adversaries (and even subexponential-time adversaries), we require that $k = \mathrm{poly}(n)$ and $\mu > \frac{1}{k^{1-c}}$ for some constant $c > 0$. To achieve error probability $o(1)$, we additionally require that $\mu = o(1/N)$. For concreteness, let $N, L = \Theta(\sqrt{n})$ and set $k = N^{1+\delta}$ and $\mu = 1/N^{1+0.5\delta}$ for some constant $\delta > 0$.

Key Generation: The key generation algorithm $\mathrm{KGEN}_n^{\mathbb{F}}$, samples a random key $\mathsf{sk} = (R, S)$ where $R \xleftarrow{R} \mathbb{F}^{(k+N)\times L}$ is a random masking matrix and $S = (s_1, \ldots, s_N) \xleftarrow{R} \mathbb{F}^{k \times N}$ is a matrix that each of its columns will be used as an LPN secret vector of length k. Let C_S denote the linear code whose parity-check matrix is the matrix $(I_k, S) \in \mathbb{F}^{k \times (k+N)}$ where I_k is the $k \times k$ identity matrix, and let $(G_S, I_N) \in \mathbb{F}^{(k+N)\times N}$ be the generating matrix of this code.

Database Encoding: The database encoding algorithm $\mathrm{ENC}_n^{\mathbb{F}}(\mathsf{sk}, X)$ given the secret key $\mathsf{sk} = (R, S)$, and an input database $X \in \mathbb{F}^{N \times L}$, outputs the encoded database

$$\hat{X} = (G_S, I_N) \cdot X + R \in \mathbb{F}^{(k+N)\times L}.$$

Query: The query algorithm $\mathrm{QUERY}_n^{\mathbb{F}}(\mathsf{sk}, (i_0, i_1); r)$ generates the query $q \in \mathbb{F}^{k+N}$ as follows: Sample a public row vector $a \xleftarrow{R} \mathbb{F}^k$, and a secret row vector $e = (e_1, \ldots, e_N)$ of "noise" elements, i.e., each entry is i.i.d. and is taken to be 0 with probability $1 - \mu$ and random field element with probability μ, and output

$$q = (a, aS + e + 1_{i_0}),$$

where $1_{i_0} \in \mathbb{F}^N$ is the row vector that has 1 on its i_0 coordinate and zero elsewhere.

Answer: The answer algorithm $\mathrm{ANS}_n^{\mathbb{F}}(\hat{X}, q)$, outputs $\mathbf{ans} = q\hat{X} \in \mathbb{F}^{1\times L}$.

Decode: The decoding algorithm $\mathrm{DEC}_{i_0,i_1,n}^{\mathbb{F}}$, given $\mathsf{sk} = (S, R)$, q, and $\mathbf{ans}$, outputs $x_{i_0,i_1} = (\mathbf{ans} - q \cdot R)_{i_1}$.

Figure 1: CIMR-PIR

Sketch. The communication complexity is immediate. We claim that correctness holds whenever the error vector e is the all-zero vector, which happens with probability of at least $1 - \mu N$. Indeed, in this case, the vector (a, as) is in the span of (I_k, S) which is the left kernel of (G_S, I_N) and therefore

$$q\hat{X} - qR = (a, as + 1_{i_0})(G_S, I_N) \cdot X = 1_{i_0} \cdot X,$$

as required.

To prove privacy, assume towards a contradiction that the basic sk-APIR is not private and let $\mathcal{A} = \{\mathcal{A}_n\}$ denote an adversary (modeled as a polynomial Boolean circuit) that wins the PIR privacy game with $m(n) = \text{poly}(n)$ queries and advantage $\frac{1}{2} + \epsilon(n)$, for some non-negligible function $\epsilon(n)$. We describe an adversary $\mathcal{B}$ that breaks the (k, m, μ)-LPN assumption in its matrix form with $N = N(n)$ instances. (See Remark 7.4.)

The adversary works as follows:

1. Simulate $\mathcal{A}$ and forward the field $\mathbb{F}$ returned by $\mathcal{A}$ to the LPN challenger.
2. Get a set of row vectors $V = \left\{(a_j, y_j) \in \mathbb{F}^k \times \mathbb{F}^N\right\}_{j \in [m]}$, sampled by the matrix-LPN challenger.
3. Send a uniformly random "encoded database" $\hat{X} \in \mathbb{F}^{(k+N) \times L}$ to $\mathcal{A}$ and sample a private random bit $b \xleftarrow{R} \{0, 1\}$.
4. Answer the jth oracle query (i_0, i_1) of $\mathcal{A}$ with the vector

   $$(a_j, y_j + 1_{i_0}) \quad \text{if } b = 0,$$

   and with the vector

   $$(a_j, y_j + 1_1), \quad \text{if } b = 1.$$

5. Given $\mathcal{A}$'s output b' output 1 ("LPN instance") if the guess is correct ($b' = b$). Otherwise, output 0 ("random instance").

If the LPN challenger provides an LPN instance, the emulation is perfect and the adversary $\mathcal{B}$ outputs 1 with probability $\frac{1}{2} + \epsilon(n)$. On the other hand, if V is a set of uniformly random vectors, the answers to $\mathcal{A}$ queries are uniformly random, independently of the choice of b. In this case, $\mathcal{B}$ outputs 0 with probability $\frac{1}{2}$. Overall, $\mathcal{B}$ wins the LSN game with probability

$$\geq \frac{1}{2}\left(\frac{1}{2} + \epsilon(n)\right) + \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{2} + \frac{\epsilon(n)}{2}$$

where $\epsilon(n)$ is a non-negligible function, and the claim follows. □

Remark 7.6 (Amplification)**.** *Assume that $N, L = \Theta(\sqrt{n})$ and set $k = N^{1+\delta}$ and $\mu = 1/N^{1+0.5\delta}$ for some small constant $\delta > 0$. Then the communication complexity is $O(n^{0.5+0.5\delta})$ and the error is $1/n^{0.25\delta}$. The decoding error can be made negligible by repeating the query, say $\text{polylog}(n)$ times, resulting in a scheme with communication complexity $O(n^{0.5+\delta})$.*

Moreover, since both the server computation and the preprocessing are linear, the answer length can also be reduced to n^ϵ via recursion, as explained in [24]. (The transformation generalizes smoothly to the arithmetic setting.) Furthermore, just like in [24], if one replaces the aLPN assumption with the (arithmetic) Learning Subspace with Noise assumption, it is possible to get an almost optimal encoding length of size $(1 + \epsilon)n$.

Combining Theorem 7.5 with Remark 7.6, we obtain the following result (previously stated as Informal Theorem 1.3).

Theorem 7.7. *Assuming that the aLPN assumption holds, there exists a single-server arithmetic sk-PIR scheme with computational security. For a length-n database, the online communication complexity is $O(n^{\epsilon})$ for an arbitrary constant $\epsilon > 0$.*

References

[1] Damiano Abram, Giulio Malavolta, and Lawrence Roy. Slightly sublinear trapdoor hash functions and PIR from low-noise LPN. In Benny Applebaum and Huijia Rachel Lin, editors, *Theory of Cryptography - 23rd International Conference, TCC 2025, Aarhus, Denmark, December 1-5, 2025, Proceedings, Part I*, volume 16268 of *Lecture Notes in Computer Science*, pages 280–300. Springer, 2025. doi:10.1007/978-3-032-12287-2_10.

[2] Ishtiyaque Ahmad, Yuntian Yang, Divyakant Agrawal, Amr El Abbadi, and Trinabh Gupta. Addra: Metadata-private voice communication over fully untrusted infrastructure. In Angela Demke Brown and Jay R. Lorch, editors, *15th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2021, July 14-16, 2021*. USENIX Association, 2021. URL: https://www.usenix.org/conference/osdi21/presentation/ahmad.

[3] Michael Alekhnovich. More on average case vs approximation complexity. In *44th Symposium on Foundations of Computer Science, FOCS 2003, Cambridge, MA, USA, October 11-14, 2003, Proceedings*, pages 298–307. IEEE Computer Society, 2003. doi:10.1109/SFCS.2003.1238204.

[4] Asra Ali, Tancrède Lepoint, Sarvar Patel, Mariana Raykova, Phillipp Schoppmann, Karn Seth, and Kevin Yeo. Communication-computation trade-offs in PIR. In Michael D. Bailey and Rachel Greenstadt, editors, *30th USENIX Security Symposium, USENIX Security 2021, August 11-13, 2021*, pages 1811–1828. USENIX Association, 2021. URL: https://www.usenix.org/conference/usenixsecurity21/presentation/ali.

[5] Bar Alon, Amos Beimel, and Or Lasri. Simplified PIR and CDS protocols and improved linear secret-sharing schemes. In Benny Applebaum and Huijia Rachel Lin, editors, *Theory of Cryptography - 23rd International Conference, TCC 2025, Aarhus, Denmark, December 1-5, 2025, Proceedings, Part II*, volume 16269 of *Lecture Notes in Computer Science*, pages 365–398. Springer, 2025. doi:10.1007/978-3-032-12293-3_12.

[6] Sebastian Angel, Hao Chen, Kim Laine, and Srinath T. V. Setty. PIR with compressed queries and amortized query processing. In *2018 IEEE Symposium on Security and Privacy, SP 2018, Proceedings, 21-23 May 2018, San Francisco, California, USA*, pages 962–979. IEEE Computer Society, 2018. doi:10.1109/SP.2018.00062.

[7] Benny Applebaum, Jonathan Avron, and Chris Brzuska. Arithmetic cryptography. *J. ACM*, 64(2):10:1–10:74, 2017. doi:10.1145/3046675.

[8] Benny Applebaum, David Cash, Chris Peikert, and Amit Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In Shai Halevi, editor, *Advances in Cryptology - CRYPTO 2009, 29th Annual International Cryptology Conference, Santa Barbara, CA, USA,*

August 16-20, 2009. Proceedings, volume 5677 of *Lecture Notes in Computer Science*, pages 595–618. Springer, 2009. doi:10.1007/978-3-642-03356-8_35.

[9] Benny Applebaum, Ivan Damgård, Yuval Ishai, Michael Nielsen, and Lior Zichron. Secure arithmetic computation with constant computational overhead. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part I*, volume 10401 of *Lecture Notes in Computer Science*, pages 223–254. Springer, 2017. doi:10.1007/978-3-319-63688-7_8.

[10] Benny Applebaum, Naama Haramaty, Yuval Ishai, Eyal Kushilevitz, and Vinod Vaikuntanathan. Low-complexity cryptographic hash functions. In Christos H. Papadimitriou, editor, *8th Innovations in Theoretical Computer Science Conference, ITCS 2017, Berkeley, CA, USA, January 9-11, 2017*, volume 67 of *LIPIcs*, pages 7:1–7:31. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPICS.ITCS.2017.7.

[11] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. How to garble arithmetic circuits. In Rafail Ostrovsky, editor, *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011*, pages 120–129. IEEE Computer Society, 2011. doi:10.1109/FOCS.2011.40.

[12] Benny Applebaum and Niv Konstantini. Actively secure arithmetic computation and VOLE with constant computational overhead. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology - EUROCRYPT 2023 - 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Lyon, France, April 23-27, 2023, Proceedings, Part II*, volume 14005 of *Lecture Notes in Computer Science*, pages 190–219. Springer, 2023. doi:10.1007/978-3-031-30617-4_7.

[13] Amos Beimel. *Secure Schemes for Secret Sharing and Key Distribution*. PhD thesis, Technion, 1996.

[14] Amos Beimel, Yuval Ishai, and Eyal Kushilevitz. General constructions for information-theoretic private information retrieval. *J. Comput. Syst. Sci.*, 71(2):213–247, 2005. doi:10.1016/J.JCSS.2005.03.002.

[15] Fabrice Benhamouda, Caicai Chen, Shai Halevi, Yuval Ishai, Hugo Krawczyk, Tamer Mour, Tal Rabin, and Alon Rosen. Encrypted matrix-vector products from secret dual codes. In Chun-Ying Huang, Jyh-Cheng Chen, Shiuh-Pyng Shieh, David Lie, and Véronique Cortier, editors, *Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security, CCS 2025, Taipei, Taiwan, October 13-17, 2025*, pages 394–408. ACM, 2025. doi:10.1145/3719027.3765194.

[16] Sebastian Bitzer, Maximilian Egger, Mumin Liu, and Antonia Wachter-Zeh. Post-quantum secure aggregation via code-based homomorphic encryption, 2026. URL: https://arxiv.org/abs/2601.13031, arXiv:2601.13031.

[17] Avrim Blum, Merrick L. Furst, Michael J. Kearns, and Richard J. Lipton. Cryptographic primitives based on hard learning problems. In Douglas R. Stinson, editor, *Advances in Cryptology - CRYPTO '93, 13th Annual International Cryptology Conference, Santa Barbara, California, USA, August 22-26, 1993, Proceedings*, volume 773 of *Lecture Notes in Computer Science*, pages 278–291. Springer, 1993. doi:10.1007/3-540-48329-2_24.

[18] Şeyma Bodur, Edgar Martínez-Moro, and Diego Ruano. Single server private information retrieval protocols with codes over rings. *Journal of Algebra and Its Applications*, 24(13n14):2541012, 2025. doi:10.1142/S0219498825410129.

[19] Sarah Bordage and Julien Lavauzelle. On the privacy of a code-based single-server computational PIR scheme. *Cryptography and Communications*, 13(4):519–526, 2021.

[20] Elette Boyle, Niv Gilboa, and Yuval Ishai. Function secret sharing: Improvements and extensions. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 1292–1303, 2016.

[21] Elette Boyle, Yuval Ishai, Rafael Pass, and Mary Wootters. Can we access a database both locally and privately? In Yael Kalai and Leonid Reyzin, editors, *Theory of Cryptography - 15th International Conference, TCC 2017, Baltimore, MD, USA, November 12-15, 2017, Proceedings, Part II*, volume 10678 of *Lecture Notes in Computer Science*, pages 662–693. Springer, 2017. doi:10.1007/978-3-319-70503-3_22.

[22] Ran Canetti, Justin Holmgren, and Silas Richelson. Towards doubly efficient private information retrieval. *IACR Cryptol. ePrint Arch.*, page 568, 2017. URL: http://eprint.iacr.org/2017/568.

[23] Ran Canetti, Yuval Ishai, Ravi Kumar, Michael K. Reiter, Ronitt Rubinfeld, and Rebecca N. Wright. Selective private function evaluation with applications to private statistics. In Ajay D. Kshemkalyani and Nir Shavit, editors, *Proceedings of the Twentieth Annual ACM Symposium on Principles of Distributed Computing, PODC 2001, Newport, Rhode Island, USA, August 26-29, 2001*, pages 293–304. ACM, 2001. doi:10.1145/383962.384047.

[24] Caicai Chen, Yuval Ishai, Tamer Mour, and Alon Rosen. Secret-key PIR from random linear codes. *Cryptology ePrint Archive*, 2025. To appear in STOC 2026.

[25] Benny Chor, Oded Goldreich, Eyal Kushilevitz, and Madhu Sudan. Private information retrieval. *J. ACM*, 45(6):965–981, 1998. Conference version in FOCS 1995. doi:10.1145/293347.293350.

[26] Gil Cohen, Ivan Bjerre Damgård, Yuval Ishai, Jonas Kölker, Peter Bro Miltersen, Ran Raz, and Ron D. Rothblum. Efficient multiparty protocols via log-depth threshold formulae - (extended abstract). In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part II*, volume 8043 of *Lecture Notes in Computer Science*, pages 185–202. Springer, 2013. doi:10.1007/978-3-642-40084-1_11.

[27] Ronald Cramer and Serge Fehr. Optimal black-box secret sharing over arbitrary abelian groups. *IACR Cryptol. ePrint Arch.*, page 36, 2002. URL: http://eprint.iacr.org/2002/036.

[28] Ronald Cramer, Serge Fehr, Yuval Ishai, and Eyal Kushilevitz. Efficient multi-party computation over rings. In Eli Biham, editor, *Advances in Cryptology - EUROCRYPT 2003, International Conference on the Theory and Applications of Cryptographic Techniques, Warsaw, Poland, May 4-8, 2003, Proceedings*, volume 2656 of *Lecture Notes in Computer Science*, pages 596–613. Springer, 2003. doi:10.1007/3-540-39200-9_37.

[29] Giovanni Di Crescenzo, Tal Malkin, and Rafail Ostrovsky. Single database private information retrieval implies oblivious transfer. In Bart Preneel, editor, *Advances in Cryptology - EUROCRYPT 2000, International Conference on the Theory and Application of Cryptographic Techniques, Bruges, Belgium, May 14-18, 2000, Proceeding*, volume 1807 of *Lecture Notes in Computer Science*, pages 122–138. Springer, 2000. doi:10.1007/3-540-45539-6_10.

[30] Alex Davidson, Gonçalo Pestana, and Sofía Celi. FrodoPIR: Simple, scalable, single-server private information retrieval. *Proc. Priv. Enhancing Technol.*, 2023(1):365–383, 2023. doi:10.56553/POPETS-2023-0022.

[31] Yvo Desmedt and Yair Frankel. Shared generation of authenticators and signatures (extended abstract). In Joan Feigenbaum, editor, *Advances in Cryptology - CRYPTO '91, 11th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1991, Proceedings*, volume 576 of *Lecture Notes in Computer Science*, pages 457–469. Springer, 1991. doi:10.1007/3-540-46766-1_37.

[32] Yvo Desmedt, Josef Pieprzyk, Ron Steinfeld, and Huaxiong Wang. On secure multi-party computation in black-box groups. In Alfred Menezes, editor, *Advances in Cryptology - CRYPTO 2007, 27th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2007, Proceedings*, volume 4622 of *Lecture Notes in Computer Science*, pages 591–612. Springer, 2007. doi:10.1007/978-3-540-74143-5_33.

[33] Yevgeniy Dodis, Yael Tauman Kalai, and Shachar Lovett. On cryptography with auxiliary input. In Michael Mitzenmacher, editor, *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009*, pages 621–630. ACM, 2009. doi:10.1145/1536414.1536498.

[34] Zeev Dvir, Ariel Gabizon, and Avi Wigderson. Extractors and rank extractors for polynomial sources. *SIAM Journal on Computing*, 38(6):2144–2155, 2009. doi:10.1137/080715372.

[35] Zeev Dvir and Sivakanth Gopi. 2-server PIR with sub-polynomial communication. In Rocco A. Servedio and Ronitt Rubinfeld, editors, *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 577–584. ACM, 2015. doi:10.1145/2746539.2746546.

[36] Zeev Dvir and Amir Shpilka. Locally decodable codes with two queries and polynomial identity testing for depth 3 circuits. *SIAM J. Comput.*, 36(5):1404–1434, 2007. doi:10.1137/05063605X.

[37] Shimon Even, Oded Goldreich, and Abraham Lempel. A randomized protocol for signing contracts. In David Chaum, Ronald L. Rivest, and Alan T. Sherman, editors, *Advances in Cryptology: Proceedings of CRYPTO '82, Santa Barbara, California, USA, August 23-25, 1982*, pages 205–210. Plenum Press, New York, 1982. doi:10.1007/978-1-4757-0602-4_19.

[38] Niv Gilboa and Yuval Ishai. Distributed point functions and their applications. In Phong Q. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology - EUROCRYPT 2014 - 33rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Copenhagen, Denmark, May 11-15, 2014. Proceedings*, volume 8441 of *Lecture Notes in Computer Science*, pages 640–658. Springer, 2014. doi:10.1007/978-3-642-55220-5_35.

[39] Oded Goldreich, Howard J. Karloff, Leonard J. Schulman, and Luca Trevisan. Lower bounds for linear locally decodable codes and private information retrieval. *Comput. Complex.*, 15(3):263–296, 2006. doi:10.1007/S00037-006-0216-3.

[40] Oded Goldreich, Hugo Krawczyk, and Michael Luby. On the existence of pseudorandom generators. *SIAM J. Comput.*, 22(6):1163–1175, 1993. doi:10.1137/0222069.

[41] Iftach Haitner, Jonathan J. Hoch, Omer Reingold, and Gil Segev. Finding collisions in interactive protocols - tight lower bounds on the round and communication complexities of statistically hiding commitments. *SIAM J. Comput.*, 44(1):193–242, 2015. doi:10.1137/130938438.

[42] Alexandra Henzinger, Matthew M. Hong, Henry Corrigan-Gibbs, Sarah Meiklejohn, and Vinod Vaikuntanathan. One server for the price of two: Simple and fast single-server private information retrieval. In Joseph A. Calandrino and Carmela Troncoso, editors, *32nd USENIX Security Symposium, USENIX Security 2023, Anaheim, CA, USA, August 9-11, 2023*, pages 3889–3905. USENIX Association, 2023.

[43] Camilla Hollanti and Neehar Verma. CB-cPIR: Code-based computational private information retrieval. *arXiv preprint arXiv:2505.03407*, 2025. Full version of [62]. Available at https://arxiv.org/pdf/2505.03407 (accessed January 19, 2026).

[44] Lukas Holzbaur, Camilla Hollanti, and Antonia Wachter-Zeh. Computational code-based single-server private information retrieval. In *2020 IEEE international symposium on information theory (ISIT)*, pages 1065–1070. IEEE, 2020.

[45] Yuval Ishai, Manoj Prabhakaran, and Amit Sahai. Secure arithmetic computation with no honest majority. *IACR Cryptol. ePrint Arch.*, page 465, 2008. URL: http://eprint.iacr.org/2008/465.

[46] Neeraj Kayal. The complexity of the annihilating polynomial. In *24th Annual IEEE Conference on Computational Complexity (CCC)*, pages 184–193. IEEE, 2009. doi:10.1109/CCC.2009.37.

[47] Iordanis Kerenidis and Ronald de Wolf. Exponential lower bound for 2-query locally decodable codes via a quantum argument. In Lawrence L. Larmore and Michel X. Goemans, editors, *Proceedings of the 35th Annual ACM Symposium on Theory of Computing, June 9-11, 2003, San Diego, CA, USA*, pages 106–115. ACM, 2003. doi:10.1145/780542.780560.

[48] Luana Kurmann, Svenja Lage, and Violetta Weger. Cryptanalysis of a pir scheme based on linear codes over rings, 2026. URL: https://arxiv.org/abs/2603.26409, arXiv:2603.26409.

[49] Eyal Kushilevitz and Rafail Ostrovsky. Replication is NOT needed: SINGLE database, computationally-private information retrieval. In *38th Annual Symposium on Foundations of Computer Science, FOCS 1997, Miami Beach, Florida, USA, October 19-22, 1997*, pages 364–373. IEEE Computer Society, 1997. doi:10.1109/SFCS.1997.646125.

[50] Svenja Lage and Hannes Bartz. On the security of a code-based PIR scheme. *arXiv preprint arXiv:2507.19295*, 2025. https://www.arxiv.org/pdf/2507.19295 (accessed January 19, 2026).

[51] Ueli Maurer. Abstract models of computation in cryptography. In Nigel Smart, editor, *Cryptography and Coding 2005*, volume 3796 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2005. doi:10.1007/11586821_1.

[52] Robert J. McEliece. A public-key cryptosystem based on algebraic coding theory. DSN Progress Report 42–44, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, 1978.

[53] Carlos Aguilar Melchor, Joris Barrier, Laurent Fousse, and Marc-Olivier Killijian. XPIR : Private information retrieval for everyone. *Proc. Priv. Enhancing Technol.*, 2016(2):155–174, 2016. doi:10.1515/POPETS-2016-0010.

[54] Samir Jordan Menon and David J. Wu. SPIRAL: fast, high-rate single-server PIR via FHE composition. In *43rd IEEE Symposium on Security and Privacy, SP 2022, San Francisco, CA, USA, May 22-26, 2022*, pages 930–947. IEEE, 2022. doi:10.1109/SP46214.2022.9833700.

[55] Samir Jordan Menon and David J. Wu. YPIR: high-throughput single-server PIR with silent preprocessing. In Davide Balzarotti and Wenyuan Xu, editors, *33rd USENIX Security Symposium, USENIX Security 2024, Philadelphia, PA, USA, August 14-16, 2024*. USENIX Association, 2024. URL: https://www.usenix.org/conference/usenixsecurity24/presentation/menon.

[56] Muhammad Haris Mughees, Hao Chen, and Ling Ren. OnionPIR: Response efficient single-server PIR. *IACR Cryptol. ePrint Arch.*, page 1081, 2021. URL: https://eprint.iacr.org/2021/1081.

[57] Moni Naor and Benny Pinkas. Oblivious transfer and polynomial evaluation. In Jeffrey Scott Vitter, Lawrence L. Larmore, and Frank Thomson Leighton, editors, *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing, May 1-4, 1999, Atlanta, Georgia, USA*, pages 245–254. ACM, 1999. doi:10.1145/301250.301312.

[58] Alexander A. Razborov and Sergey Yekhanin. An omega($n^{1/3}$) lower bound for bilinear group based private information retrieval. *Theory Comput.*, 3(1):221–238, 2007. doi:10.4086/TOC.2007.V003A012.

[59] Lajos Rónyai. Computing the structure of finite algebras. *Journal of Symbolic Computation*, 9(3):355–373, 1990. doi:10.1016/S0747-7171(08)80052-7.

[60] Victor Shoup. Lower bounds for discrete logarithms and related problems. In Walter Fumy, editor, *Advances in Cryptology—EUROCRYPT '97*, volume 1233 of *Lecture Notes in Computer Science*, pages 256–266. Springer, 1997. doi:10.1007/3-540-69053-0_18.

[61] Amir Shpilka and Amir Yehudayoff. Arithmetic circuits: A survey of recent results and open questions. *Found. Trends Theor. Comput. Sci.*, 5(3-4):207–388, 2010. doi:10.1561/0400000039.

[62] Neehar Verma and Camilla Hollanti. Code-based single-server private information retrieval: Circumventing the sub-query attack. In *IEEE International Symposium on Information Theory, ISIT 2024, Athens, Greece, July 7-12, 2024*, pages 2880–2885. IEEE, 2024. Conference version of [43].

[63] David Woodruff and Sergey Yekhanin. A geometric approach to information-theoretic private information retrieval. In *20th Annual IEEE Conference on Computational Complexity (CCC'05)*, pages 275–284. IEEE, 2005.

[64] Mingxun Zhou, Andrew Park, Wenting Zheng, and Elaine Shi. Piano: Extremely simple, single-server PIR with sublinear server computation. In *IEEE Symposium on Security and Privacy, SP 2024, San Francisco, CA, USA, May 19-23, 2024*, pages 4296–4314. IEEE, 2024. doi:10.1109/SP54263.2024.00055.

A Attacking PIR with General Arithmetic Circuits

Lemma 3.1 assumes an arithmetic PIR with *strictly arithmetic circuits*. In this section, we extend the proof of the lemma to arithmetic PIR with division gates and zero-check gates. While formal partial derivatives are well defined for rational functions (and accordingly to circuits with division gates), it is unclear how to define them for zerocheck gates. Nevertheless, we can cope with such gates via the machinery of [7].

A.1 Some useful lemmas

The following text is taken verbatim from [7].

Definition A.1 (The algorithm $\mathcal{T}$; Def. 3.13 in [7])**.** *The algorithm $\mathcal{T}$ takes as input an arithmetic circuit $C : \mathbb{F}^n \to \mathbb{F}^m$ defined over a prime[21] field $\mathbb{F}$, and a point $\hat{x} \in \mathbb{F}^n$. The algorithm $\mathcal{T}$ evaluates the circuit C on the point $\hat{x}$, and replaces each* zerocheck *gate g with the constant 1 or the constant 0 depending on the value that $\hat{x}$ induces on g. At the end, it outputs the resulting* zerocheck*-free circuit $C'(x)$.*

The following lemma follows from Proposition 3.12 of [7].

Lemma A.2. *Let $C : \mathbb{F}^n \to \mathbb{F}^m$ be an arithmetic circuit of size s that may use zerocheck gates and division gates but never divides by zero. Sample a random $\hat{x} \xleftarrow{R} \mathbb{F}^n$ and let $C'(x)$ be the circuit obtained from $\mathcal{T}(C, \hat{x})$. Then, the following hold:*

1. *For every $\hat{x}$, the circuit C' computes a rational function C'_1/C'_2 of degree $d \leq 2^s$ that agrees with C over $\hat{x}$. In particular, $C'_2(\hat{x}) \neq 0$.*
2. *Except with probability $\delta = s2^{s+1}/|\mathbb{F}|$ over $\hat{x}$, the output circuit C' agrees with C over all but δ-fraction of the inputs.*

Here we say that a rational function A/B is degree d if both A and B are degree-d polynomials. We will also need the following lemma whose proof follows from the Schwartz-Zippel lemma.

Lemma A.3. *Let $\mathbb{F}$ be a finite field, and let $A, B \in \mathbb{F}[x_1, \ldots, x_n]$ be polynomials of total degree at most d. Let $L = A/B$ be a rational function, and let $C \in \mathbb{F}[x_1, \ldots, x_n]$ be a polynomial of total degree at most c. Suppose that $L(x) = C(x)$ for all but an α-fraction of points $x \in \mathbb{F}^n$. If $|\mathbb{F}| > \frac{d+c}{1-\alpha}$, then the i-th formal partial derivatives of L and C are equal for each i, and their evaluations agree for all $x \in \mathbb{F}^n$ such that $B(x) \neq 0$.*

Proof. Define the polynomial $P(x) = A(x) - B(x)C(x)$. The total degree of P satisfies $\deg(P) \leq \deg(B) + \deg(C) \leq d + c$. By hypothesis, $L(x) = C(x)$ for a $1 - \alpha$ fraction of the domain $\mathbb{F}^n$. For the evaluation of the rational function $L(x) = A(x)/B(x)$ to be valid and equal to $C(x)$, it must hold that $B(x) \neq 0$ and $A(x) = B(x)C(x)$ on this set of points. Consequently, $P(x) = 0$ on at least a $1 - \alpha$ fraction

[21] In fact, everything in this section generalizes to the case of extension fields.

of $\mathbb{F}^n$. By the Schwartz-Zippel lemma, $P \equiv 0$ in the polynomial ring $\mathbb{F}[x_1, \ldots, x_n]$. This establishes the formal algebraic identity:

$$A \equiv BC.$$

Taking the i-th formal partial derivative for each $i \in \{1, \ldots, n\}$ and applying the product rule yields:

$$\partial_i A \equiv (\partial_i B)C + B(\partial_i C),$$

where ∂_i denotes the i-th formal partial derivative. Using the quotient rule and substituting A and $\partial_i A$ into this expression gives:

$$\begin{aligned}
\partial_i L &= \frac{(\partial_i A)B - A(\partial_i B)}{B^2} \\
&= \frac{\big[(\partial_i B)C + B(\partial_i C)\big]B - (BC)(\partial_i B)}{B^2} \\
&= \frac{(\partial_i B)CB + B^2(\partial_i C) - BC(\partial_i B)}{B^2} \\
&= \frac{B^2(\partial_i C)}{B^2} \equiv \partial_i C.
\end{aligned}$$

This proves the formal equivalence of the i-th partial derivatives for each i. To evaluate this equivalence over $\mathbb{F}^n$, observe that $\partial_i L(x)$ is well-defined if and only if $B(x) \neq 0$. Since the formal identity $\partial_i L \equiv \partial_i C$ holds globally, their evaluated outputs must match wherever both are defined. Thus, $\partial_i L(x) = \partial_i C(x)$ for each i and for all inputs $x \in \mathbb{F}^n$ where $B(x) \neq 0$. □

A.2 Proving the key lemma for generalized arithmetic circuits

We can now prove the following generalization of Lemma 3.1. Recall that for PIR $(\text{QUERY}_n^{\mathbb{F}}, \text{ANS}_n^{\mathbb{F}}, \text{DEC}_n^{\mathbb{F}})$ a query q is i-correct if there exists a randomness r for which $q = \text{QUERY}_n^{\mathbb{F}}(i; r)$ and

$$\Pr_x[\text{DEC}_n^{\mathbb{F}}(i, r, \text{ANS}_n^{\mathbb{F}}(x, q)) = x[i]] > 0.99.$$

Lemma A.4 (key Lemma (generalized))**.** *Let $(\text{QUERY}_n^{\mathbb{F}}, \text{ANS}_n^{\mathbb{F}}, \text{DEC}_n^{\mathbb{F}})$ be a single-server $\mathbb{F}$-arithmetic PIR with answer length $\ell(n)$ and circuit complexity of $s(n)$ where $\mathbb{F} = \mathbb{F}_p$ is a prime-order field of cardinality at least $10s2^{2s}$.*

Then, there exists an efficient randomized algorithm $\mathcal{M}$ that takes a query q and outputs a matrix $M_q \in \mathbb{F}^{\ell(n)\times n}$ such that for every index $i \in [n]$ and i-correct query q, except with probability $4s2^{2s}/|\mathbb{F}|$ over the randomness of $\mathcal{M}$, the rows of $M_q = \mathcal{M}(q)$ span the ith unit e_i.

Note that s is polynomial in n, and therefore the field size may be taken to be exponential in n. Consequently, each field element admits a $\text{poly}(n)$-size representation, and all field operations can be performed in $\text{poly}(n)$ time. Thus, although the field is large, it remains efficiently implementable.[22]

[22] We remind the reader that the goal of this appendix is primarily theoretical, namely, to rule out the existence of arithmetic PIR in a broader sense. Accordingly, it suffices to show that any such scheme can be broken over *some* efficiently implementable family of fields.

Proof. The attack is almost identical to the original one. Given a query q the algorithm $\mathcal{M}$ computes the matrix M as follows. Define the circuit $A(x) = \textsc{Ans}_n^{\mathbb{F}}(x, q)$. Sample a random database $\hat{x} \xleftarrow{R} \mathbb{F}^n$ and consider the zerocheck-free circuit $A' := \mathcal{T}(A, \hat{x})$. Let $\mathcal{J}A'(x) : \mathbb{F}^n \to \mathbb{F}^{\ell \times n}$ be the *formal Jacobian* matrix of $A'(x)$, i.e., $\mathcal{J}A'(x)$ is a matrix of rational functions whose (i, j)-th entry is the formal partial derivative of the i-th output of $A'(x) : \mathbb{F}^n \to \mathbb{F}^\ell$ with respect to the j-th input (the formal variable x_j). Compute the matrix of field elements $M = \mathcal{J}A'(\hat{x}) \in \mathbb{F}^{\ell \times n}$.

Analysis. Assume that q is i-correct and let r be the corresponding randomness. We show that e_i is spanned by M. Define the circuit $D(\mathbf{ans}) = \textsc{Dec}_{i,n}^{\mathbb{F}}(r, \mathbf{ans})$ and let $D' := \mathcal{T}(D, A(\hat{x}))$, where $A(\hat{x})$ is viewed as a point in $\mathbb{F}^\ell$. By assumption,

$$\Pr_x[D(A(x)) = x_i] \geq 0.99.$$

Let $\delta = (2s)2^{2s+1}/|\mathbb{F}| < 0.4$. By applying Lemma A.2 over the $2s$-size circuit $C(x) := D(A(x))$, we get that with probability $1 - \delta$ over $\hat{x}$, it holds that the circuit $C' := \mathcal{T}(C, \hat{x})$, satisfies

$$\Pr_x[C'(x) = D(A(x))] > 1 - \delta. \tag{5}$$

From now on, we condition on the event that Eq. 5 holds. By definition, $C'(x) = D'(A'(x))$ for every x, and therefore

$$D'(A'(x)) = x_i$$

for all but α-fraction of the inputs for $\alpha \leq (\delta + 0.01) < 0.5$. Recalling that D' and A' are degree-d rational functions for $d \leq 2^s$, the composition $L(x) := D'(A'(x))$ is a degree-d^2 rational function. Since $L(x) = x_i$ on all but an α-fraction of the inputs, and x_i is a degree-1 polynomial, Lemma A.3 implies that their formal derivatives agree. Hence, by the chain rule,

$$\partial D'(A'(x)) \cdot \mathcal{J}A'(x) = e_i,$$

for every x for which $D'(A'(x))$ is well-defined. The latter holds for $\hat{x}$ (by the first item of Lemma A.2), and since $M = \mathcal{J}A'(\hat{x})$, the lemma follows. □

Theorem A.5. *Every single-server arithmetic perfectly-correct (resp., statistically-correct) PIR with circuit complexity s over a prime-order field $\mathbb{F}$ of size $|\mathbb{F}| > 3n \cdot (4s2^{2s})$ has a trivial answer length of $\ell(n) \geq n$. Consequently, this also applies for every single-server arithmetic PIR.*

We did not attempt to optimize the size of the field in the above statement.

Proof. The proof of the first part is identical to the proof of the first part of Theorem 3.4, except that we use Lemma A.4 instead of Lemma 3.1, and reduce the success probability by $\frac{(4s2^{2s})}{|\mathbb{F}|} \leq 1/3n$, yielding a success probability of

$$\frac{1}{2} + \frac{1}{2n} - \frac{\epsilon}{2} - \frac{1}{3n} \geq \frac{1}{2} + \frac{1}{6n} - \frac{\epsilon}{2},$$

for negligible ϵ.

The proof of the second part now follows from the first part, just like in Theorem 3.4, except that we choose a prime-order field of size larger than $3n \cdot (4s2^{2s})$ where $s = \text{poly}(n)$ is the circuit size of the arithmetic-PIR algorithms. □

B PIR over Rings

Lemma 3.1 assumes an arithmetic PIR over finite fields. In this section, we extend the proof of the lemma to PIR over rings.

We first show that any arithmetic PIR over a ring $\mathbb{Z}_m$ with degree $d < p$ where m is an integer and p its largest prime divisor implies an arithmetic PIR over $\mathbb{F}_p$ with query length, answer length, and at most the same degree. We prove this by observing that since p divides m, the modulo p operation defines a surjective ring homomorphism from $\mathbb{Z}_m$ to $\mathbb{Z}_p$. We then apply Lemma 3.1 to the transformed $\mathbb{F}_p$-arithmetic PIR and conclude that any single server arithmetic PIR over a ring of integers modulo an arbitrary composite integer must have trivial communication cost. Finally, we extend the argument to commutative rings in Remark B.4.

Definition B.1 (Arithmetic PIR over Ring)**.** *A circuit implementation of a ring, an arithmetic circuit over a ring and an arithmetic PIR scheme over a ring are defined identically to definitions 2.1, 2.2 and 2.3 respectively, with the following modifications:*

1. *The underlying algebraic structure is over a ring $\mathbb{Z}_m$ where $m = \prod_{j=1}^{k} p_j^{e_j}$, p_j is prime and $e_j \geq 1$.*
2. *The* divide *and* zerocheck *operations are omitted from the allowed gates.*

Lemma B.2 (Modulo Reduction of Arithmetic PIR)**.** *Let m and p be integers such that p divides m. If there exists a $\mathbb{Z}_m$ arithmetic single-server (computationally-private) PIR with query length t, answer length ℓ, and degree $d < p$, then there exists an arithmetic single-server PIR scheme over $\mathbb{Z}_p$ with the same query length, answer length and degree $d' \leq d$. If the original scheme is perfectly (resp., statistically) correct then so is the new scheme.*

Proof. Let m and p be integers such that p divides m and let $(\textsc{Query}_n^{\mathbb{Z}_m}, \textsc{Ans}_n^{\mathbb{Z}_m}, \textsc{Dec}_n^{\mathbb{Z}_m})$ be the arithmetic circuits of a single-server $\mathbb{Z}_m$-arithmetic PIR with degree $d < p$. We transform each circuit into an arithmetic circuit over $\mathbb{Z}_p$ by modifying the values of the hard-coded ring elements into their modulo p match. The resulting circuit defines a new $\mathbb{Z}_p$-arithmetic PIR scheme with the same query length t and answer length ℓ. To prove correctness, we observe that modulo p operation is a surjective ring homomorphism from $\mathbb{Z}_m$ to $\mathbb{Z}_p$. Let $x \in \mathbb{Z}_p^n$ be the database and denote by $\hat{x} \in \mathbb{Z}_m^n$ a database such that $x = \hat{x} \pmod p$. Fix i and x, and let r be a random string for which correctness holds (with respect to i and x). Let $\hat{q} = \textsc{Query}_n^{\mathbb{Z}_m}(i, r)$ and $q = \textsc{Query}_n^{\mathbb{Z}_p}(i, r)$. Since, by definition, $q = \hat{q} \pmod p$, it holds that

$$\begin{aligned} x_i &= \hat{x}_i \bmod p \\ &= \textsc{Dec}_n^{\mathbb{Z}_m}(i, r, \textsc{Ans}_n^{\mathbb{Z}_m}(\hat{x}, \hat{q})) \bmod p \\ &= \textsc{Dec}_n^{\mathbb{Z}_p}(\textsc{Ans}_n^{\mathbb{Z}_p}(x, q)), \end{aligned}$$

Thus, the induced $\mathbb{Z}_p$-arithmetic PIR scheme has the same correctness error as the original scheme.

To prove privacy, we show how to map any efficient adversary $\mathcal{A}$ that wins the privacy game of the transformed $\mathbb{Z}_p$-arithmetic PIR with advantage δ into an efficient adversary $\mathcal{B}$ that wins the privacy game of the original $\mathbb{Z}_m$ scheme with the same advantage.

1. Upon receiving a challenge $\hat{q} \in \mathbb{Z}_m^t$, compute $q = \hat{q} \bmod p$.
2. Evaluate the adversary $\mathcal{A}$ with q and output whatever $\mathcal{A}$ outputs.

Because the modulo p operation is a homomorphism, sampling $\hat{q}$ and then taking the modulo p of $\hat{q}$ yields the same distribution as evaluating the $\mathbb{Z}_p$-circuit with the same randomness. Therefore, q is distributed identically to a legitimate query challenge in the transformed scheme. It follows that $\mathcal{B}$ wins with the same advantage as $\mathcal{A}$.

Finally, the degree of the composition of the decoder and answer circuits over the original scheme is d. Applying the modulo p operation can only decrease the degree of the polynomial. Therefore, the transformed PIR has degree $d' \leq d < p$ as well. We conclude that the transformed circuits define a valid arithmetic PIR over $\mathbb{Z}_p$ with degree $d' \leq d < p$, query length t and answer length ℓ. □

By combining Lemma B.2 with Theorem 3.4, we derive the following corollary.

Corollary B.3. *Let m be an integer and p its largest prime divisor. Every single-server perfectly correct (resp., statistically correct) arithmetic PIR over the ring $\mathbb{Z}_m$ with degree $d < p$ (resp., $d < 0.99p$) has a trivial answer length of $\ell(n) \geq n$.*

Note that we implicitly use the fact that the adversary can efficiently transform the $\mathbb{Z}_m$-PIR into $\mathbb{Z}_p$-PIR. Indeed, if p is not given as part of the description of the scheme, we can always hard-wire it to the adversary as a non-uniform advice.

Remark B.4 (Extension to other rings)**.** *We note that the reduction extends to any ring $\mathcal{R}$ that admits an efficiently computable surjective ring homomorphism onto a finite field $\mathbb{F}$. Such homomorphisms exist for most rings commonly used in cryptography, including polynomial quotient rings. More generally, every finite nontrivial commutative ring with unity admits a surjective ring homomorphism onto a finite field. Furthermore, when the ring is given explicitly by its addition and multiplication tables, such a homomorphism can be computed efficiently [59].*

C Arithmetic PRG

The following definition of an APRG is taken from [7]. This definition requires the output to be pseudorandom over $\mathbb{F}^\ell$ but allows the input (the seed) to contain both random field elements and random bits as long their total number is n.

Definition C.1 (Arithmetic Pseudorandom Generator)**.** *Let $\mathsf{APRG} = \{\mathsf{APRG}_n\}$ be a sequence of polynomial sized arithmetic circuits where APRG_n outputs a vector of $\ell(n)$ field elements and uses at most n random gates (some of them may be for random field elements and some may be for sampling random bits) and no deterministic input gates. We say that* APRG *is an* arithmetic pseudorandom generator *(APRG) if it satisfies the following two properties:*

1. *Expansion: $\ell > n$.*
2. *Pseudorandomness: The winning probability of every efficient adversary $\mathcal{A}$ in the following pseudorandomness game is at most $1/2 + \mathrm{negl}(n)$:*
   (a) *Given a security parameter 1^n, the adversary $\mathcal{A}$ picks a field implementation $\mathbb{F}$ and sends it to the challenger.*
   (b) *The challenger samples $b \xleftarrow{R} \{0,1\}$.*
      i. *If $b = 0$: the challenger sends to the adversary a sample $y \xleftarrow{R} \mathsf{APRG}_n$.*

ii. If $b = 1$: the challenger sends to the adversary a sample $u \xleftarrow{R} \mathbb{F}^\ell$.

(c) The adversary outputs b' and wins if $b' = b$.

An APRG is simple *if it does not contain division or* zerocheck *gates (but may use random bits). We will sometimes view the* APRG *as a mapping from the* seed s*, i.e., randomness sampled by the random gates, to the output, denoted by* $\mathsf{APRG}(s)$*. Note that the total length of s is n and it may consist of random field elements and random bits.*

APRG from the aLPN Assumption. It is shown in [7] that APRG with arbitrary polynomial stretch can be constructed based on the aLPN assumption (see Definition 7.2).

Theorem C.2 (Thm 9.3 of [7])**.** *Suppose that there exists some constants $p \in (0,1), \epsilon > 0$ for which the $(n, \ell = \frac{n}{1-p-\epsilon}, p)$-aLPN assumption holds. Then, there exists a simple APRG with an arbitrary polynomial output length.*

D Arithmetic Distributed Point Function

The use of Distributed Point Functions (DPFs) in computational two-server PIR is highly useful due to their compact key size and efficient evaluation. However, standard DPF constructions rely on non-arithmetic gates like XOR and require conversion between binary and field elements, in addition to assuming the existence of a binary PRG. We present an arithmetic DPF (ADPF) construction that uses only arithmetic operations defined under the existence assumption of an APRG.

Definition D.1 (Arithmetic Distributed Point Function)**.** *A distributed point function consists of the following two families of polynomial-sized arithmetic circuits, each parameterized by the number of possible evaluate indices n and the programmed index $\alpha \in [n]$:*

- *A randomized circuit $\textsc{Gen}_n^{\mathbb{F}}(\alpha, \beta; r)$ that given an index $\alpha \in [n]$ in binary form, a programmed value $\beta \in \mathbb{F}$ and the randomness $r = (r_A, r_B) \xleftarrow{R} \mathbb{F}^{\rho_A(n)} \times \{0,1\}^{\rho_B(n)}$ that can contain both random field elements and random bits outputs a pair of keys $(k_0, k_1) \in \mathbb{F}^{t(n)} \times \mathbb{F}^{t(n)}$ we refer to $t(n)$ as the key length.*
- *A deterministic Evaluation circuit $\textsc{Eval}_n^{\mathbb{F}}(k)$ that given an input key $k \in \mathbb{F}^{t(n)}$ outputs a vector $s \in \mathbb{F}^n$.*

We will typically write the index α as a subscript and write $\textsc{Gen}_{\alpha,n}(\beta; r)$ to emphasize that α is binary.
The algorithms should satisfy the following properties:

- *Correctness: For every field $\mathbb{F}$ and every pair $(\alpha, \beta) \in [n] \times \mathbb{F}$, consider the randomness r and let $(k_0, k_1) \leftarrow \textsc{Gen}_{\alpha,n}^{\mathbb{F}}(\beta; r)$, then the error probability (over randomness r)*

$$\Pr_r \left[\textsc{Eval}_n^{\mathbb{F}}(k_0) - \textsc{Eval}_n^{\mathbb{F}}(k_1) \neq \beta \cdot \mathbf{1}_\alpha \right]$$

  is negligible in n. When the error is zero, we say that the DPF is perfectly correct.

- *Security: The winning probability of every efficient adversary $\mathcal{A}$ in the following pseudorandomness game is at most $1/2 + \mathrm{negl}(n)$:*

1. *Given a security parameter 1^n, the adversary $\mathcal{A}$ picks a field implementation $\mathbb{F}$, a bit σ and two pairs $(\alpha_0, \beta_0) \in [n] \times \mathbb{F}$ and $(\alpha_1, \beta_1) \in [n] \times \mathbb{F}$ and sends these values to the challenger.*
2. *The challenger samples a bit $b \xleftarrow{R} \{0,1\}$, and keys $(k_0, k_1) \xleftarrow{R} \text{GEN}_n^{\mathbb{F}}(\alpha_b, \beta_b)$, and sends to the adversary the key k_σ.*
3. *The adversary outputs b' and wins if $b' = b$.*

Our ADPF is a modification of the Distributed Point Function defined in [20].

Construction D.2 (Arithmetic Distributed Point Function)**.** *Let λ be the security parameter of the construction and let $G : \mathbb{F}^\lambda \mapsto \mathbb{F}^{2\lambda+2}$ be an arithmetic pseudorandom Generator (APRG). We define the scheme for a block-point function $f_{\alpha,\beta} \in (\mathbb{F}^\lambda)^n$, whose αth block is equal to $\beta \in \mathbb{F}^\lambda$, where the target index α has a bit-length $d = \log n$. We refer to α_i as the ith bit of α.The scalar case where $\hat{\beta} \in \mathbb{F}$ follows immediately by defining the block β as $\beta = (0, \dots, \hat{\beta}, 0, \dots) \in \mathbb{F}^\lambda$ where the position of $\hat{\beta}$ is determined by the last $\log \lambda$ bits of α.*

- *The generation algorithm $\text{GEN}_{\alpha,n}^{\mathbb{F}}(\beta; r)$:*

  1. *Parses the randomness $r \in \mathbb{F}^{2\lambda}$ as the initial shared seeds $(s_0^{(0)}, s_1^{(0)}) \in (\mathbb{F}^\lambda)^2$.*
  2. *Let $t_0^{(0)} \xleftarrow{R} \mathbb{F}, t_1^{(0)} := t_0^{(0)} - 1$ be the control elements initial values.*
  3. *For $i = 1$ to d do:*
     (a) *For server $b \in \{0,1\}$, compute the temporary expanded values*
     $$(s'^{(0)}_b, t'^{(0)}_b, s'^{(1)}_b, t'^{(1)}_b) := G(s_b^{(i-1)}). \tag{6}$$
     (b) *Let $z := \alpha_i$ be the "on-path" bit and $\bar{z} := 1 - \alpha_i$ the "off-path" bit.*
     (c) *Compute correction word:*
        i. *Set $s_{\mathsf{CW}} := s'^{(\bar{z})}_1 - s'^{(\bar{z})}_0$.*
        ii. *Set $t^{(z)}_{\mathsf{CW}} := 1 + t'^{(z)}_1 - t'^{(z)}_0$ and $t^{(\bar{z})}_{\mathsf{CW}} := t'^{(\bar{z})}_1 - t'^{(\bar{z})}_0$.*
        iii. *$\mathsf{CW}^{(i)} := (s_{\mathsf{CW}}, t^{(0)}_{\mathsf{CW}}, s_{\mathsf{CW}}, t^{(1)}_{\mathsf{CW}}) \in \mathbb{F}^{2\lambda+2}$.*
     (d) *Update active path: For $b \in \{0,1\}$, set:*
        i. *$s_b^{(i)} := s'^{(z)}_b + t_b^{(i-1)} \cdot s_{\mathsf{CW}}$.*
        i. *$t_b^{(i)} := t'^{(z)}_b + t_b^{(i-1)} \cdot t^{(z)}_{\mathsf{CW}}$.*
  4. *Compute the final correction word, $\mathsf{CW}^{(d+1)} := \beta - s_0^{(d)} + s_1^{(d)} \in \mathbb{F}^\lambda$.*
  5. *Let $k_\sigma := (s_\sigma^{(0)}, t_\sigma^{(0)}, \mathsf{CW}^{(1)}, \dots, \mathsf{CW}^{(d+1)})$ for $\sigma \in \{0,1\}$.*
  6. *Return the keys (k_0, k_1).*

- *The evaluation algorithm $\text{EVAL}_n^{\mathbb{F}}(k_\sigma)$:*

  1. *Parse the key $k_\sigma = (s^{(0)}, t^{(0)}, \mathsf{CW}^{(1)}, \dots, \mathsf{CW}^{(d+1)})$.*
  2. *Initialize new array $S^{(0)}, T^{(0)}$ of size 1, representing the root level such that $S^{(0)}[0] = s^{(0)}$ and $T^{(0)}[0] = t^{(0)}$.*
  3. *For $i = 1$ to d do:*

(a) Initialize new arrays $S^{(i)}, T^{(i)}$ of size 2^i.

(b) For each node index j from 0 to $2^{i-1}-1$:

i. Expand the parent seed,

$$(S^{(i)}[2j], T^{(i)}[2j], S^{(i)}[2j+1], T^{(i)}[2j+1]) := G(S^{(i-1)}[j]) + T^{(i-1)}[j] \cdot \mathsf{CW}^{(i)}.$$

4. Initialize new array $V \in (\mathbb{F}^\lambda)^n$ the final vector.

5. For each node index j from 0 to 2^d-1:

(a) Set $V[j] := S^{(d)}[j] + T^{(d)}[j] \cdot \mathsf{CW}^{(d+1)}$.

6. Return the vector V.

Theorem D.3. *Construction D.2 is perfectly correct.*

Proof. Let $\alpha = (\alpha_1, \ldots, \alpha_d)$ denote the index passed as input to GEN and let r be the randomness used by the generation algorithm to create the keys k_0 and k_1. During evaluation, the servers generate arrays of seeds $S_0^{(i)}$, $S_1^{(i)}$ and control elements $T_0^{(i)}, T_1^{(i)}$ at each depth $i \in [d]$, where $S_0^{(0)}, S_1^{(0)}$ are initialized according to the initial seeds derived from the randomness and $T_0^{(0)}, T_1^{(0)}$ are initialized as a secret sharing of 1. We refer to the output vector of server b as V_b.

Let $\Delta S^{(i)} = S_0^{(i)} - S_1^{(i)}$ denote the seed difference, similarly $\Delta T^{(i)}$ denotes the control difference, and ΔV denotes the evaluated vectors difference. For every $i \in \{0, \ldots, d\}$ and $j \in \left\{0, \ldots, 2^i - 1\right\}$, we say that index j is *on-path* at depth i if its binary representation is $(\alpha_1 \ldots \alpha_i)$ and say that it is *off-path* otherwise. (For $i = 0$ we think of $j = 0$ as being on-path.) We prove that the following invariant holds for any "depth" i.

Claim D.4. *For every $i \in \{0, \ldots, d\}$, and every index $j \in \left\{0, \ldots, 2^i - 1\right\}$: If j is off-path at depth i then the corresponding seed/control differences are zero, i.e.,*

$$\Delta S^{(i)}[j] = 0 \quad \textit{and} \quad \Delta T^{(i)}[j] = 0.$$

If j is on-path at depth i then the corresponding control difference is one $\Delta T^{(i)}[j] = 1$ (and the seed difference $\Delta S^{(i)}[j]$ may be arbitrary).

Proof of Claim. By induction on i. The base case ($i = 0, j = 0$) follows by construction since the generation algorithm samples two independent initial seeds $s_0^{(0)}$ and $s_1^{(0)}$, and sets $t_1^{(0)} = t_0^{(0)} - 1$, thus, $\Delta T^{(0)} = 1$. (Note that $\Delta T^{(0)} = \Delta T^{(0)}[0]$.) For the induction step, assume that the claim holds for depth $i-1$. Fix index $j \in \left\{0, \ldots, 2^{i-1} - 1\right\}$ it suffices to prove the claim for depth i and for the indices $2j$ and $2j+1$.

Case 1: j is off-path. By our assumption, $\Delta S^{(i-1)}[j] = 0$ and $\Delta T^{(i-1)}[j] = 0$. Because the seeds are identical the APRG produced the same output on both servers. Since, the control difference is 0, the correction word $\mathsf{CW}^{(i)}$ is scaled in the same way on both servers. Therefore, both servers output the same seed and control bit at location j which satisfies $\Delta S^{(i)}[2j] = 0, \Delta S^{(i)}[2j+1] = 0$ and $\Delta T^{(i)}[2j] = 0, \Delta T^{(i)}[2j+1] = 0$, as required.

Case 2: j is on-path. By our assumption, $\Delta T^{(i-1)}[j] = 1$ and the seeds $S_0^{(i-1)}[j], S_1^{(i-1)}[j]$ may be arbitrary. For server $b \in \{0,1\}$, denote the APRG expanded seed by $(s_b^{(0)}, t_b^{(0)}, s_b^{(1)}, t_b^{(1)}) = G(S_b^{(i-1)}[j])$. Without loss of generality, assume that the left child is on-path (i.e. $A^{(i)} = 2j$). Thus, the generation algorithm programmed $\mathsf{CW}^{(i)} = s_{\mathsf{CW}}, t_{\mathsf{CW}}^{(0)}, s_{\mathsf{CW}}, t_{\mathsf{CW}}^{(1)}$, where $s_{\mathsf{CW}} = s_1^{(1)} - s_0^{(1)}$, $t_{\mathsf{CW}}^{(0)} = 1 + t_1^{(0)} - t_0^{(0)}$ and $t_{\mathsf{CW}}^{(1)} = t_1^{(1)} - t_0^{(1)}$. Indeed, for the off-path child:

$$\begin{aligned}\Delta S^{(i)}[2j+1] &= (s_0^{(1)} + T_0^{(i-1)}[j] \cdot s_{\mathsf{CW}}) - (s_1^{(1)} + T_1^{(i-1)}[j] \cdot s_{\mathsf{CW}}) \\ &= -s_{\mathsf{CW}} + \Delta T^{(i-1)}[j] \cdot s_{\mathsf{CW}} \\ &= 0,\end{aligned}$$

and similarly,

$$\begin{aligned}\Delta T^{(i)}[2j+1] &= (t_0^{(1)} + T_0^{(i-1)}[j] \cdot t_{\mathsf{CW}}^{(1)}) - (t_1^{(1)} + T_1^{(i-1)}[j] \cdot t_{\mathsf{CW}}^{(1)}) \\ &= t_0^{(1)} - t_1^{(1)} + \Delta T^{(i-1)}[j] \cdot t_{\mathsf{CW}}^{(1)} \\ &= -t_{\mathsf{CW}}^{(1)} + t_{\mathsf{CW}}^{(1)} = 0.\end{aligned}$$

This perfectly maintains the off-path invariant. On the other hand, for the on-path child:

$$\begin{aligned}\Delta T^{(i)}[2j] &= (t_0^{(0)} + T_0^{(i-1)}[j] \cdot t_{\mathsf{CW}}^{(0)}) - (t_1^{(0)} + T_1^{(i-1)}[j] \cdot t_{\mathsf{CW}}^{(0)}) \\ &= t_0^{(0)} - t_1^{(0)} + \Delta T^{(i-1)}[j] \cdot t_{\mathsf{CW}}^{(0)} \\ &= t_0^{(0)} - t_1^{(0)} + (1 + t_1^{(0)} - t_0^{(0)}) = 1.\end{aligned}$$

This perfectly maintains the on-path invariant, and the claim follows. □

We can now complete the proof of the theorem. At depth d, the invariant guarantees that the seed and control differences for every $j \neq \alpha$ are exactly zero (i.e. $\Delta T^{(d)}[j] = \Delta S^{(d)}[j] = 0$). At index α, the control difference is $\Delta T^{(d)}[\alpha] = 1$, and the seed difference is $\Delta S^{(d)}[\alpha] = s_0^{(d)} - s_1^{(d)}$. The GEN algorithm sets the final correction word $\mathsf{CW}^{(d+1)} = \beta - \Delta S^{(d)}[\alpha]$. Hence, $\Delta V[j]$ evaluates to:

$$\begin{aligned}\Delta V[j] &= (S_0^{(d)}[j] + T_0^{(d)}[j] \cdot \mathsf{CW}^{(d+1)}) - (S_1^{(d)}[j] + T_1^{(d)}[j] \cdot \mathsf{CW}^{(d+1)}) \\ &= \Delta S^{(d)}[j] + \Delta T^{(d)}[j] \cdot \mathsf{CW}^{(d+1)} \\ &= \Delta S^{(d)}[j] + \Delta T^{(d)}[j] \cdot (\beta - \Delta S^{(d)}[\alpha]) \\ &= \begin{cases} \beta, & j = \alpha \\ 0, & \text{otherwise.} \end{cases}\end{aligned}$$

We conclude that $\Delta V = \beta \cdot \mathbf{1}_\alpha$. □

Theorem D.5. *For every $\lambda(n)$, if the APRG is secure against $\mathrm{poly}(n)$-size circuits then Construction D.2 is a secure ADPF. In particular, assuming that the APRG is secure against polynomial-size circuits, we can take $\lambda = n^c$ for arbitrarily small constant $c > 0$.*

Proof. Assume, towards a contradiction, that there exists an efficient adversary $\mathcal{A}$ that wins the security game with probability $0.5 + \epsilon(n)$ for some non-negligible advantage $\epsilon(n)$. We will show how to construct an adversary $\mathcal{A}'$ that breaks the APRG with non-negligible advantage $\epsilon(n)/2d$.

For security parameter 1^λ, let $\mathbb{F}$ denote the field implementation chosen by $\mathcal{A}$ and let $\sigma \in \{0,1\}$ and $(\alpha_0, \beta_0) \in [n] \times \mathbb{F}$ and $(\alpha_1, \beta_1) \in [n] \times \mathbb{F}$ be the other values that are selected by $\mathcal{A}$. The adversary $\mathcal{A}'$ uses the same field implementation $\mathbb{F}$, and given a challenge string $w \in \mathbb{F}^{2\lambda+2}$ (which is either uniform or pseudorandom) proceeds as follows.

1. Sample a challenge bit $b \in \{0,1\}$ and an index $i \in [2d]$.
2. Sample a pair of keys (k_0, k_1) by invoking a modified version of $\textsc{Gen}_n^{\mathbb{F}}(\alpha_b, \beta_b; r)$ in which the first $i-1$ calls to the APRG (Eq. (6)) are replaced by uniform strings, and the ith call is replaced by the challenge string w.
3. Return to $\mathcal{A}$ the key k_σ and get back its guess b'. Output 1 (pseudorandom) if $b = b'$; and 0 otherwise.

The analysis is standard. For every $i \in \{0, \ldots, 2d\}$ let H^i denote the distribution of k_σ obtained by calling $\textsc{Gen}_n^{\mathbb{F}}(\alpha_b, \beta_b; r)$ with the first i calls to the APRG replaced by uniform strings and where all other calls are generated via the APRG (and where b is chosen at random). Let w^i denote the probability that $\mathcal{A}$ guesses b correctly. Then winning advantage of $\mathcal{A}'$ is

$$\frac{1}{2d} \sum_{i=1}^{2d} (w^{i-1} - w^i) = \frac{w^0 - w^{2d}}{2d}.$$

By assumption $w^0 = 0.5 + \epsilon(n)$, and so it remains to show that $w^{2d} = 0.5$. To see this, recall that in H^{2d} all the APRG calls are replaced with uniform strings and in this case the key k_σ is statistically independent of b. Indeed,

$$k_\sigma := (s_\sigma^{(0)}, t_\sigma^{(0)}, \mathsf{CW}^{(1)}, \ldots, \mathsf{CW}^{(d+1)}),$$

where $s_\sigma^{(0)}$ and $t_\sigma^{(0)}$ are uniform and the correction words generated are of the form (a, b, a, c) where a, b, c are uniformly generated. □

We conclude this section with an analysis of the key size. The tree generated during construction has depth $d = \log n$. At each level $i \in [d]$ the correction word consists of two seeds and two control elements masks ($2\lambda + 2$ field elements). The final correction word $\mathsf{CW}^{(d+1)}$ is a block of λ elements. Including the initial seed and control element, the total key size in field elements is:

$$(\lambda + 1) + d \cdot (2\lambda + 2) + \lambda = O(\lambda \log n).$$