Algebraic Restriction Codes and their Applications

Divesh Aggarwal[1], Nico Döttling[2], Jesko Dujmovic[2,3],
Mohammad Hajiabadi[4], Giulio Malavolta[5], and Maciej Obremski[1]

[1] National University of Singapore
[2] Helmholtz Center for Information Security (CISPA)
{doettling, jesko.dujmovic}@cispa.de
[3] Saarland University
[4] University of Waterloo
[5] Max Planck Institute for Security and Privacy

Abstract. Consider the following problem: You have a device that is supposed to compute a linear combination of its inputs, which are taken from some finite field. However, the device may be faulty and compute arbitrary functions of its inputs. Is it possible to encode the inputs in such a way that only linear functions can be evaluated over the encodings? I.e., learning an arbitrary function of the encodings will not reveal more information about the inputs than a linear combination.
In this work, we introduce the notion of *algebraic restriction codes* (AR codes), which constrain adversaries who might compute any function to computing a linear function. Our main result is an information-theoretic construction AR codes that restrict any class of function with a bounded number of output bits to linear functions. Our construction relies on a seed which is not provided to the adversary.
While interesting and natural on its own, we show an application of this notion in cryptography. In particular, we show that AR codes lead to the first construction of rate-1 oblivious transfer with statistical sender security from the Decisional Diffie-Hellman assumption, and the first-ever construction that makes black-box use of cryptography. Previously, such protocols were known only from the LWE assumption, using non-black-box cryptographic techniques. We expect our new notion of AR codes to find further applications, e.g., in the context of non-malleability, in the future.

1 Introduction

In this work, we consider leakage problems of the following kind: Assume we have a device which takes an input x and *is supposed to* compute a function $f(x)$ from a certain class of legitimate functions $\mathcal{F}$. For concreteness, assume that the class $\mathcal{F}$ consists of functions computing linear combinations, e.g., $f(x_1, x_2) = a_1 x_1 + a_2 x_2$. However, the device might be faulty, and instead of computing f it might compute another function g. We want to find a way to encode x into an $\hat{x}$ such that the following two properties hold:

- If the device correctly implements a linear function f, then we can efficiently decode the output y to $f(x)$.
- If, on the other hand, the device implements a non-linear function g, then the output $g(\hat{x})$ does not reveal more information about x than $f(x)$ for some linear function f.

First, note that this notion is trivially achievable if $\mathcal{F}$ includes the identity function, or in fact any invertible function, as in this case we can simulate $g(\hat{x})$ from $f(x)$ by first recovering x from $f(x)$, encoding x to $\hat{x}$ and finally evaluating g on $\hat{x}$. For this reason, in this work, we will focus on function classes $\mathcal{F}$ whose output-length is smaller than their input-length, such as the linear combination functions mentioned above. In general, we will allow both the encoding and decoding procedure to depend on a secret seed, which is not given to the evaluating device/adversary.

It is worthwhile comparing the type of security this notion provides to tamper-resilient primitives such as *non-malleable codes* (NM-codes) [DPW10, DKO13, ADL14] and non-malleable extractors [DW09, DLWZ11, Li12, CG14]. Such notions are geared towards prohibiting tampering altogether. Moreover, a central aspect for security for such notions is that the decoder tries to detect if some tampering happened, and indeed the decoder plays a crucial role in modelling the security of non-malleable codes. In contrast, AR codes do and should allow manipulation by benign functions from the class $\mathcal{F}$. Furthermore, we only require a decoder for correctness purposes, whereas security is defined independently of the decoder.

One motivation to study the above problem comes from cryptography, specifically secure computation, where this is, in fact, a natural scenario. Indeed, a typical blueprint for secure two-party computation [Yao82] in two rounds proceeds as follows: One party, called the receiver, encrypts his input y under a homomorphic encryption scheme [Pai99, Gen09, BV11, GSW13] obtaining a ciphertext c, and sends both the public key pk and the ciphertext c to the other party, called the sender. The sender, in possession of an input x *homomorphically performs a computation* f on input x and ciphertext c, obtaining a ciphertext c' which encrypts $f(x, y)$. The ciphertext c' is sent back to the receiver who can then decrypt it to $f(x, y)$.

For the case of a malicious receiver, the security of this blueprint breaks down completely: A malicious receiver can choose both the public key pk and the ciphertext c maliciously, i.e. they are generally not well-formed. Effectively, this means that the sender's homomorphic evaluation will result in some value $\tilde{f}(x)$ (where $\tilde{f}$ will be specified by the receiver's protocol message) instead of an encryption of $f(x, y)$. Critically, the value $\tilde{f}(x)$ might reveal substantially more information about x than $f(x, y)$ and compromise the sender's security.

Generally speaking, in this situation, there is no direct way for the sender to enforce which information about x the receiver obtains. A typical cryptographic solution for achieving malicious security involves using zero-knowledge proofs to enforce honest behavior for the receiver. This technique, however, is typically undesirable as it often leads to less efficient protocols (due to these tools using non-black-box techniques) and the need for several rounds of interaction or a trusted setup. We aim to upgrade such protocols to achieve security against malicious receivers without additional cryptographic machinery.

To see how algebraic restriction codes will help in this scenario, consider the following. Upon receiving a public key pk and a ciphertext c from the receiver (who potentially generated them in a malicious way) the sender proceeds as follows. First, he encodes his own input x into $\hat{x}$ using a suitable AR code with a fresh seed s. Next, also then sender evaluates the function $f(\hat{x}, \cdot)$ homomorphically on the ciphertext c (which encrypts the receiver's input y), resulting in a ciphertext $c' = \mathsf{Eval}(\mathsf{pk}, f(\hat{x}, \cdot), c)$. For simplicity's sake, assume that the sender now sends c' and the seed s back to the receiver, who decrypts c' to $\hat{z} = f(\hat{x}, y)$ and uses the seed s to decode $\hat{z}$ to his output z using the decoding algorithm of the AR code.

How can we argue that even a malicious receiver cannot learn more than the legitimate output z? Let's take a closer look on the computation which is actually performed on the encoding $\hat{x}$. The output ciphertext c' is computed via $c' = \mathsf{Eval}(\mathsf{pk}, f(\hat{x}, \cdot), c)$. Thus, if we can assure that the function $g(\hat{x}) = \mathsf{Eval}(\mathsf{pk}, f(\hat{x}, \cdot), c)$ is in the class $\mathcal{G}$ which is restricted by the AR code, then security of the AR code guarantees that c' does not leak more than $z = f(x, y)$ about x, irrespective of the choice of pk and c.

1.1 Our Results

In this work, we formalize the notion of algebraic restriction codes and provide constructions which restrict general function classes to linear functions over finite fields. Let $\mathcal{G}$ and $\mathcal{F}$ be two function classes. Roughly, a $\mathcal{G}$-$\mathcal{F}$ AR code provides a way to encode an x in the domain of the functions in $\mathcal{F}$ into a codeword $\hat{x}$ in the domain of the functions in $\mathcal{G}$, in a way that any function $f \in \mathcal{F}$ can still be evaluated on $\hat{x}$, by evaluating a function $f' \in \mathcal{G}$ on $\hat{x}$. Furthermore, given $f'(\hat{x})$ we can decode to $f(x)$. Security-wise, we require that for any $g \in \mathcal{G}$ there exists a function $f \in \mathcal{F}$, such that $g(\hat{x})$ can be simulated given only *the legitimate output* $f(x)$. AR codes provide an *information-theoretic interface* to limit the capabilities of an unbounded adversary in protocols in which some weak restrictions (characterized by the class $\mathcal{G}$) are already in place. In this way, AR codes will allow us to harness simple structural restrictions of protocols to implement very strong security guarantees.

In this work we consider *seeded* AR codes, where both the encoding and decoding procedures of the AR code have access to a random seed s, which is not provided to the function g.

Our first construction of AR-codes restricts general linear functions to linear combinations.

Theorem 1 (Formal: Theorem 4, Page 14). *Let $\mathbb{F}_q$ be a finite field, let $\mathcal{F}$ be the class of functions $\mathbb{F}_q^k \times \mathbb{F}_q^k \to \mathbb{F}_q^k$ of the form $(\mathbf{x}, \mathbf{y}) \mapsto a\mathbf{x} + \mathbf{y}$, and let $\mathcal{G}$ be the class of all linear functions $\mathbb{F}_q^n \times \mathbb{F}_q^n \to \mathbb{F}_q^n$ of the form $(\mathbf{x}, \mathbf{y}) \mapsto \mathbf{A}\mathbf{x} + \mathbf{y}$. There exists a seeded AR code AR_1 which restricts $\mathcal{G}$ to $\mathcal{F}$.*

Our main contribution is a construction of seeded AR codes restricting arbitrary functions with bounded output length to linear combinations.

Theorem 2 (Formal: Theorem 5, Page 19). *Let $\mathbb{F}_q$ be a finite field, let $\mathcal{F}$ be the class of functions $\mathbb{F}_q \times \mathbb{F}_q \to \mathbb{F}_q$ of the form $(x, y) \mapsto ax + by$, and let $\mathcal{G}$ be the class of* all *functions $\mathbb{F}_q^n \times \mathbb{F}_q^n \to \{0,1\}^{1.5 \cdot n \log(q)}$. There exists a seeded AR code AR_2 which restricts $\mathcal{G}$ to $\mathcal{F}$.*

We note that the constant 1.5 in the theorem is arbitrary and can in fact be replaced with any constant between 1 and 2.

The main ingredient of this construction is the following theorem, which may be of independent interest and which we will discuss in some greater detail. The theorem exhibits a new correlation-breaking property of the inner-product extractor.

In essence, it states that for a suitable parameter choice, if $\mathbf{x}_1, \ldots, \mathbf{x}_t$ are uniformly random vectors in a finite vector space and $\mathbf{s}$ is a random seed (in the same vector space), then anything that can be inferred about the $\langle \mathbf{x}_1, \mathbf{s} \rangle, \ldots, \langle \mathbf{x}_t, \mathbf{s} \rangle$ via a *joint leak* $f(\mathbf{x}_1, \ldots, \mathbf{x}_t)$ of bounded length can also be inferred from a linear combination $\sum_i a_i \langle \mathbf{x}_i, \mathbf{s} \rangle$, i.e. $f(\mathbf{x}_1, \ldots, \mathbf{x}_t)$ does not leak more than $\sum_i a_i \langle \mathbf{x}_i, \mathbf{s} \rangle$.

Theorem 3 (Formal: Theorem 5, Page 19). *Let q be a prime power, let t, s be positive integers, and $\varepsilon > 0$ and $n = O(t + s/\log(q) + (\log \frac{1}{\varepsilon})/\log(q))$. Let $\mathbf{x}_1, \ldots, \mathbf{x}_t$ be uniform in $\mathbb{F}_q^n$ and $\mathbf{s}$ is uniform in $\mathbb{F}_q^n$ and independent of the $\mathbf{x}_i$. For any $f : \mathbb{F}_q^{tn} \to \{0,1\}^{n \log q + s}$, there exists a simulator Sim and random variables $a_1, \ldots, a_t \in \mathbb{F}_q$ such that*

$$\begin{aligned}&\mathbf{s}, f(\mathbf{x}_1, \ldots, \mathbf{x}_t), \langle \mathbf{x}_1, \mathbf{s} \rangle, \ldots, \langle \mathbf{x}_t, \mathbf{s} \rangle, a_1, \ldots, a_t \\ \approx_{2\varepsilon} \; &\mathsf{Sim}\left(\mathbf{s}, a_1, \ldots, a_t, \sum_{i=1}^t a_i u_i\right), u_1, \ldots, u_t, a_1, \ldots, a_t\end{aligned}$$

where $u_1, \ldots, u_t$ are uniform and independent random variables in $\mathbb{F}_q$, independent of $(a_1, \ldots, a_t)$.

One way to interpret the theorem is that the inner product extractor breaks all correlations (induced by a leak $f(\mathbf{x}_1, \ldots, \mathbf{x}_t)$), except linear ones. Recall that our notion of AR codes it is crucial that linear relations are preserved.

We then demonstrate an application of AR codes in upgrading the security of oblivious transfer (OT) protocols while simultaneously achieving optimal communication, a question that had remained opened due to insurmountable difficulties, explained later. Specifically, we obtain the first rate-1 OT protocol with statistical sender privacy from the decisional Diffie Hellman (DDH) assumption. While our motivation to study AR codes is to construct efficient and high rate statistically sender private OT protocols, we expect AR codes and in particular the ideas used to construct them to be useful in a broader sense.

2 Technical Outline

In what follows, we provide an informal overview of the techniques developed in this work.

2.1 Warmup: Algebraic Restriction Codes for General Linear Functions

Before discussing the ideas leading up to our main result, we will first discuss the instructive case of AR codes restricting general linear functions to *simple* linear functions. Specifically, fix a finite field $\mathbb{F}_q$ and let $\mathcal{G}$ be the class of linear functions $\mathbb{F}_q^{2m} \to \mathbb{F}_q^m$ of the form $g(\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2) = \mathbf{A}\hat{\mathbf{x}}_1 + \hat{\mathbf{x}}_2$, where $\mathbf{A} \in \mathbb{F}_q^{m \times m}$ is an arbitrary matrix. We want to restrict $\mathcal{G}$ to the class $\mathcal{F}$ consisting of linear functions $\mathbb{F}_q^{2n} \to \mathbb{F}^n$ of the form $f(\mathbf{x}_1, \mathbf{x}_2) = a \cdot \mathbf{x}_1 + \mathbf{x}_2$, where $a \in \mathbb{F}_q$ is a scalar.

Our construction proceeds as follows. The seed s specifies a random matrix $\mathbf{R} \in \mathbb{F}_q^{n \times m}$, such a matrix has full rank except with probability $\leq 2^{-(m-n)}$. To encode a pair of input vectors $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{F}_q^n$, the encoder samples uniformly random $\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2 \leftarrow_\$ \mathbb{F}_q^m$ such that $\mathbf{R}\hat{\mathbf{x}}_1 = \mathbf{x}_1$ and $\mathbf{R}\hat{\mathbf{x}}_2 = \mathbf{x}_2$, and outputs the codeword $(\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2)$. To evaluate a scalar linear function given by $a \in \mathbb{F}_q$ on such a codeword, we (unsurprisingly) compute $\hat{\mathbf{y}} = a\hat{\mathbf{x}}_1 + \hat{\mathbf{x}}_2$. To decode $\hat{\mathbf{y}}$ we compute $\mathbf{y} = \mathbf{R}\hat{\mathbf{y}}$. Correctness of this AR code construction follows routinely:

$$\mathbf{y} = \mathbf{R}\hat{\mathbf{y}} = \mathbf{R}(a\hat{\mathbf{x}}_1 + \hat{\mathbf{x}}_2) = \mathbf{R}a\hat{\mathbf{x}}_1 + \mathbf{R}\hat{\mathbf{x}}_2 = a\mathbf{R}\hat{\mathbf{x}} + \mathbf{R}\hat{\mathbf{x}}_2 = a\mathbf{x}_1 + \mathbf{x}_2.$$

i.e. correctness holds as the scalar a commutes with the matrix $\mathbf{R}$.

In this case it will also be more convenient to look at the problem from the angle of randomness extraction; Specifically, assume that $\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2 \leftarrow_\$ \mathbb{F}_q^m$ are chosen uniformly random. We want to show that for any matrix $\mathbf{A} \in \mathbb{F}_q^{m \times m}$ anything that can be learned about $\mathbf{R}\hat{\mathbf{x}}_1$ and $\mathbf{R}\hat{\mathbf{x}}_2$ from $\mathbf{A}\hat{\mathbf{x}}_1 + \hat{\mathbf{x}}_2$ can also be learned from $a \cdot \mathbf{R}\hat{\mathbf{x}}_1 + \mathbf{R}\hat{\mathbf{x}}_2$ for some $a \in \mathbb{F}_q$.

How can we find such an a for any given $\mathbf{A}$? First notice that if $\hat{\mathbf{x}}_1$ happens to be an eigenvector of $\mathbf{A}$ with respect to an eigenvalue a_i, then it indeed holds that $\mathbf{A}\mathbf{x}_1 + \mathbf{x}_2 = a_i\mathbf{x}_1 + \mathbf{x}_2$. Thus, a reasonable approach is to set the extracted scalar $a \in \mathbb{F}_q$ to one of the eigenvalues of $\mathbf{A}$ (or 0 if there are no eigenvalues). If the matrix $\mathbf{A}$ has several distinct eigenvalues a_i, we will set a to be the eigenvalue whose eigenspace V_i has maximal dimension. Note that since the sum of the dimensions of all eigenspaces of $\mathbf{A}$ is at most n, there can be at most one eigenspace whose dimension is larger than $m/2$. Furthermore, the eigenvalue a_i corresponding to this eigenspace will necessarily be the extracted value a.

Rather than showing how we can simulate $\hat{\mathbf{y}} = \mathbf{A}\hat{\mathbf{x}}_1 + \hat{\mathbf{x}}_2$ in general, in this sketch we will only briefly argue the following special case. Namely, if all the eigenspaces of $\mathbf{A}$ have dimension smaller than or equal to $m/2$, then with high probability over the choice of the random matrix $\mathbf{R} \leftarrow_\$ \mathbb{F}_q^{n \times m}$ it holds that $\mathbf{x}_1 = \mathbf{R}\hat{\mathbf{x}}_1$ and $\mathbf{x}_2 = \mathbf{R}\hat{\mathbf{x}}_2$ are uniform and independent of $\hat{\mathbf{y}}$. Thus assume that $\hat{\mathbf{y}} = \mathbf{A}\hat{\mathbf{x}}_1 + \hat{\mathbf{x}}_2$ was not independent of $\mathbf{x}_1 = \mathbf{R}\hat{\mathbf{x}}_1$ and $\mathbf{x}_2 = \mathbf{R}\hat{\mathbf{x}}_2$. Since these three variables are linear functions of the uniformly random $\hat{\mathbf{x}}_1$ and $\hat{\mathbf{x}}_2$ there must exist a non-zero linear relation given by vectors $\mathbf{u}, \mathbf{v} \in \mathbb{F}_q^n$ and $\mathbf{w} \in \mathbb{F}_q^m$ such that $\mathbf{u}^\top \mathbf{x}_1 + \mathbf{v}^\top \mathbf{x}_2 + \mathbf{w}^\top \hat{\mathbf{y}} = 0$ for all choices of $\hat{\mathbf{x}}_1$ and $\hat{\mathbf{x}}_2$. But this means that it holds that $\mathbf{u}^\top \mathbf{R} + \mathbf{w}^\top \mathbf{A} = 0$ and $\mathbf{v}^\top \mathbf{R} + \mathbf{w}^\top = 0$. Eliminating $\mathbf{w}^\top$, this simplifies to the equation $\mathbf{u}^\top \mathbf{R} = \mathbf{v}^\top \mathbf{R}\mathbf{A}$.

We will now argue that for any such matrix $\mathbf{A} \in \mathbb{F}_q^{m \times m}$ (whose eigenspaces all have dimension $\leq m/2$) with high probability over the choice of the random matrix $\mathbf{R}$, such a relation given by $(\mathbf{u}, \mathbf{v}) \neq 0$ does not exist. We will take a union bound over all non-zero $\mathbf{u}, \mathbf{v}$ and distinguish the following cases:

- If $\mathbf{u}$ and $\mathbf{v}$ are linearly independent, then $\mathbf{u}^\top \mathbf{R}$ and $\mathbf{v}^\top \mathbf{R}$ are uniformly random and independent (over the random choice of $\mathbf{R}$). Thus the probability that $\mathbf{u}^\top \mathbf{R}$ and $\mathbf{v}^\top \mathbf{R}\mathbf{A}$ collide is $1/q^m$.
- If $\mathbf{u}$ and $\mathbf{v}$ are linearly dependent, then (say) $\mathbf{u} = \alpha\mathbf{v}$. In this case $\mathbf{u}^\top \mathbf{R} = \mathbf{v}^\top \mathbf{R}\mathbf{A}$ is equivalent to $\alpha\mathbf{v}^\top \mathbf{R} = \mathbf{v}^\top \mathbf{R}\mathbf{A}$, i.e. the uniformly random vector $\mathbf{v}^\top \mathbf{R}$ is an eigenvector of the matrix $\mathbf{A}$ with respect to the eigenvalue α. However, since all eigenspaces of $\mathbf{A}$ have dimension at most $m/2$, the probability that $\mathbf{v}^\top \mathbf{R}$ lands in one of the eigenspaces bounded by $m/q^{m/2}$.

Since there are q^{2n} possible choices for the vectors $\mathbf{u}, \mathbf{v} \in \mathbb{F}_q^n$, choosing m sufficiently large (e.g. $m > 5n$) implies that the probability that such $\mathbf{u}, \mathbf{v} \in \mathbb{F}_q^n$ exist is negligible. The full proof is provided in Section 6.

2.2 Algebraic Restriction Codes for Bounded Output Functions

We will now turn to algebraic restriction codes for arbitrary functions with bounded output length. Now let $\mathbb{F}_q$ be the finite field of size q, let $\mathcal{G}$ be the class of all functions from $\mathbb{F}_q^{2n} \to \{0,1\}^{1.5n\log(q)}$ and let $\mathcal{F}$ be the class of linear functions $\mathbb{F}_q^2 \to \mathbb{F}_q$, i.e. all functions of the form $f(x_1, x_2) = a_1 x_1 + a_2 x_2$ for some $a_1, a_2 \in \mathbb{F}_q$. Our AR code construction follows naturally from the inner product extractor. The seed s consists of a random vector $\mathbf{s} \leftarrow_\$ \mathbb{F}_q^n$, to encode $x_1, x_2 \in \mathbb{F}_q$ we choose uniformly random $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{F}_q^n$ with $\langle \mathbf{x}_1, \mathbf{s} \rangle = x_1$ and $\langle \mathbf{x}_2, \mathbf{s} \rangle = x_2$. Likewise, to decode a value $\mathbf{y}$ we compute $y = \langle \mathbf{y}, \mathbf{s} \rangle$, correctness follows immediately as above. To show that this construction restricts $\mathcal{G}$ to $\mathcal{F}$, we will again take the extractor perspective. Thus, assume that $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{F}_q^n$ are distributed uniformly random and let $g : \mathbb{F}_q^n \times \mathbb{F}_q^n \to \{0,1\}^{1.5n\log(p)}$ be an arbitrary function.

We need to argue that for any $g \in \mathcal{G}$ there exist exist $a_1, a_2 \in \mathbb{F}_q$ such that $g(\mathbf{x}_1, \mathbf{x}_2)$ can be simulated given $y = a_1\langle \mathbf{x}_1, \mathbf{s}\rangle + a_2\langle \mathbf{x}_2, \mathbf{s}\rangle$, but no further information about $\langle \mathbf{x}_1, \mathbf{s}\rangle$ and $\langle \mathbf{x}_2, \mathbf{s}\rangle$. Our analysis distinguishes two cases.

- In the first case, both $\langle \mathbf{x}_1, \mathbf{s}\rangle$ and $\langle \mathbf{x}_2, \mathbf{s}\rangle$ are statistically close to uniform given $g(\mathbf{x}_1, \mathbf{x}_2)$. In other words, it directly holds that $g(\mathbf{x}_1, \mathbf{x}_2)$ contains no information about $\langle \mathbf{x}_1, \mathbf{s}\rangle$ and $\langle \mathbf{x}_2, \mathbf{s}\rangle$. We can simulate $g(\mathbf{x}_1, \mathbf{x}_2)$ by choosing two independent $\mathbf{x}_1'$ and $\mathbf{x}_2'$ and computing $g(\mathbf{x}_1', \mathbf{x}_2')$.
- In the second case $\langle \mathbf{x}_1, \mathbf{s}\rangle$ and $\langle \mathbf{x}_2, \mathbf{s}\rangle$ are (jointly) statistically far from uniform given $g(\mathbf{x}_1, \mathbf{x}_2)$. In this case we will rely on a variant of the XOR Lemma [Vaz86] to conclude that there must exist $a_1, a_2 \in \mathbb{F}_q$ such that $a_1 x_1 + a_2 x_2$ is also far from uniform given $g(\mathbf{x}_1, \mathbf{x}_2)$. Roughly, the XOR Lemma states that if it holds for two (correlated) random variables z_1, z_2 that for all $a_1, a_2 \in \mathbb{F}_q$ (such that one of them is non-zero) that $a_1 z_1 + a_2 z_2$ are statistically close to uniform, then (z_1, z_2) must be statistically close to uniform in $\mathbb{F}_q^2$. Consequently, the existence of such $a_1, a_2 \in \mathbb{F}_q$ in our setting follows directly from the contrapositive of the XOR Lemma. But this implies that $a_1\mathbf{x}_1 + a_2\mathbf{x}_2$ must have very low min-entropy given $g(\mathbf{x}_1, \mathbf{x}_2)$. Otherwise, the leftover hash lemma would imply that $a_1 x_1 + a_2 x_2 = \langle a_1\mathbf{x}_1 + a_2\mathbf{x}_2, \mathbf{s}\rangle$ is close to uniform given $g(\mathbf{x}_1, \mathbf{x}_2)$, in contradiction to the conclusion above. But this means that $a_1\mathbf{x}_1 + a_2\mathbf{x}_2$ is essentially fully specified by $g(\mathbf{x}_1, \mathbf{x}_2)$. In other words $g(\mathbf{x}_1, \mathbf{x}_2)$ carries essentially the entire information about $a_1\mathbf{x}_1 + a_2\mathbf{x}_2$. But now recall that the bit size of $g(\mathbf{x}_1, \mathbf{x}_2)$ is $1.5n\log(q)$ bits and the bit size of $a_1\mathbf{x}_1 + a_2\mathbf{x}_2$ is $n\log(q)$ bits. Thus, there is essentially not enough *room* in $g(\mathbf{x}_1, \mathbf{x}_2)$ to carry significant further information about $\mathbf{x}_1$ or $\mathbf{x}_2$. Again relying on the leftover hash lemma, we then conclude that given $g(\mathbf{x}_1, \mathbf{x}_2)$, $\langle \mathbf{x}_1, \mathbf{s}\rangle$ and $\langle \mathbf{x}_2, \mathbf{s}\rangle$ are statistically close to uniform subject to $a_1\langle \mathbf{x}_1, \mathbf{s}\rangle + a_2\langle \mathbf{x}_2, \mathbf{s}\rangle = y$.

While this sketch captures the very high level ideas of our proof, the actual proof needs to overcome some additional technical challenges and relies on a careful partitioning argument. The proof can be found in Section 7.

2.3 From AR Codes to Efficient Oblivious Transfer

We display the usefulness of AR codes in cryptography by constructing a new oblivious transfer (OT) [Rab05, EGL82] protocol. OT is a protocol between two parties, a sender, who has a pair of messages (m_0, m_1), and a receiver who has a bit b, where at the end, the receiver learns m_b, while the sender should learn nothing. OT is a central primitive of study in the field of secure computation: Any multiparty functionality can be securely computed given a secure OT protocol [Yao86, Kil88]. In particular, statistically-sender private (SSP) [NP01, AIR01] 2-message OT has recently received a lot of attention due to its wide array of applications, such as statistical ZAPs [BFJ+20, GJJM20] and maliciously circuit-private homomorphic encryption [OPP14]. While the standard security definitions for OT are simulation-based (via *efficient simulators*), SSP OT settles for a weaker indistinguishability-based security notion for the receiver and an inefficient simulation notion for the sender. On the other hand, SSP OT can be realized in just two messages, without a setup and from standard assumptions, a regime in which no OT protocols with simulation-based security are known[6]. In this work, we obtain the first OT protocol that simultaneously satisfies the following properties:

(1) It is round-optimal (2 messages) and it does not assume a trusted setup.
(2) It satisfies the notion of *statistical* sender privacy (and computational receiver privacy). That is, a receiver who may (potentially) choose her first round message maliciously will be statistically oblivious to at least one of the two messages of the sender.
(3) It achieves optimal rate for information transfer (i.e., it is rate-1).
(4) It makes only black-box use of cryptographic primitives, in the sense that our protocol does not depend on circuit-level implementations of the underlying primitives.

[6]In fact, it can be shown that any simulator for such a protocol would need to make non-black-box use of the adversary, as it would immediately imply a two-message zero-knowledge protocol, which was shown black-box impossible in [GO94]

Prior to our work, we did not know any OT protocol that simultaneously satisfied all of the above properties from *any assumption.* The only previous construction was based on LWE (using expensive fully-homomorphic encryption techniques), which only satisfies the first three conditions, but not the last one. (See Section 3.) We obtain constructions that satisfy all the above conditions from DDH/LWE. Optimal-rate OT is an indispensable tool in relazing various MPC functionalities with sublinear communication [IP07]. As direct corollaries, we obtain two-message maliciously secure protocols for keyword search [IP07] and *symmetric* private information retrieval (PIR) protocols [KO97] with statistical server privacy and with asymptotically optimal communication complexity from DDH/LWE. Our scheme is the first that makes only *black-box* use of cryptography, which we view as an important step towards the practical applicability of these protocols.

Packed ElGamal. Before delving into the description of our scheme, we recall the *vectorized* variant of the ElGamal encryption scheme [ElG84]. Let $\mathbb{G}$ be an Abelian group of prime order p and let g be a generator of $\mathbb{G}$. In the packed ElGamal scheme, a public key pk consists of a vector $\mathbf{h} = (h_1, \ldots, h_n) \in \mathbb{G}^n$ where $h_i = g^{x_i}$ for random $x_i \leftarrow_\$ \mathbb{Z}_p$. The secret sk is the vector $\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{Z}_p^n$. To encrypt a $\mathbf{m} = (m_1, \ldots, m_n) \in \{0,1\}^n$, we choose a uniformly random $r \leftarrow_\$ \mathbb{Z}_p$ and set the ciphertext $\mathbf{c}$ to

$$\mathbf{c} = (d_0, \mathbf{d}) = (g^r, \mathbf{h}^r \cdot g^{\mathbf{m}})$$

where both exponentiations and group operations of vectors are component-wise. We call d_0 the *header* of the ciphertext and $\mathbf{d} = (d_1, \ldots, d_n)$ the payload of $\mathbf{c}$, we further call $d_1, \ldots, d_n$ the *slots.* To decrypt a ciphertext $\mathbf{c}$, we compute $\mathbf{m} = \mathsf{dlog}_g(d_0^{-\mathbf{x}} \cdot \mathbf{d})$. If we disregard the need for efficient decryption, we can encrypt arbitrary $\mathbb{Z}_p^n$ vectors rather than just binary vectors. For such *full range plaintexts* the rate of packed ElGamal, i.e. the ratio between plaintext size and ciphertext size comes down to $(1 - 1/(n+1)) \log(p)/\lambda$, assuming a group element can be described using λ bits. If $\lambda \approx \log(p)$, as is the case for *dense groups*, the rate approaches 1, for sufficiently large n. Finally, for a matrix $\mathbf{X} \in \{0,1\}^{n \times k}$, we encrypt $\mathbf{X}$ column-wise, to obtain a ciphertext-matrix $\mathbf{C}$.

Homomorphism and Ciphertext Compression. Packed ElGamal supports two types of homomorphism. It is linearly homomorphic with respect to $\mathbb{Z}_p$-linear combinations. Namely, if $\mathbf{c}$ is an encryption of a vector $\mathbf{m} \in \mathbb{Z}_p^n$ and $\mathbf{c}'$ is an encryption of a vector $\mathbf{m}' \in \mathbb{Z}_p^n$, then for any $\alpha, \beta \in \mathbb{Z}_p$ it holds that $\mathbf{c}'' = \mathbf{c}^\alpha \cdot \mathbf{c}'^\beta$ is a well-formed encryption of $\alpha\mathbf{m} + \beta\mathbf{m}'$ (again, disregarding the need for efficient decryption for large plaintexts). This routinely generalizes to arbitrary linear combinations, namely we can define a homomorphic evaluation algorithm Eval_1 which takes as input a public key pk, a ciphertext matrix $\mathbf{C}$ encrypting a matrix $\mathbf{X} \in \mathbb{Z}_p^{n \times m}$, and two vectors $\mathbf{a} \in \mathbb{Z}_p^m$ and $\mathbf{b} \in \mathbb{Z}_p^n$ and outputs an encryption of $\mathbf{Xa} + \mathbf{b}$. By re-randomizing the resulting ciphertext this can be made function private, i.e. the output ciphertext leaks nothing beyond $\mathbf{Xa} + \mathbf{b}$ about $\mathbf{a}$ and $\mathbf{b}$.

The second type of homomorphism supported by packed ElGamal is a limited type of homomorphism *across the slots.* Specifically, let $\mathbf{c} = (d_0, \mathbf{d})$ be an encryption of a message $\mathbf{m} \in \mathbb{Z}_p^n$ and let $\mathbf{M} \in \mathbb{Z}_p^{m \times n}$ be a matrix. Then there is a homomorphic evaluation algorithm Eval_2 which takes the public key pk, the ciphertext $\mathbf{c}$ and a matrix $\mathbf{M} \in \mathbb{Z}_p^{m \times n}$ and outputs a ciphertext $\mathbf{c}'$, such that $\mathbf{c}'$ encrypts the message $\mathbf{m}' = \mathbf{Mm}$ under a modified public key $\mathsf{pk}' = g^{\mathbf{Mx}}$. Furthermore, if the decrypter knows the matrix $\mathbf{M}$, it can derive the modified secret $\mathsf{sk}' = \mathbf{Mx}$ and decrypt $\mathbf{c}'$ to $\mathbf{m}'$ (given that $\mathbf{m}' \in \{0,1\}^m$).

Finally, the packed ElGamal scheme supports *ciphertext compression* for bit-encryptions [DGI+19]. There is an efficient algorithm Shrink which takes a ciphertext $\mathbf{c} = (d_0, \mathbf{d})$ and produces a compressed ciphertext $\tilde{\mathbf{c}} = (d_0, K, \mathbf{b})$, where K is a (short) key and $\mathbf{b} \in \{0,1\}^n$ is a binary vector. Consequently, compressed ciphertexts are of size $n + \mathsf{poly}(\lambda)$ bits and therefore have rate $1 - \mathsf{poly}(\lambda)/n$, which approaches 1 for a sufficiently large n (independent of the description size of group elements). Such compressed ciphertexts can then be decrypted using a special algorithm $\mathsf{ShrinkDec}$, using the same secret key sk. Compressed ciphertexts generally do not support any further homomorphic operations, so ciphertext compression is performed after all homomorphic operations.

Semi-Honest Rate-1 OT from Packed ElGamal. The packed ElGamal encryption scheme with ciphertext compression immediately gives rise to a *semi-honestly secure* OT protocol with download rate 1. Specifically, the receiver whose choice-bit is b generates a key-pair pk, sk, encrypts the matrix $b \cdot \mathbf{I}$ to a ciphertext matrix $\mathbf{C}$, and sends $ot_1 = (\mathsf{pk}, \mathbf{C})$ to the sender. The sender, whose input are two strings $\mathbf{m}_0$ and

$\mathbf{m}_1 \in \{0,1\}^n$ uses Eval_1 to homomorphically evaluate the function

$$f(\mathbf{X}) = \mathbf{X}(\mathbf{m}_1 - \mathbf{m}_0) + \mathbf{m}_0$$

on the ciphertext $\mathbf{C}$, obtaining a ciphertext $\mathbf{c}$. It then compresses the ciphertext $\mathbf{c}$ to a compressed ciphertext $\tilde{\mathbf{c}}$ and sends $ot_1 = \tilde{\mathbf{c}}$ back to the receiver who can decrypt it to a value $\mathbf{m}'$ using the $\mathsf{ShrinkDec}$ algorithm. By homomorphic correctness it holds that $b \cdot \mathbf{I} \cdot (\mathbf{m}_1 - \mathbf{m}_0) + \mathbf{m}_0 = \mathbf{m}_b$.

However, note that the sender privacy of this protocol completely breaks down against malicious receivers. Specifically, a malicious receiver is not bound to encrypting the scalar matrix $b \cdot \mathbf{I}$, but could instead encrypt an arbitrary matrix $\mathbf{A} \in \mathbb{Z}_p^{n \times n}$, thereby learning $\mathbf{A}(\mathbf{m}_1 - \mathbf{m}_0) + \mathbf{m}_0$ instead of $\mathbf{m}_b$. By e.g. choosing

$$\mathbf{A} = \begin{pmatrix} 0 & 0 \\ 0 & \mathbf{I} \end{pmatrix}$$

the receiver could learn half of the bits of $\mathbf{m}_0$ and half of the bits of $\mathbf{m}_1$, thus breaking sender privacy.

Malicious Security via AR Codes. Next we show how to make the above protocol statistically sender private against malicious receivers using AR codes. The protocol follows the same outline as above, except that the sender samples a seed $\mathbf{R}$ for an AR code and encodes its inputs

$$\hat{\mathbf{x}}_1 = \mathsf{Encode}(\mathbf{R}, \mathbf{m}_1 - \mathbf{m}_0) \text{ and } \hat{\mathbf{x}}_2 = \mathsf{Encode}(\mathbf{R}, \mathbf{m}_0).$$

Then it computes a ciphertext $\mathbf{c} = \mathsf{Eval}_1(\mathsf{pk}, \mathbf{C}, \hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2)$. If the sender were to transmit directly this ciphertext, the rate of the scheme would degrade (due to the size of the encodings) and the decryption would not be efficient, since $\mathbf{c}$ contains an encoding $\hat{\mathbf{y}} \in \mathbb{Z}_p^m$. To deal with this issue, we observe that decoding $\hat{\mathbf{y}}$ to $\mathbf{y}$ via $\mathbf{y} = \mathbf{R}\hat{\mathbf{y}}$ is exactly the type of operation supported by the homomorphic evaluation Eval_2. Thus, we let the sender further compute $\mathbf{c}' = \mathsf{Eval}_2(\mathsf{pk}, \mathbf{c}, \mathbf{R})$. By homomorphic correctness of Eval_2, it holds that $\mathbf{c}'$ is an encryption of $\mathbf{R}\hat{\mathbf{y}} = \mathbf{y} = \mathbf{m}_b \in \{0,1\}^n$ under a modified public key pk' (which depends on $\mathbf{R}$). Since $\mathbf{c}'$ encrypts a binary message, the sender can further use the ciphertext compression algorithm Shrink to shrink $\mathbf{c}'$ into a rate-1 ciphertext $\tilde{\mathbf{c}}$. The sender now sends $\mathbf{R}$ and $\tilde{\mathbf{c}}$ back to the receiver, who derives a key from sk and $\mathbf{R}$, and uses it to decrypt $\tilde{\mathbf{c}}$ via $\mathsf{ShrinkDec}$.

If we were to do things naively, the protocol would still not achieve rate-1 since we have to also attach to the OT second message a potentially large matrix $\mathbf{R}$. This can be resolved via a standard trick: By reusing the same matrix $\mathbf{R}$ in several parallel instances of the protocol, we can amortize the cost of sending the matrix $\mathbf{R}$. Note that $\mathbf{R}$ can be reused as we only need to ensure that the matrix $\mathbf{A}$ does not depend on $\mathbf{R}$. Thus, we have achieved a rate-1 protocol.

There is one subtle aspect that we need to address before declaring victory: The security of AR codes only guarantees that a malicious receiver may learn $a(\mathbf{m}_1 - \mathbf{m}_0) + \mathbf{m}_0$ for some $a \in \mathbb{Z}_p$, rather than $b(\mathbf{m}_1 - \mathbf{m}_0) + \mathbf{m}_0 = \mathbf{m}_b$ for $b \in \{0,1\}$. To address this last issue, we let the sender compute $\hat{\mathbf{x}}_1$ and $\hat{\mathbf{x}}_2$ by

$$\hat{\mathbf{x}}_1 = \mathsf{Encode}(\mathbf{R}, \mathbf{x}_1)$$
$$\hat{\mathbf{x}}_2 = \mathsf{Encode}(\mathbf{R}, \mathbf{x}_2),$$

where $\mathbf{x}_1 = \begin{pmatrix} \mathbf{m}_1 - \mathbf{m}_0 + \mathbf{r}_0 \\ \mathbf{m}_1 - \mathbf{m}_0 + \mathbf{r}_1 \end{pmatrix}$ and $\mathbf{x}_2 = \begin{pmatrix} \mathbf{m}_0 \\ \mathbf{m}_0 - \mathbf{r}_1 \end{pmatrix}$ and $\mathbf{r}_0, \mathbf{r}_1$ are uniformly random.

Consequently, instead of $a(\mathbf{m}_1 - \mathbf{m}_0) + \mathbf{m}_0$ the ciphertext $\mathbf{c}$ now encrypts

$$f(\mathbf{x}_1, \mathbf{x}_2) = a \cdot \begin{pmatrix} \mathbf{m}_1 - \mathbf{m}_0 + \mathbf{r}_0 \\ \mathbf{m}_1 - \mathbf{m}_0 + \mathbf{r}_1 \end{pmatrix} + \begin{pmatrix} \mathbf{m}_0 \\ \mathbf{m}_0 - \mathbf{r}_1 \end{pmatrix},$$

and by the security of the AR code $\mathbf{c}$ does not leak more information about $\mathbf{x}_1$ and $\mathbf{x}_2$ then $f(\mathbf{x}_1, \mathbf{x}_2)$. Now, note that if $a = 0$, then

$$f(\mathbf{x}_1, \mathbf{x}_2) = \begin{pmatrix} \mathbf{m}_0 \\ \mathbf{m}_0 - \mathbf{r}_1 \end{pmatrix},$$

where we note that $\mathbf{r}_1' = \mathbf{m}_0 - \mathbf{r}_1$ is uniformly random. On the other hand, if $a = 1$, then

$$f(\mathbf{x}_1, \mathbf{x}_2) = \begin{pmatrix} \mathbf{m}_1 + \mathbf{r}_0 \\ \mathbf{m}_1 \end{pmatrix},$$

where we note that $\mathbf{r}_0' = \mathbf{m}_1 + \mathbf{r}_0$ is uniformly random. Finally, if $a \notin \{0, 1\}$, then

$$f(\mathbf{x}_0, \mathbf{x}_1) = a \cdot \begin{pmatrix} \mathbf{m}_1 - \mathbf{m}_0 + \mathbf{r}_0 \\ \mathbf{m}_1 - \mathbf{m}_0 + \mathbf{r}_1 \end{pmatrix} + \begin{pmatrix} \mathbf{m}_0 \\ \mathbf{m}_0 - \mathbf{r}_1 \end{pmatrix} = \begin{pmatrix} a\mathbf{m}_1 + (1-a)\mathbf{m}_0 \\ a\mathbf{m}_1 + (1-a)\mathbf{m}_0 \end{pmatrix} + \begin{pmatrix} a \cdot \mathbf{r}_0 \\ (1-a) \cdot \mathbf{r}_1 \end{pmatrix},$$

which is uniformly random as the last term is uniformly random. I.e. if $a \notin \{0, 1\}$ the receiver will learn nothing about $\mathbf{m}_0$ and $\mathbf{m}_1$. Thus, we can conclude that even for a malformed public key pk and ciphertext $\mathbf{C}$ the view of the receiver can be simulated given at most one $\mathbf{m}_b$, and statistical sender privacy follows.

Back to Rate-1. Note that now the ciphertext $\mathbf{c}$ is twice as long as before, which again ruins the rate of our scheme. However, note that in order to get a correct scheme, if $a = 0$ the receiver only needs to recover the first half $\mathbf{z}_0$ of the vector $f(\mathbf{x}_1, \mathbf{x}_2) = \begin{pmatrix} \mathbf{z}_0 \\ \mathbf{z}_1 \end{pmatrix}$, whereas if $a = 1$ she needs the second part $\mathbf{z}_1$. Our final idea is to facilitate this by additionally using a rate-1 OT protocol $\mathsf{OT}' = (\mathsf{OT}_1', \mathsf{OT}_2', \mathsf{OT}_3')$ with *semi-honest security* (e.g. as given in [DGI+19]). We will further use the fact that the packed ElGamal ciphertext $\tilde{\mathbf{c}}$ can be written as $(h, \tilde{\mathbf{c}}_0, \tilde{\mathbf{c}}_1)$, where h is the ciphertext header, $\tilde{\mathbf{c}}_0$ is a rate-1 ciphertext encrypting $\mathbf{z}_0$ and $\tilde{\mathbf{c}}_1$ is a rate-1 ciphertext encrypts $\mathbf{z}_1$ (both with respect to the header h).

We modify the above protocol such that the receiver additionally includes a first message ot_1' computed using his choice bit b. Instead of sending both $\tilde{\mathbf{c}}_0$ and $\tilde{\mathbf{c}}_1$ to the receiver (which would ruin the rate), we compute the sender message ot_2' for OT' as $ot_2' \leftarrow \mathsf{OT}_2(ot_1', \tilde{\mathbf{c}}_0, \tilde{\mathbf{c}}_1)$ and send (h, ot_2') to the receiver. The receiver can now recover $\tilde{\mathbf{c}}_b$ from ot_2' and decrypt the ciphertext $(h, \tilde{\mathbf{c}}_b)$ as above. Note that now the communication rate from sender to receiver is 1. Note that we do not require any form of sender security from the rate-1 OT. Finally, note that as discussed above the the protocol can be made overall rate-1 by amortizing for the size of the receiver's message (i.e. repeating the protocol in parallel for the same receiver message but independent blocks of the sender message).

Certified vs Uncertified Groups. We conclude this overview by discussing two variants of groups where we can implement the OT as specified above. In *certified groups*, we can assume that $\mathbb{G}$ in fact implements a group of prime order p, even if maliciously chosen. In these settings, our simpler variant of AR codes suffices, since we are warranted that a malicious receiver can only obtain information of the form $\mathbf{A}\hat{\mathbf{x}}_1 + \hat{\mathbf{x}}_2$ (for an arbitrarily chosen matrix $\mathbf{A}$). In *non-certified groups*, the linearity of the group is no longer checkable by just looking at its description $\mathbb{G}$. Here we can only appeal to the fact that have a bound on the size of the output learned by the receiver, enforced by the fact that our OT achieves rate-1: The second OT message is too short to encode both $\hat{\mathbf{x}}_1$ and $\hat{\mathbf{x}}_2$. In these settings, we need the full power of bounded-output AR codes, in order to show the statistical privacy of the above protocol.

2.4 Roadmap

We discuss some related works in Section 3. The preliminaries are provided in Section 4. We will introduce algebraic restriction codes in Section 5. In Section 6 we show that canonic AR codes restrict general linear functions to simple linear functions. In Section 7 we show that canonic AR codes restrict output-bounded functions to simple linear combinations, where the main result of this section is stated in Theorem 5. In Section 8 we provide our construction of rate-1 SSP OT from DDH and we discuss novel applications in Section 9.

3 Related Work

A recent line of works [DGI+19] proposed a new approach to constructing semi-honest OT with a rate approaching 1. This framework can be instantiated from a wide range of standard assumptions, such as the

DDH, QR and LWE problems. The core idea of this approach is to construct OT from a special type of packed linearly homomorphic encryption scheme which allows compressing ciphertexts after homomorphic evaluation. Pre-evaluation ciphertexts in such packed encryption schemes typically need to encrypt a *structured plaintext* containing redundant information to guarantee correctness of homomorphic evaluation. In the context of statistical sender privacy, this presents an issue as a malicious receiver may deviate from the structure required by the protocol to (potentially) learn correlated information about m_0 and m_1.

Regarding the construction of SSP OT, all current schemes roughly follow one of three approaches sketched below.

The Two Keys Approach [NP01, AIR01, HK12, BD18]. In this construction blueprint, the receiver message ot_1 specifies two (correlated) public keys pk_0 and pk'_1 under potentially different public key encryption schemes. The sender's message ot_2 now consists of two ciphertexts $c_0 = \mathsf{Enc}(\mathsf{pk}_0, m_0)$ and $c_1 = \mathsf{Enc}'(\mathsf{pk}'_1, m_1)$. Statistical sender privacy is established by choosing the correlation between the keys pk_0 and pk'_1 in such a way that one of these keys *must be lossy*, and that this is either directly enforced by the underlying structure or checkable by the sender. Here, lossiness means that either c_0 or c_1 loses information about their respective encrypted message. In group-based constructions following this paradigm [NP01, AIR01, HK12], the sender must trust that the structure on which the encryption schemes are defined actually implements a group in order to be convinced that either pk_0 or pk'_1 is lossy. We say that the group $\mathbb{G}$ must be a *certified group*. This is problematic if the group $\mathbb{G}$ is chosen by the receiver, as the group $\mathbb{G}$ could e.g. have non-trivial subgroups which prevent lossiness.

Furthermore, note that since the sender's message ot_2 contains two ciphertexts, each of which should, from the sender's perspective *be potentially decryptable*, this approach is inherently limited to rates below $1/2$.

The Compactness Approach [BGI$^+$17]. The second approach to construct SSP OT is based on high rate OT. Specifically, assume we are starting with any two round OT protocol with a (download) rate greater than $1/2$, say for the sake of simplicity with rate close to 1. This means that the sender's message ot_2 is *shorter than the concatenation of m_0 and m_1*. But this means that, from an information theoretic perspective ot_2 must lose information about either m_0 or m_1. This lossiness can now be used to bootstrap statistical sender privacy as follows. The sender chooses two random messages r_0 and r_1 and uses them as his input to the OT. Moreover, he uses a randomness extractor to derive a key k_0 from r_0 and k_1 from r_1 respectively. Now the sender provides two one-time pad encrypted ciphertexts $c_0 = k_0 \oplus m_0$ and $c_1 = k_1 \oplus m_1$ to the receiver. A receiver with choice bit b can then recover r_b from the OT, derive the key k_b via the randomness extractor and obtain m_b by decrypting c_b.

To argue statistical sender privacy using this approach, we need to ensure that one of the keys k_0 or k_1 is uniformly random from a malicious receivers perspective. Roughly speaking, due to the discussion above the second OT message ot_2 needs to lose either half of the information in r_0 or r_1. Thus, in the worst case, the receiver could learn half of the information in each r_0 and r_1 from ot_2. Consequently, we need a randomness extractor which produces a uniformly random output as long as its input has $n/2$ bits of min-entropy. Thus, we can prove statistical sender privacy for messages of length smaller than $n/2$.

But in terms of communication efficiency, this means that we used a high rate n-bit string OT to implement a string OT of length $\leq n/2$, which means that the rate of the SSP OT we've constructed is less than $1/2$. This is true without even taking into account the addition communication cost required to transmit the ciphertexts c_0 and c_1. Thus, this approach effectively trades high rate for statistical sender privacy at the expense of falling back to a lower rate. We conclude that this approach is also fundamentally stuck at rate $1/2$.

The Non Black-Box Approach [BDGM19, GH19]. While the above discussion seems to imply that there might be an inherent barrier in achieving SSP OT with rate $> 1/2$, there is in fact a way to convert any SSP OT protocol into a rate-1 SSP OT protocol using sufficiently powerful tools. Specifically, using a rate-1 fully-homomorphic encryption (FHE) scheme [BDGM19, GH19], the receiver can *delegate* the decryption of ot_2 to the sender. In more detail, assume that $\mathsf{OT}_3(st, ot_2)$ is the decryption operation which is performed by the receiver at the end of the SSP OT protocol. By providing an FHE encryption $FHE.\mathsf{Enc}(st)$ of the OT receiver state st along with the first message ot_1, the receiver enables the sender to perform $\mathsf{OT}_3(st, ot_2)$

homomorphically, resulting in an FHE encryption c of the receivers output m_b. Now the receiver merely has to decrypt c to recover m_b. In terms of rate, note that the OT sender message now merely consists of c, which is rate-1 as the FHE scheme is rate-1. Further note that this transformation does not harm SSP security, as from the sender's view the critical part of the protocol is over once ot_2 has been computed. I.e. for the sender performing the homomorphic decryption is merely a *post-processing operation*. On the downside, this transformation uses quite heavy tools. In particular, this transformation needs to make *non black-box* use of the underlying SSP OT protocol by performing the OT_3 operation homomorphically.

In summary, to the best of our knowledge, all previous approaches to construct SSP OT are either fundamentally stuck at rate 1/2 or make non black-box usage of the underlying cryptographic machinery, making it prohibitively expensive to run such a protocol in practice.

Finally, we mention that if one wishes to settle on a computational instead of statistical privacy for the sender, it is possible to build rate-1 OT using existing techniques by relying on super-polynomial hardness assumptions. The idea is that the parties will first engage in a (low-rate) OT protocol OT_1, so that the receiver will learn one of the two random PRG seeds (s_0, s_1) sampled by the sender. In parallel, the sender prepares two ciphertexts $(\mathsf{ct}_0 := \mathsf{PRG}(s_0) \oplus m_0, \mathsf{ct}_1 := \mathsf{PRG}(s_1) \oplus m_1)$ for his two input messages (m_0, m_1), and communicates one of them to the receiver using a semi-honest rate-1 OT protocol. Even given both $(\mathsf{ct}_0, \mathsf{ct}_1)$ the receiver cannot recover both m_0 and m_1, because OT_1 will guarantee at least one of the seeds remains computationally hidden to the receiver. The above protocol is rate-1 because the added communication of obliviously transferring (s_0, s_1) is independent of the size of m_0. The main drawback of this above protocol is that, since we do not rely on a trusted setup, we cannot extract the choice bit in polynomial time from the receiver, and hence we will have to rely on complexity leveraging to establish sender security. In particular, the best we can guarantee is that a malicious computationally-bounded receiver cannot compute both messages of the sender. This notion will fall short in replacing rate-1 SSP OT in the aforementioned applications.

4 Preliminaries

We will denote finite fields of unspecified size by $\mathbb{F}$, and for any prime-power q we will denote the finite field of size q by $\mathbb{F}_q$. We will use $u_{\mathbb{F}}$ to denote a uniform and independent random variable over $\mathbb{F}$, and likewise $\mathbf{u}_{\mathbb{F}^t}$ to denote a uniform and independent random variable over $\mathbb{F}^t$.

$\mathbb{Z}$ are the integers and $\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z}$ are the integers modulo q. Vectors are small, bold letter (i.e. $\mathbf{a}, \mathbf{b}$) while matrices are big, bold letters (i.e. $\mathbf{A}, \mathbf{B}$). For sets of functions we use big, italic letters (i.e. $\mathcal{F}, \mathcal{G}$). We also use $[n]$ instead of $\{1, \ldots, n\}$.

For a cyclic group we use $\mathbb{G}$ and usually call its generator g. As a shorthand for the matrix of group elements $(g^{M_{i,j}})_{i,j \in [n]}$ we write $g^{\mathbf{M}}$ where $\mathbf{M}$ is a matrix is from $\mathbb{Z}^{n^n}$ and similarly for vectors and rectangular matrices. This allows for notations such as $(g^{\mathbf{M}})^{\mathbf{v}} = g^{\mathbf{Mv}}$ where $\mathbf{M}$ is a $n \times n$ matrix of group elements and $\mathbf{v}$ is a n vector of group elements.

We use $\mathsf{span}(\mathbf{M})$ to indicate the column span of matrix $\mathbf{M}$ and $\mathsf{LKer}(\mathbf{M})$ its kernel.

Definition 1 (Computational Indistinguishability). *Two random variables B and C are computationally indistinguishable if for every polynomial adversary $\mathcal{A}$*

$$|\mathsf{Pr}_{b \sim B}[\mathcal{A}(b) = 1] - \mathsf{Pr}_{c \sim C}[\mathcal{A}(c) = 1]|$$

is negligible in the security parameter

Sometimes we denote this with $A \approx B$.

4.1 Statistical Measures

We introduce some standard concepts for statistical measures.

Definition 2 (Statistical Distance). *We define the statistical distance between two discrete random variables A, B to be*

$$\varDelta(A\,;B) = \frac{1}{2}\sum_{v} |\mathsf{Pr}[A = v] - \mathsf{Pr}[B = v]|$$

We use $\varDelta(A\,;B|C)$ as a shorthand for $\varDelta((A,C)\,;(B,C))$. Sometimes we write $A \approx_\epsilon B$ instead of $\varDelta(A\,;B) \leq \epsilon$.

We call two random variables statistically indistinguishable if their statistical distance is negligible in some the security parameter. We denote this as $\approx$ or $\approx_s$. Since $\approx$ by itself is ambiguous we will make it clear from the context.

Definition 3 (Min-Entropy). *We define the min-entropy of a random variable A to be*

$$\mathbf{H}_\infty(A) = -\log(\max_v \mathsf{Pr}[A = v])$$

Definition 4 (Average Conditional Min-Entropy). *We define the average conditional min-entropy of random variable A given the random variable B*

$$\widetilde{\mathbf{H}}_\infty(A|B) = -\log\left(\mathbb{E}_{b\sim B}\left[\max_a \mathsf{Pr}[A = a|B = b]\right]\right)$$

We will make use of the following simple Lemma.

Lemma 1 (See e.g. [vLW01]). *Let $\mathbb{F}$ be a finite field and $m > n$ be integers. A uniformly random matrix $\mathbf{R} \leftarrow_\$ \mathbb{F}^{n\times m}$ has full rank, except with probability $2^{-(m-n)}$.*

We will use the following variant of the leftover hash lemma.

Lemma 2. *Let $\mathbf{r}$ be uniform in $\mathbb{F}^n$, and $\mathbf{l}$ be a random variable in $\mathbb{F}^n$, $Z \in \mathcal{Z}$ such that $(\mathbf{l}, Z)$ is independent of $\mathbf{r}$. If*

$$\widetilde{\mathbf{H}}_\infty(\mathbf{l}|Z) \geq \log q + 2\log\left(\frac{1}{\varepsilon}\right) ,$$

then

$$\varDelta((\mathbf{r}, Z, \langle \mathbf{l}, \mathbf{r}\rangle)\ ;\ (\mathbf{r}, Z, u_\mathbb{F})) \leq \varepsilon ,$$

where $\langle\cdot,\cdot\rangle$ is the inner product over $\mathbb{F}$.

We will need the following simple lemma from [AO20]. Variants of this lemma have been used in the past to prove the security of various non-malleable code constructions (such as [DKO13, ADL14]).

Lemma 3. *Let S be some random variable distributed over a set $\mathcal{S}$, and let $\mathcal{S}_1, \ldots, \mathcal{S}_j$ be a partition of $\mathcal{S}$. Let $\phi : \mathcal{S} \to \mathcal{T}$ be some function, and let $D_1, \ldots, D_j$ be some random variables over the set $\mathcal{T}$. Assume that for all $1 \leq i \leq j$,*

$$\varDelta\left(\phi(S)|_{S\in\mathcal{S}_i}\ ;\ D_i\right) \leq \varepsilon_i .$$

Then

$$\varDelta\left(\phi(S)\ ;\ D\right) \leq \sum \varepsilon_i \mathsf{Pr}[S \in S_i] ,$$

for some random variable $D \in \mathcal{T}$ such that for all d $\mathsf{Pr}[D = d] = \sum_i \mathsf{Pr}[S \in \mathcal{S}_i]\cdot\mathsf{Pr}[D_i = d]$. In particular, if $\varepsilon_i \leq \varepsilon$ for $i = 1, \ldots, j-1$, and $\mathsf{Pr}[S \in \mathcal{S}_j] \leq \delta$, then

$$\varDelta\left(\phi(S)|_{S\in\mathcal{S}_i}\ ;\ D_i\right) \leq \varepsilon\sum_{i=1}^{j-1}\mathsf{Pr}[S \in \mathcal{S}_i] + \mathsf{Pr}[S \in \mathcal{S}_j] \leq \varepsilon + \delta\ ,.$$

The following is a fundamental property of statistical distance.

Lemma 4. *For any, possibly random, function α, if $\varDelta(A\ ;\ B) \leq \varepsilon$, then $\varDelta(\alpha(A)\ ;\ \alpha(B)) \leq \varepsilon$.*

We will need the following lemma.

Lemma 5. *[ADL14, Claim 4] Let X_1, X_2, Y_1, Y_2 be random variables such that $(X_1, X_2) \approx_\epsilon (Y_1, Y_2)$. Then, for any non-empty set $\mathcal{A}$, we have:*

$$\Delta(X_2|_{X_1 \in \mathcal{A}}; Y_2|_{Y_1 \in \mathcal{A}}) \leq \frac{2\epsilon}{\mathsf{Pr}[X_1 \in \mathcal{A}]} .$$

5 Algebraic Restriction Codes

In this section, we will define our main technical tool: Algebraic Restriction Codes. An algebraic restriction code allows encoding a linear function so that any (suitably bounded) *malicious evaluation algorithm* cannot exfiltrate information that could not have been obtained via a valid evaluation of the function. We will use algebraic restriction codes as a powerful interface to achieve circuit privacy without sacrificing other crucial properties such as high rate. Algebraic restriction codes can be seen as a specific type of secret sharing which allows for certain homomorphic operations while inhibiting others.

In particular, algebraic restriction codes will become useful in striking a balance between seemingly conflicting goals: Relying on additional structure to achieve advanced functionality while not making this additional structure a potential avenue to attack function privacy. Generally, we allow AR-codes to be seeded, i.e. all operations take as additional input a seed s. We now will define algebraic restriction codes as follows.

Definition 5. *An algebraic restriction code consists of three algorithms* Encode*,* Eval *and* Decode *with the following syntax.*

- $\mathsf{Encode}(s, x)$*: Takes as input a seed s, an input x and outputs an encoding c*
- $\mathsf{Eval}(c, f)$*: Takes as input an encoding c, a function $f \in \mathcal{F}$ and outputs an encoding d*
- $\mathsf{Decode}(s, d)$*: Takes as input a seed s, an encoding d and outputs a value y*

In terms of correctness, we require that for all seeds s, all inputs x and all functions $f \in \mathcal{F}$ that

$$\mathsf{Decode}(s, \mathsf{Eval}(\mathsf{Encode}(s, x), f)) = f(x).$$

In terms of security we require that AR codes *restrict* a potentially larger class $\mathcal{G}$ of functions to $\mathcal{F}$. Specifically, we require that for any malicious evaluation function $g \in \mathcal{G}$ that evaluating g on an encoding of an input x corresponds to an honest evaluation of a function $f \in \mathcal{F}$ on x. We formalize this via a simulation-based security notion.

Definition 6 (Restriction Security). *We say that a code* AR *is $\mathcal{G}$-$\mathcal{F}$ restriction secure, if there exists a (randomized) extractor $\mathcal{E}$, which takes as input a function $g \in \mathcal{G}$ and outputs a function $f \in \mathcal{F}$ and auxiliary information* aux*, and a simulator $\mathcal{S}$ such that for every x and every function $g \in \mathcal{G}$ it holds that*

$$(s, g(\mathsf{Encode}(s, x)), \mathsf{aux}) \approx (s, \mathcal{S}(s, \mathsf{aux}, f(x)), \mathsf{aux}),$$

where s is a uniformly random seed and $(f, \mathsf{aux}) \leftarrow \mathcal{E}(g)$. Here, $\approx$ is either computational or statistical indistinguishability.

A crucial aspect of algebraic restriction codes will be the complexity of both evaluation and decoding. Specifically, we will be interested in algebraic restriction codes for which both Eval and Decode are linear functions.

5.1 Concatenating AR Codes

Concatenation is a powerful concept in coding theory, allowing to combine properties of different codes. We will now briefly show that concatenating AR codes has the expected effect: If AR_1 restricts a class $\mathcal{H}$ to a class $\mathcal{G}'$ and AR_2 restricts a class $\mathcal{G} \supseteq \mathcal{G}'$ to another class $\mathcal{F}$, then the code AR_3 obtained by first encoding with AR_2 and then with AR_1 restricts $\mathcal{H}$ to $\mathcal{F}$.

Lemma 6. *Let AR_1 be an AR code which restricts a class $\mathcal{H}$ to a class $\mathcal{G}$. Let further AR_2 be an AR code which restricts the class $\mathcal{G}$ to a class $\mathcal{F}$. Let AR_3 be the AR code obtained by first encoding with AR_2 and then with AR_1, i.e. $\mathsf{AR}_3.\mathsf{Encode}_{s_1,s_2}(x) = \mathsf{AR}_1.\mathsf{Encode}_{s_1}(\mathsf{AR}_2.\mathsf{Encode}_{s_2}(m))$. Then the code AR_3 restricts $\mathcal{H}$ to $\mathcal{F}$.*

Proof. Let $\mathcal{S}_1$ be the simulator for AR_1 and $\mathcal{S}_2$ be the simulator for AR_2. We define the extractor in the canonic way via $\mathcal{E}_3$ via $\mathcal{E}_3(h) = (f, (\mathsf{aux}_1, \mathsf{aux}_2))$ where $(f, \mathsf{aux}_2) = \mathcal{E}_2(g)$ and $(g, \mathsf{aux}_1) = \mathcal{E}(h)$. Furthermore, we define the simulator $\mathcal{S}_3$ via $\mathcal{S}_3((\mathsf{aux}_1, \mathsf{aux}_2), y) = \mathcal{S}_1(\mathsf{aux}_1, \mathcal{S}_2(\mathsf{aux}_2, y))$. Let $h \in \mathcal{H}$ be a tampering function. We get that

$$\begin{aligned}(h(\mathsf{AR}_1.\mathsf{Encode}_{s_1}(\mathsf{AR}_2.\mathsf{Encode}_{s_2}(x))), \mathsf{aux}_1, \mathsf{aux}_2) &\approx (\mathcal{S}_1(\mathsf{aux}_1, g(\mathsf{AR}_2.\mathsf{Encode}_{s_2}(x))), \mathsf{aux}_1, \mathsf{aux}_2)\\ &\approx (\mathcal{S}_1(\mathsf{aux}_1, \mathcal{S}_2(\mathsf{aux}_2, f(x))), \mathsf{aux}_1, \mathsf{aux}_2)\\ &\approx (\mathcal{S}_3((\mathsf{aux}_1, \mathsf{aux}_2), y), \mathsf{aux}_1, \mathsf{aux}_2).\end{aligned}$$

While Lemma 6 provides a general concatenation theorem for AR codes, in our applications we will rely on a slight variant for specific function classes where AR_1 is a $\mathcal{H} - \mathcal{G}$ AR code and AR_2 is a $\mathcal{G}' - \mathcal{F}$ AR code for which the classes $\mathcal{G}$ and $\mathcal{G}'$ are not identical, but rather $\mathcal{G}'$ is a subclass of $\mathcal{G}$. In the following, we will identify the extension field $\mathbb{F}_{q^k}$ with $\mathbb{F}_q^k$ as a vector space.

Lemma 7. *Let $\mathbb{F}_q$ be a finite field and let $\mathbb{F}_{q^k}$ be its extension field of degree k. Let $\mathcal{G}$ be the class of functions $\mathbb{F}_{q^k} \times \mathbb{F}_{q^k} \to \mathbb{F}_{q^k}$ of the form $(x, y) \mapsto ax + by$ (for $a, b \in \mathbb{F}_{q^k}$), and let $\mathcal{H}$ be a class of functions containing $\mathcal{G}$. Let $\mathcal{G}'$ be the class of functions $\mathbb{F}_q^k \times \mathbb{F}_q^k \to \mathbb{F}_q^k$ of the form $(\mathbf{x}, \mathbf{y}) \mapsto \mathbf{A}\mathbf{x} + \mathbf{y}$ (for $\mathbf{A} in \mathbb{F}_q^{k \times k}$). Finally, let $\mathcal{F}$ be the class of functions $\mathbb{F}_q^n \times \mathbb{F}_q^n \to \mathbb{F}_q^n$ which are either of the form $(\mathbf{x}, \mathbf{y}) \mapsto \alpha \cdot \mathbf{x} + \mathbf{y}$ (for $\alpha \in \mathbb{F}_q$) or $(\mathbf{x}, \mathbf{y}) \mapsto \mathbf{x}$. If AR_1 is a $\mathcal{H} - \mathcal{G}$ AR code and AR_2 is a $\mathcal{G}' - \mathcal{F}$ AR code, then the concatenation of AR_1 and AR_2 is a $\mathcal{H} - \mathcal{F}$ AR code AR_3.*

Proof. Let $\mathcal{E}_1$ and $\mathcal{S}_1$ be the extractor and simulator for AR_1, and let $\mathcal{E}_2$ and $\mathcal{S}_2$ be the extractor and simulator for AR_2. We start by constructing the extractor $\mathcal{E}_3$ for AR_3. On input $h \in \mathcal{H}$, $\mathcal{E}_3$ proceeds as follows:

– Compute $(g, \mathsf{aux}_1) \leftarrow \mathcal{E}_1(h)$, and parse g as a function $(\mathbf{x}, \mathbf{y}) \mapsto ax + by$ for $a, b \in \mathbb{F}_{q^k}$.
– If $b = 0$, set f to be the function $(\mathbf{x}, \mathbf{y}) \mapsto \mathbf{x}$ and set $\mathsf{aux}_2 = \emptyset$.
– Otherwise:
  - Let $\mathbf{A}, \mathbf{B} \in \mathbb{F}_q^{k \times k}$ be the multiplication matrices corresponding to $a, b \in \mathbb{F}_{q^k}$ (Notice that $\mathbf{B}$ is invertible as $b \neq 0$) and set g' to be the function $(\mathbf{x}, \mathbf{y}) \mapsto \mathbf{B}^{-1}\mathbf{A}\mathbf{x} + \mathbf{y}$.
  - Compute $(f, \mathsf{aux}_2) \leftarrow \mathcal{E}_2(g')$
– Set $\mathsf{aux}_3 = (\mathsf{aux}_1, \mathsf{aux}_2)$
– Output (f, aux_3).

Now the simulator $\mathcal{S}_3$ is given as follows. On input $(s = (s_1, s_2), \mathsf{aux}_3, z)$, $\mathcal{S}_3$ proceeds as follows:

– If $\mathsf{aux}_2 = \emptyset$, set $z' \leftarrow \mathsf{AR}_2.\mathsf{Encode}_{s_2}(z, 0)$. Otherwise, compute $z' \leftarrow b \cdot \mathcal{S}_2(s_2, \mathsf{aux}_2, z)$.
– Compute and output $z'' \leftarrow \mathcal{S}_1(s_1, \mathsf{aux}_1, z')$.

Now fix a function $h \in \mathcal{H}$ and let $(g, \mathsf{aux}_1) \leftarrow \mathcal{E}_1(h)$, where we parse g as a function $(\mathbf{x}, \mathbf{y}) \mapsto ax + by$ for $a, b \in \mathbb{F}_{q^k}$. We will distinguish two cases, $b = 0$ and $b \neq 0$.

1. In the first case, conditioned on $b = 0$ it holds that

$$\begin{aligned}&((s_1, s_2), h(\mathsf{AR}_1.\mathsf{Encode}_{s_1}(\mathsf{AR}_2.\mathsf{Encode}_{s_2}(\mathbf{x}, \mathbf{y}))), \mathsf{aux}_1, \mathsf{aux}_2)\\ &\approx ((s_1, s_2), \mathcal{S}_1(s_1, \mathsf{aux}_1, g(\mathsf{AR}_2.\mathsf{Encode}_{s_2}(\mathbf{x}, \mathbf{y}))), \mathsf{aux}_1, \mathsf{aux}_2)\\ &\equiv ((s_1, s_2), \mathcal{S}_1(s_1, \mathsf{aux}_1, g(\mathsf{AR}_2.\mathsf{Encode}_{s_2}(\mathbf{x}, 0))), \mathsf{aux}_1, \mathsf{aux}_2)\\ &\approx ((s_1, s_2), \mathcal{S}_3((s_1, s_2), (\mathsf{aux}_1, \mathsf{aux}_2), f(\mathbf{x}, \mathbf{y})), \mathsf{aux}_1, \mathsf{aux}_2).\end{aligned}$$

2. In the second case, conditioned on $b \neq 0$ it holds that

$$\begin{aligned}
&((s_1, s_2), h(\mathsf{AR}_1.\mathsf{Encode}_{s_1}(\mathsf{AR}_2.\mathsf{Encode}_{s_2}(\mathbf{x}, \mathbf{y}))), \mathsf{aux}_1, \mathsf{aux}_2) \\
&\qquad \approx ((s_1, s_2), \mathcal{S}_1(s_1, \mathsf{aux}_1, g(\mathsf{AR}_2.\mathsf{Encode}_{s_2}(\mathbf{x}, \mathbf{y}))), \mathsf{aux}_1, \mathsf{aux}_2) \\
&\qquad \equiv ((s_1, s_2), \mathcal{S}_1(s_1, \mathsf{aux}_1, b \cdot g'(\mathsf{AR}_2.\mathsf{Encode}_{s_2}(\mathbf{x}, \mathbf{y}))), \mathsf{aux}_1, \mathsf{aux}_2) \\
&\qquad \approx ((s_1, s_2), \mathcal{S}_1(s_1, \mathsf{aux}_1, b \cdot \mathcal{S}_2(s_2, \mathsf{aux}_2, f(\mathbf{x}, \mathbf{y}))), \mathsf{aux}_1, \mathsf{aux}_2) \\
&\qquad \approx ((s_1, s_2), \mathcal{S}_3((s_1, s_2), (\mathsf{aux}_1, \mathsf{aux}_2), f(\mathbf{x}, \mathbf{y})), \mathsf{aux}_1, \mathsf{aux}_2).
\end{aligned}$$

Overall, we conclude that

$$((s_1, s_2), h(\mathsf{AR}_1.\mathsf{Encode}_{s_1}(\mathsf{AR}_2.\mathsf{Encode}_{s_2}(\mathbf{x}, \mathbf{y}))), \mathsf{aux}_1, \mathsf{aux}_2) \approx ((s_1, s_2), \mathcal{S}_3((s_1, s_2), (\mathsf{aux}_1, \mathsf{aux}_2), f(\mathbf{x}, \mathbf{y})), \mathsf{aux}_1, \mathsf{aux}_2),$$

which concludes the proof.

6 From Arbitrary Linear to Simple Linear Functions

In this section, we will show a simple construction of AR codes which constrain an adversary from arbitrary linear functions to *simple* linear functions. Specifically, we consider the following two classes of functions:

- The class $\mathcal{F}$ consists of all functions $f : \mathbb{F}_q^n \times \mathbb{F}_q^n \to \mathbb{F}_q^n$ of the form $f(\mathbf{x}, \mathbf{y}) = a\mathbf{x} + \mathbf{y}$, where $a \in \mathbb{F}_q$
- The class $\mathcal{G}$ consists of all functions $g : \mathbb{F}_q^m \times \mathbb{F}_q^m \to \mathbb{F}_q^m$ of the form $g(\mathbf{x}, \mathbf{y}) = \mathbf{A}\mathbf{x} + \mathbf{y}$, where $\mathbf{A} \in \mathbb{F}_q^{m \times m}$.

Note that the functions in the class the class $\mathcal{F}$ have two degrees of freedom, whereas the functions in class $\mathcal{G}$ have $2n^2$ degrees of freedom.

Let $s = \mathbf{R} \leftarrow_\$ \mathbb{F}_q^{n \times m}$ be a uniformly random matrix. The AR code AR_1 is given as follows.

```
Encode(s, x_1, x_2):
    – Choose $\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2 \leftarrow_\$ \mathbb{F}_q^m$ uniformly at random under the restriction that $\mathbf{R}\hat{\mathbf{x}}_1 = \mathbf{x}_1$ and $\mathbf{R}\hat{\mathbf{x}}_2 = \mathbf{x}_2$.
    – Output $c \leftarrow (\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2)$
Eval(c, a_1, a_2):
    – Parse $c = (\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2)$
    – Compute and output $\hat{\mathbf{y}} \leftarrow \hat{\mathbf{x}}_1 a_1 + \hat{\mathbf{x}}_2 a_2$
Decode(s, ŷ):
    – Compute and output $\mathbf{y} \leftarrow \mathbf{R}\hat{\mathbf{y}}$.
```

The technical core of this section is Lemma 8.

Lemma 8. *Let $q > 0$ be a modulus and $n > 0$. Let $\mathbf{A} \in \mathbb{F}_q^{m \times m}$ be a square matrix. Let $a \in \mathbb{F}_q$ be the eigenvalue of $\mathbf{A}$ for which the dimension of the corresponding eigenspace V_a is maximal. Let $\mathbf{x}_1, \mathbf{x}_2, \mathbf{u} \leftarrow_\$ \mathbb{F}_q^m$ be chosen uniformly at random. Let further $\mathbf{R} \leftarrow_\$ \mathbb{F}_q^{n \times m}$ be chosen uniformly at random. Given that $m \geq 2n + 2 + 2t$ it holds that*

$$(\mathbf{R}, \mathbf{R}\mathbf{x}_1, \mathbf{R}\mathbf{x}_2, \mathbf{A}\mathbf{x}_1 + \mathbf{x}_2) \equiv (\mathbf{R}, \mathbf{R}\mathbf{x}_1, \mathbf{R}\mathbf{x}_2, a\mathbf{x}_1 + \mathbf{x}_2 + (\mathbf{A} - a \cdot \mathbf{I})\mathbf{u}), \tag{1}$$

except with probability $2q^{-t}$ over the choice of $\mathbf{R}$.

Using Lemma 8, we will establish the main result of this section, Theorem 4.

Theorem 4. *Let $\mathcal{F}$, $\mathcal{G}$ be the two classes defined above. The AR code AR_1 restricts $\mathcal{G}$ to $\mathcal{F}$.*

Proof (of Theorem 4). Let $g(\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2) = \mathbf{A}\hat{\mathbf{x}}_1 + \hat{\mathbf{x}}_2$, and let $a \in \mathbb{F}_q$ be the eigenvalue of $\mathbf{A}$ for which the corresponding eigenspace has the largest dimension, if no non-zero eigenvalue exists set $a = 0$. By Lemma 1 the matrix $\mathbf{R}$ has full rank, except with negligible probability $2^{-(m-n)}$. By Lemma 8, for uniformly random $\hat{\mathbf{x}}_1$ and $\hat{\mathbf{x}}_2$ it holds that

$$(\mathbf{R}, \mathbf{R}\hat{\mathbf{x}}_1, \mathbf{R}\hat{\mathbf{x}}_2, \mathbf{A}\hat{\mathbf{x}}_1 + \hat{\mathbf{x}}_2) \equiv (\mathbf{R}, \mathbf{R}\hat{\mathbf{x}}_1, \mathbf{R}\hat{\mathbf{x}}_2, a\hat{\mathbf{x}}_1 + \hat{\mathbf{x}}_2 + (\mathbf{A} - a \cdot \mathbf{I})\mathbf{u}), \tag{2}$$

except with negligible probability over the choice of $\mathbf{R}$. Thus fix a $\mathbf{R}$ which has both full rank and for which (2) holds, and fix two vectors $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{F}_q^n$. Since $\mathbf{R}$ has full rank, we can condition on $\mathbf{R}\hat{\mathbf{x}}_1 = \mathbf{x}_1$ and $\mathbf{R}\hat{\mathbf{x}}_2 = \mathbf{x}_2$ and obtain that

$$\mathbf{A}\hat{\mathbf{x}}_1 + \hat{\mathbf{x}}_2 \equiv a\hat{\mathbf{x}}_1 + \hat{\mathbf{x}}_2 + (\mathbf{A} - a \cdot \mathbf{I})\mathbf{u}. \tag{3}$$

This implies that for all but a negligible fraction of the $\mathbf{R}$ we can simulate $g(\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2)$ from $\mathbf{y} = a\mathbf{x}_1 + \mathbf{x}_2$ by choosing a uniformly random random $\hat{\mathbf{y}}$ with $\mathbf{R}\hat{\mathbf{y}} = \mathbf{y}$, and a uniformly random $\mathbf{u} \in \mathbb{F}_q^m$ and outputting $\mathbf{z} = \hat{\mathbf{y}} + (\mathbf{A} - a \cdot \mathbf{I})\mathbf{u}$. By (3) it holds that $\mathbf{z}$ and $g(\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2)$ are identically distributed.

Proof (Proof of Lemma 8). First note that leaving out $\mathbf{R}$, the lefthandside of (1) can be written as $\mathbf{M}_0 \cdot (\mathbf{x}_1, \mathbf{x}_2)^\top$ where

$$\mathbf{M}_0 = \begin{pmatrix} \mathbf{R} & \\ & \mathbf{R} \\ \mathbf{A} & \mathbf{I} \end{pmatrix},$$

whereas the righthandside of (1) (again leaving out $\mathbf{R}$) can be written as $\mathbf{M}_1 \cdot (\mathbf{x}_1, \mathbf{x}_2, \mathbf{u})^\top$ where

$$\mathbf{M}_1 = \begin{pmatrix} \mathbf{R} & & \\ & \mathbf{R} & \\ a\mathbf{I} & \mathbf{I} & \mathbf{A} - a\mathbf{I} \end{pmatrix}.$$

Consequently, since $\mathbf{x}_1, \mathbf{x}_2$ and $\mathbf{u}$ are chosen uniformly random from $\mathbb{F}_q^m$, it holds that the two distributions on the lefthand side and the righthand side are identically distributed, if and only if the columns of $\mathbf{M}_0$ and $\mathbf{M}_1$ span the same space. First observe that $\mathsf{span}(\mathbf{M}_0) \subseteq \mathsf{span}(\mathbf{M}_1)$, as $\mathbf{M}_0 \cdot (\mathbf{x}_1, \mathbf{x}_2)^\top = \mathbf{M}_1(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_1)^\top$.

To show the other inclusion, note that $\mathsf{span}(\mathbf{M}_1) \subseteq \mathsf{span}(\mathbf{M}_0)$, if and only if $\mathsf{LKer}(\mathbf{M}_0) \subseteq \mathsf{LKer}(\mathbf{M}_1)$. Therefore, let $(\mathbf{v}_1, \mathbf{v}_2, \mathbf{w})$ be a vector in $\mathsf{LKer}(\mathbf{M}_0)$, i.e. it holds that $\mathbf{v}_1\mathbf{R} + \mathbf{w}\mathbf{A} = 0$ and $\mathbf{v}_2\mathbf{R} + \mathbf{w} = 0$. This immediately implies that $\mathbf{v}_1\mathbf{R} = \mathbf{v}_2\mathbf{R}\mathbf{A}$. We will show that this implies that $\mathbf{v}_2\mathbf{R}$ is an eigenvector of $\mathbf{A}$. As $\mathbf{R}$ is chosen uniformly from $\mathbb{F}_q^{n \times m}$, it holds by Theorem 4 that $\mathbf{R}$ has full rank, except with negligible probability $2^{-(m-n)}$. Now recall that a is the eigenvalue of $\mathbf{A}$ with the eigenspace of highest dimension and recall that $\mathbf{v}_2\mathbf{R}\mathbf{A} = \mathbf{v}_1\mathbf{R}$. We will show that this implies that $\mathbf{v}_1\mathbf{R}\mathbf{A} = a \cdot \mathbf{v}_2\mathbf{R}$, except with negligible probability over the choice of $\mathbf{R}$. That is, we will show that

$$\mathsf{Pr}_{\mathbf{R}}[\exists \mathbf{v}_1, \mathbf{v}_2 \neq 0 \text{ s.t. } \mathbf{v}_2\mathbf{R}\mathbf{A} = \mathbf{v}_1\mathbf{R} \text{ and } \mathbf{v}_2\mathbf{R}\mathbf{A} \neq a\mathbf{v}_2\mathbf{R}] \leq \mathsf{negl}. \tag{4}$$

In other words, it holds for all $\mathbf{v}_1, \mathbf{v}_2 \neq 0$ that $\mathbf{v}_2\mathbf{R}\mathbf{A} = \mathbf{v}_1\mathbf{R}$ implies $\mathbf{v}_2\mathbf{R}\mathbf{A} \neq a\mathbf{v}_2\mathbf{R}$, except with negligible probability over the choice of $\mathbf{R}$. From this is follows immediately that $\mathsf{LKer}(\mathbf{M}_0) \subseteq \mathsf{LKer}(\mathbf{M}_1)$, as $\mathbf{w} = -\mathbf{v}_2\mathbf{R}$ and therefore $\mathbf{w}(\mathbf{A} - a\mathbf{I}) = -a\mathbf{v}_2\mathbf{R} + a\mathbf{v}_2\mathbf{R} = 0$. We will establish (4) via a union-bound over the $\mathbf{v}_1, \mathbf{v}_2$, and towards this goal we will distinguish two cases.

1. In the first case, $\mathbf{v}_1$ and $\mathbf{v}_2$ are linearly dependent, i.e. there exists an $\alpha \in \mathbb{F}_q$ such that $\mathbf{v}_1 = \alpha\mathbf{v}_2$. If $\alpha = a$ then the probability of the event is 0. Thus consider $\alpha \neq a$, and let V_α be the eigenspace of $\mathbf{A}$ corresponding to the eigenvalue α, where $\mathsf{V}_\alpha = \{0\}$ if α is not an eigenvalue of $\mathbf{A}$. Observe that it must hold that the dimension of V_α is at most $m/2$, as otherwise α would be the eigenvalue with the eigenspace of the largest dimension and therefore $\alpha = a$. Consequently, it holds that

$$\begin{aligned} \mathsf{Pr}[\mathbf{v}_1\mathbf{R} = \mathbf{v}_2\mathbf{R}\mathbf{A}] &= \mathsf{Pr}[\alpha\mathbf{v}_2\mathbf{R} = \mathbf{v}_2\mathbf{R}\mathbf{A}] \\ &= \mathsf{Pr}[\mathbf{v}_2\mathbf{R} \in \mathsf{V}_\alpha] \leq q^{m/2}, \end{aligned}$$

as $\mathbf{v}_2\mathbf{R}$ is distributed uniformly random over $\mathbb{F}_q^m$ and the dimension of V_α is at most $m/2$.
We further note that there are at most q^n choices for $\mathbf{v}_2$ and q choices for α, thus in this case there are are q^{n+1} possible choices for the pair $(\mathbf{v}_1, \mathbf{v}_2)$.

2. In the second case, $\mathbf{v}_1$ and $\mathbf{v}_2$ are linearly independent. In this case $\mathbf{v}_1\mathbf{R}$ and $\mathbf{v}_2\mathbf{R}$ are distributed independently and uniformly random. Consequently, it holds that

$$\mathsf{Pr}_{\mathbf{R}}[\mathbf{v}_1\mathbf{R} = \mathbf{v}_2\mathbf{R}\mathbf{A} \text{ and } \mathbf{v}_2\mathbf{R}\mathbf{A} \neq a\mathbf{v}_2\mathbf{R}] \leq \mathsf{Pr}_{\mathbf{R}}[\mathbf{v}_1\mathbf{R} = \mathbf{v}_2\mathbf{R}\mathbf{A}] \leq 1/q^m.$$

Note that in this case there are less than q^{2n} choices for the pair $\mathbf{v}_1, \mathbf{v}_2$.

We can conclude that

$$\mathsf{Pr}_{\mathbf{R}}[\exists \mathbf{v}_1, \mathbf{v}_2 \text{ s.t. } \mathbf{v}_1\mathbf{R} = \mathbf{v}_2\mathbf{R}\mathbf{A} \text{ and } \mathbf{v}_2\mathbf{R}\mathbf{A} \neq a\mathbf{v}_2\mathbf{R}] \leq q^{n+1}q^{-m/2} + q^{2n}q^{-m}.$$

As $m \geq 2n + 2 + 2t$, we can bound this probability by $2q^{-t}$.

7 From Output-Bounded Functions to Linear Combinations

In this section, we will show that the AR code induced by the inner product extractor restricts arbitrary functions of bounded output length to linear functions. Specifically, consider the following two classes of functions:

– The class $\mathcal{F}$ consists of all functions $f : (\mathbb{F}_q)^t \to \mathbb{F}_q$ of the form $f(x_1, \ldots, x_t) = \sum_{i=1}^t a_i x_i$, where $a_1, \ldots, a_t \in \mathbb{F}_q$
– The class $\mathcal{G}$ consists of all functions $g : (\mathbb{F}_q^n)^t \to \{0,1\}^{n\log(q)+s}$ (for some $s < n\log(q)$).

Let $s = \mathbf{s} \leftarrow_\$ \mathbb{F}_q^n$ be a uniformly random vector. The AR code AR_2 is given as follows.

$\mathsf{Encode}(s, x_1, \ldots, x_t)$**:**
  – Choose $\mathbf{x}_1, \ldots, \mathbf{x}_t \leftarrow_\$ \mathbb{F}_q^n$ uniformly at random under the restriction that $\langle \mathbf{x}_i, \mathbf{s} \rangle = x_i$ for all $i \in [t]$.
  – Output $c = (\mathbf{x}_1, \ldots, \mathbf{x}_t)$
$\mathsf{Eval}(c, \mathbf{a})$**:**
  – Parse $c = (\mathbf{x}_1, \ldots, \mathbf{x}_t)$ and $\mathbf{a} = (a_1, \ldots, a_t)$.
  – Compute and output $\mathbf{y} \leftarrow \sum_{i=1}^t a_i \mathbf{x}_i$
$\mathsf{Decode}(s, \mathbf{y})$**:**
  – Compute and output $y \leftarrow \langle \mathbf{y}, \mathbf{s} \rangle$.

Restriction security of this construction follows immediately from Corollary 1 at the end of this section.

7.1 A Conditional XOR Lemma

The following is straightforward from a Markov-like argument.

Lemma 9.

– *For any $\varepsilon > 0$, and any correlated random variables $X \in S$ and E if*

$$\Delta(X, E\ ;\ U, E) \leq \varepsilon\ ,$$

then for any $\delta > 0$, with probability at least $1 - \frac{\varepsilon}{\delta}$ over the choice of $i \leftarrow E$,

$$\Delta(X|_{E=i}\ ;\ U) \leq \delta\ .$$

– *For any $\delta > 0$, if*

$$\Delta(X|_{E=i} \; ; \; U) \leq \delta$$

holds with probability at least p over the choice of $i \leftarrow E$, then

$$\Delta(X, E \; ; \; U, E) \leq \delta + (1 - \delta) \cdot (1 - p) \; .$$

Lemma 10. *Let $X \in S$ be a random variable for some set S. Assume that $\Delta(X \; ; \; U_S) = \varepsilon$. Then if X' is an i.i.d copy of X then*

$$4\varepsilon^2 \geq \mathsf{Pr}[X = X'] - \frac{1}{|S|} \geq \frac{4\varepsilon^2}{|S|}.$$

Proof. Let $p_x = \mathsf{Pr}[X = x]$ for $x \in S$. Then

$$\mathsf{Pr}[X = X'] - \frac{1}{|S|} = \sum_{x \in S} \left(p_x - \frac{1}{|S|} \right)^2 \geq \frac{1}{|S|} \left(\sum_{x \in S} \left| p_x - \frac{1}{|S|} \right| \right)^2 = \frac{4\varepsilon^2}{|S|} \; .$$

Also,

$$\mathsf{Pr}[X = X'] - \frac{1}{|S|} = \sum_{x \in \mathbb{F}} \left(p_x - \frac{1}{|S|} \right)^2 \leq \left(\sum_{x \in S} \left| p_x - \frac{1}{|S|} \right| \right)^2 = 4\varepsilon^2 \; .$$

□

Lemma 11. *Let $X \in S$, $Z \in T$ be correlated random variables for some sets S, T. Assume that $\Delta(X, Z \; ; \; U_S, Z) = \varepsilon$. Then if (X', Z') is an i.i.d copy of (X, Z) then*

$$4\varepsilon^2 \geq \mathsf{Pr}[X = X', Z = Z'] - \frac{1}{|S|}\mathsf{Pr}[Z = Z'] \geq \frac{4\varepsilon^2}{|S| \cdot |T|}.$$

Proof. Let $p_z = \mathsf{Pr}[Z = z]$, and let $p_{x,z} = \mathsf{Pr}[X = x, Z = z]$. Then

$$\begin{aligned}
\mathsf{Pr}[X = X', Z = Z'] - \frac{1}{|S|} \cdot \mathsf{Pr}[Z = Z'] &= \sum_{x \in S, z \in T} \left(p_{x,z} - \frac{p_z}{|S|} \right)^2 \\
&\geq \frac{1}{|S| \cdot |T|} \left(\sum_{x \in S} \left| p_{x,z} - \frac{p_z}{|S|} \right| \right)^2 \\
&= \frac{4\varepsilon^2}{|S| \cdot |T|} \; .
\end{aligned}$$

Also,

$$\begin{aligned}
\mathsf{Pr}[X = X', Z = Z'] - \frac{1}{|S|} \cdot \mathsf{Pr}[Z = Z'] &= \sum_{x \in S, z \in T} \left(p_{x,z} - \frac{p_z}{|S|} \right)^2 \\
&\leq \left(\sum_{x \in S} \left| p_{x,z} - \frac{p_z}{|S|} \right| \right)^2 \\
&= 4\varepsilon^2 \; .
\end{aligned}$$

□

The following is a variant of the well known Vazirani's XOR lemma. This was proved in [ACLV19] in the quantum setting.

Lemma 12. *Let $\mathbf{x} = (x_1, \ldots, x_t) \in \mathbb{F}^t$ be a random variable, and E be some correlated random variable. Assume that for all $\alpha_1, \ldots, \alpha_t \in \mathbb{F}$ not all zero, $\Delta(\sum_{i=1}^t \alpha_i x_i, E \; ; \; u_{\mathbb{F}}, E) \le \varepsilon$. Then*

$$\Delta(\mathbf{x}, E \; ; \; \mathbf{u}_{\mathbb{F}^t}, E) \le 3p^{3t/4}\sqrt{\varepsilon} \; .$$

Proof. We start by choosing E and fixing it. By Lemma 9 and the union bound, we have that with probability at least $1 - p^t \frac{\varepsilon}{\delta}$ over the choice of E, we have that for all $\alpha_1, \ldots, \alpha_t \in \mathbb{F}$ not all zero,

$$\Delta(\sum_{i=1}^t \alpha_i x_i \; ; \; u_{\mathbb{F}}) \le \delta \; ,$$

where the distribution of x_i's is conditioned on the choice of E. Let $\mathbf{x}' = (x'_1, \ldots, x'_t)$ be i.i.d. as $\mathbf{x}$ conditioned on the choice of E. By Lemma 10, we have that for all $\alpha_1, \ldots, \alpha_t \in \mathbb{F}$ not all zero,

$$\mathsf{Pr}(\sum_{i=1}^t \alpha_i (x_i - x'_i) = 0) \le \frac{1}{p} + 4\delta^2 \; .$$

Let $\mathbf{a} = (a_1, \ldots, a_t)$ be uniform in $\mathbb{F}^t$ and independent of $\mathbf{x}, \mathbf{x}'$. Then,

$$\begin{aligned} & \mathsf{Pr}[\sum_{i=1}^t a_i (x_i - x'_i) = 0] \\ &= \mathsf{Pr}[\sum_{i=1}^t a_i (x_i - x'_i) = 0 | \mathbf{a} \neq 0] \cdot \mathsf{Pr}[\mathbf{a} \neq 0] + \mathsf{Pr}[\mathbf{a} = 0] \\ &\le \left(\frac{1}{p} + 4\delta^2\right) \cdot \left(1 - \frac{1}{p^t}\right) + \frac{1}{p^t} \; . \end{aligned}$$

Thus,

$$\begin{aligned} & \left(\frac{1}{p} + 4\delta^2\right) \cdot \left(1 - \frac{1}{p^t}\right) + \frac{1}{p^t} \\ &\ge \mathsf{Pr}[\sum_{i=1}^t a_i (x_i - x'_i) = 0] \\ &= \mathsf{Pr}[\sum_{i=1}^t a_i (x_i - x'_i) = 0 | \mathbf{x} \neq \mathbf{x}'] \cdot \mathsf{Pr}[\mathbf{x} \neq \mathbf{x}'] + \mathsf{Pr}[\mathbf{x} = \mathbf{x}'] \\ &= \mathsf{Pr}[\mathbf{x} = \mathbf{x}'] + \frac{1}{p} \cdot (1 - \mathsf{Pr}[\mathbf{x} = \mathbf{x}']) \; . \end{aligned}$$

Simplifying, we get,

$$\mathsf{Pr}[\mathbf{x} = \mathbf{x}'] \le \frac{1}{p^t} + 4\delta^2 \frac{1 - 1/p^t}{1 - 1/p} \le \frac{1}{p^t} + 8\delta^2 \; .$$

Using the inequality in Lemma 10, we get that

$$\Delta(\mathbf{x} \; ; \; \mathbf{u}_{\mathbb{F}^t}) \le \sqrt{2\delta^2 p^t} \le 2\delta \cdot p^{t/2} \; .$$

Recall that this is conditioned on the correct choice of E which we have with probability at least $1 - p^t \frac{\varepsilon}{\delta}$. Using Lemma 9 with $\delta = p^{t/4}\sqrt{\varepsilon}$, we have that

$$\Delta(\mathbf{x}, E \; ; \; \mathbf{u}_{\mathbb{F}^t}, E) \le 2\delta \cdot p^{t/2} + p^t \frac{\varepsilon}{\delta} = 3p^{3t/4}\sqrt{\varepsilon} \; .$$

□

We remark here that if there is no side information E, then there is no union bound in the first step, and $\delta = \varepsilon$ so that the statistical distance is $2p^{t/2}\varepsilon$.

7.2 Combinatorial Simulator

We now prove our main technical result, which yields algebraic restriction codes for functions of bounded output length.

Theorem 5. *Let q be a prime power, let n, t, s be positive integers and $\varepsilon > 0$ such that*

$$n \log q - (9t+3)\log q - s - 2\log t - 28 \geq 16 \log \frac{1}{\varepsilon} \ .$$

Let $\mathbf{x}_1, \ldots, \mathbf{x}_t$ be uniform in $\mathbb{F}_q^n$ and $\mathbf{s}$ is uniform in $\mathbb{F}_q^n$ and independent of the $\mathbf{x}_i$. For any $f : \mathbb{F}_q^{tn} \to \{0,1\}^{n \log q + s}$, there exists a simulator Sim and random variables $a_1, \ldots, a_t \in \mathbb{F}_q$ such that

$$\begin{aligned} &\mathbf{s}, f(\mathbf{x}_1, \ldots, \mathbf{x}_t), \langle \mathbf{x}_1, \mathbf{s}\rangle, \ldots, \langle \mathbf{x}_t, \mathbf{s}\rangle, a_1, \ldots, a_t \\ \approx_{2\varepsilon}\ &\mathbf{s}, \mathsf{Sim}\left(\mathbf{s}, a_1, \ldots, a_t, \sum_{i=1}^{t} a_i u_i\right), u_1, \ldots, u_t, a_1, \ldots, a_t \end{aligned}$$

where $u_1, \ldots, u_t$ are uniform and independent random variables in $\mathbb{F}_q$, independent of $(a_1, \ldots, a_t)$.

We will use the XOR lemma (Lemma 12) to prove this theorem. We will begin by showing that if we start with $\mathbf{x}_1, \ldots, \mathbf{x}_t$ being uniform in any large enough set $\mathcal{T}$, then there exists a large subset $\mathcal{T}' \subseteq \mathcal{T}$ and some fixed $a_1, \ldots, a_t$ in $\mathbb{F}_q$ such that conditioned on $(\mathbf{x}_1, \ldots, \mathbf{x}_t)$ being in this subset, the only information about $\langle \mathbf{x}_1, \mathbf{s}\rangle, \ldots, \langle \mathbf{x}_t, \mathbf{s}\rangle$ obtained by learning $f(\mathbf{x}_1, \ldots, \mathbf{x}_t)$ and S is $\sum_{i=1}^{t} a_i \langle \mathbf{x}_i, \mathbf{s}\rangle$. More formally,

Lemma 13. *Let q be a prime power, let n, t, s be positive integers and $\varepsilon > 0$ such that*

$$n \log q - (9t+3)\log q - s - 2\log t - 28 \geq 16 \log \frac{1}{\varepsilon} \ .$$

Let $\mathcal{T} \subseteq \mathbb{F}_q^{tn}$ such that $|\mathcal{T}| \geq \varepsilon \cdot q^{tn}$. For any $f : \mathbb{F}_q^{tn} \to \{0,1\}^{n \log q + s}$, there exist $a_1, \ldots, a_t \in \mathbb{F}_q$, non-empty set $\mathcal{T}' \subseteq \mathcal{T}$, and a simulator Sim, such that for the tuple $(\mathbf{x}_1, \ldots, \mathbf{x}_t)$ distributed uniformly in $\mathcal{T}'$ and $\mathbf{s}$ uniformly and independently in in $\mathbb{F}_q^n$,

$$\Delta\left(\mathbf{s}, f(\mathbf{x}_1, \ldots, \mathbf{x}_t), \langle \mathbf{x}_1, \mathbf{s}\rangle, \ldots, \langle \mathbf{x}_t, \mathbf{s}\rangle \ ; \ \mathbf{s}, \mathsf{Sim}\left(\mathbf{s}, \sum_{i=1}^{t} a_i u_i\right), u_1, \ldots, u_t\right) \leq \varepsilon \ ,$$

where $u_1, \ldots, u_t$ are uniform and independent random variables in $\mathbb{F}_q$.

Proof. Let $\mathbf{x}_1, \ldots, \mathbf{x}_t$ be uniform in $\mathcal{T}$. Consider the following cases.

CASE 1: $\Delta(\mathbf{x}, f(\mathbf{x}_1, \ldots, \mathbf{x}_t), \langle \mathbf{x}_1, \mathbf{s}\rangle, \ldots, \langle \mathbf{x}_t, \mathbf{s}\rangle \ ; \ \mathbf{s}, f(\mathbf{x}_1, \ldots, \mathbf{x}_t), u_1, \ldots, u_t) \leq \varepsilon$.
In this case, let $\mathcal{T}_1 = \mathcal{T}$. The simulator Sim ignores the inputs and just samples $\mathbf{s}, \mathbf{x}_1, \ldots, \mathbf{x}_t$ according to the given input distribution, and outputs $\mathbf{s}, f(\mathbf{x}_1, \ldots, \mathbf{x}_t)$. Thus, the given statement implies

$$\Delta\left(\mathbf{s}, f(\mathbf{x}_1, \ldots, \mathbf{x}_t), \langle \mathbf{x}_1, \mathbf{s}\rangle, \ldots, \langle \mathbf{x}_t, \mathbf{s}\rangle \ ; \ \mathbf{s}, \mathsf{Sim}(\mathbf{s}, \sum_{i=1}^{t} a_i u_i), u_1, \ldots, u_t\right) \leq \varepsilon \ ,$$

Notice that since the simulator ignores the input, the above statement holds for any choice of $a_1, \ldots, a_t$.

CASE 2: $\Delta(\mathbf{s}, f(\mathbf{x}_1, \ldots, \mathbf{x}_t), \langle \mathbf{x}_1, \mathbf{s}\rangle, \ldots, \langle \mathbf{x}_t, \mathbf{s}\rangle \ ; \ \mathbf{s}, f(\mathbf{x}_1, \ldots, \mathbf{x}_t), u_1, \ldots, u_t) > \varepsilon$.
Lemma 12 shows that if all non-trivial linear combinations of $\langle \mathbf{x}_i, \mathbf{s}\rangle$ are close to uniform given $E = (\mathbf{s}, f(\mathbf{x}_1, \ldots, \mathbf{s}_t))$, then the joint distribution $\langle \mathbf{x}_1, \mathbf{s}\rangle, \ldots, \langle \mathbf{x}_t, \mathbf{s}\rangle$ is close to uniform given E. Applying the contrapositive, we get that there exists $a_1, \ldots, a_t \in \mathbb{F}_q$, not all 0, such that

$$\Delta(\langle \sum_{i=1}^{t} a_i \mathbf{x}_i, \mathbf{s}\rangle, \mathbf{s}, f(\mathbf{x}_1, \ldots, \mathbf{x}_t) \ ; \ u, \mathbf{s}, f(\mathbf{x}_1, \ldots, \mathbf{x}_t)) > \frac{\varepsilon^2}{9q^{3t/2}} \ .$$

Notice that this implies that there is a non-trivial correlation between $\sum_{i=1}^t a_i \langle \mathbf{x}_i, \mathbf{s}\rangle$ and $(\mathbf{s}, f(\mathbf{x}_1, \dots, \mathbf{x}_t))$. We will show that for this choice of $a_1, \dots, a_t$, and an appropriate choice of the subset $\mathcal{T}'$, this correlation is essentially the only correlation between the joint distribution $(\langle \mathbf{x}_1, \mathbf{s}\rangle, \dots, \langle \mathbf{x}_t, \mathbf{s}\rangle)$ and $(\mathbf{s}, f(\mathbf{x}_1, \dots, \mathbf{x}_t))$.
By the Markov inequality, with probability at least $\frac{\varepsilon^2}{18q^{3t/2}}$ over the choice of $y \leftarrow f(\mathbf{x}_1, \dots, \mathbf{x}_t)$ iy holds that

$$\Delta(\langle \sum_{i=1}^t a_i \mathbf{x}_i, \mathbf{s}\rangle, \mathbf{s}|_{f(\mathbf{x}_1,\dots,\mathbf{x}_t)=y} \ ; \ u, \mathbf{s}) > \frac{\varepsilon^2}{18q^{3t/2}} \ . \tag{5}$$

Let $\mathcal{Y}$ be the set of all y which satisfy the above. For all $y \in \mathcal{Y}$, let $\mathcal{T}_y$ be the preimage of y for the function f, i.e., the set of all $(\mathbf{x}_1, \dots, \mathbf{x}_t)$ such that $f(\mathbf{x}_1, \dots, \mathbf{x}_t) = y$. We have that an element chosen uniformly at random from $\mathcal{T}$ is in $\mathcal{T}_y$ for some $y \in \mathcal{Y}$ with probability at least $\frac{\varepsilon^2}{18q^{3t/2}}$. This implies that

$$\left| \bigcup_{y \in \mathcal{Y}} \mathcal{T}_y \right| \geq \frac{\varepsilon^2}{18q^{3t/2}} \cdot |\mathcal{T}| \geq \frac{\varepsilon^3}{18q^{3t/2}} \cdot q^{tn} \ .$$

Let y be some element in $\mathcal{Y}$. By the contrapositive of the leftover hash lemma (Lemma 2), we have that the min-entropy of $\sum_{i=1}^t a_i \mathbf{x}_i$ conditioned on $f(\mathbf{x}_1, \dots, \mathbf{x}_t) = y$ is at most $\log q + 2\log \frac{1}{\Delta}$, where

$$\Delta := \Delta(\langle \sum_{i=1}^t a_i \mathbf{x}_i, \mathbf{s}\rangle, \mathbf{s}|_{f(\mathbf{x}_1,\dots,\mathbf{x}_t)=y} \ ; \ u, \mathbf{s}) > \frac{\varepsilon^2}{18q^{3t/2}} \ ,$$

using the inequality in 5. Thus

$$\mathbf{H}_\infty(\sum_{i=1}^t a_i \mathbf{x}_i | f(\mathbf{x}_1, \dots, \mathbf{x}_t) = y) \leq \log q + 4 \log \frac{1}{\varepsilon} + 3t \log q + 2 \log 18 \ . \tag{6}$$

This implies that for each $y \in \mathcal{Y}$, there is a large number of elements $(\mathbf{x}_1, \dots, \mathbf{x}_t) \in \mathcal{T}_y$ such that $\sum_{i=1}^t a_i \mathbf{x}_i$ is fixed. We now select only those elements from $\mathcal{T}_y$ which correspond $\sum_{i=1}^t a_i \mathbf{x}_i$ being fixed. For each $y \in \mathcal{Y}$, let $\phi(y)$ be the most frequently occurring value of $\sum_{i=1}^t a_i \mathbf{x}_i$ for $(\mathbf{x}_1, \dots, \mathbf{x}_t) \in \mathcal{T}_y$, and let

$$\mathcal{T}'_y = \left\{ (\mathbf{x}_1, \dots, \mathbf{x}_t) \in \mathcal{T}_y \ \middle| \ \sum_{i=1}^t a_i \mathbf{x}_i = \phi(y) \right\} \ .$$

By the inequality in 6, we have that

$$|\mathcal{T}'_y| \geq |\mathcal{T}_y| \cdot \frac{\varepsilon^4}{324q^{3t+1}} \ ,$$

which implies that

$$\left| \bigcup_{y \in \mathcal{Y}} \mathcal{T}'_y \right| \geq \frac{\varepsilon^3}{18q^{3t/2}} \cdot q^{tn} \cdot \frac{\varepsilon^4}{324q^{3t+1}} \geq \frac{\varepsilon^7}{2^{13}q^{9t/2+1}} \cdot q^{tn} \ . \tag{7}$$

Notice that for any $(\mathbf{x}_1, \dots, \mathbf{x}_t)$ in $\bigcup_{y \in \mathcal{Y}} \mathcal{T}'_y$, it holds that $\sum_{i=1}^t a_i \mathbf{x}_i$ is equal to $\phi(f(\mathbf{x}_1, \dots, \mathbf{x}_t))$, i.e., it is uniquely determined given $f(\mathbf{x}_1, \dots, \mathbf{x}_t)$.
Intuitively, since $\sum_{i=1}^t a_i \mathbf{x}_i$ carries roughly $n \log p$ bits of information, and it is a deterministic function of $f(\mathbf{x}_1, \dots, \mathbf{x}_t)$, we expect that $f(\mathbf{x}_1, \dots, \mathbf{x}_t)$ can be uniquely determined with (a little more than an) additional s bits of information. We will now remove those elements for which it requires a large number of bits to determine $f(\mathbf{x}_1, \dots, \mathbf{x}_t)$ given $\sum_{i=1}^t a_i \mathbf{x}_i$.
Let $\mathcal{Y}'$ be the set of all y such that

$$|\phi^{-1}(\phi(y))| \leq \frac{2q^{tn}}{|\bigcup_{y \in \mathcal{Y}} \mathcal{T}'_y|} \cdot 2^s \ .$$

For $y \in \mathcal{Y} \setminus \mathcal{Y}'$

$$|\phi^{-1}(\phi(y))| > \frac{2q^{tn}}{|\bigcup_{y\in\mathcal{Y}} \mathcal{T}'_y|} \cdot 2^s \ .$$

The total number of elements in $\mathcal{Y} \setminus \mathcal{Y}'$ is at most the size of the image of f, i.e., $q^n \cdot 2^s$. Hence the number of distinct values of $\phi(y)$ for $y \in \mathcal{Y} \setminus \mathcal{Y}'$ is at most

$$\frac{q^n \cdot 2^s \cdot |\bigcup_{y\in\mathcal{Y}} \mathcal{T}'_y|}{2q^{tn} \cdot 2^s} = \frac{|\bigcup_{y\in\mathcal{Y}} \mathcal{T}'_y|}{2q^{(t-1)n}} \ .$$

Notice that for any element $\mathbf{z} \in \mathbb{F}_q^n$, there are at most $q^{(t-1)n}$ values of $(\mathbf{x}_1, \ldots, \mathbf{x}_t)$ such that $\sum_{i=1}^t a_i \mathbf{x}_i = \mathbf{z}$. Thus, the number of elements in $\bigcup_{y\in\phi^{-1}(\mathbf{z})} \mathcal{T}_y$ is at most $q^{(t-1)n}$. This implies that

$$\left| \bigcup_{y\in\mathcal{Y}\setminus\mathcal{Y}'} \mathcal{T}'_y \right| \leq q^{(t-1)n} \cdot \frac{|\bigcup_{y\in\mathcal{Y}} \mathcal{T}'_y|}{2q^{(t-1)n}} = \frac{|\bigcup_{y\in\mathcal{Y}} \mathcal{T}'_y|}{2} \ .$$

Thus,

$$\left| \bigcup_{y\in\mathcal{Y}'} \mathcal{T}'_y \right| \geq \frac{|\bigcup_{y\in\mathcal{Y}} \mathcal{T}'_y|}{2} \ .$$

We let the set $\mathcal{T}'$ be $\bigcup_{y\in\mathcal{Y}'} \mathcal{T}'_y$, and let $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_t$ be uniform in $\mathcal{T}'$. We have the following two properties satisfied by $(\mathbf{x}_1, \ldots, \mathbf{x}_t)$:

– The random variable $\sum_{i=1}^t a_i \mathbf{x}_i$ is a deterministic function of $f(\mathbf{x}_1, \ldots, \mathbf{x}_t)$.

– The random variable $f(\mathbf{x}_1, \ldots, \mathbf{x}_t)$ is uniquely determined given

$$\phi(f(\mathbf{x}_1, \ldots, \mathbf{x}_t)) = \sum_{i=1}^t a_i \mathbf{x}_i \text{ and } \psi(\mathbf{x}_1, \ldots, \mathbf{x}_t)$$

for some function $\psi : \mathbb{F}_q^{tn} \to \{0,1\}^{s+14+(9t/2+1)\log q + 7\log\frac{1}{\varepsilon}}$.[7]

Since not all $a_1, \ldots, a_t$ are 0, we assume without loss of generality that $a_t \neq 0$. Then, notice that

$$\begin{aligned}
&\mathbf{H}_\infty(\mathbf{x}_i | \mathbf{x}_1, \ldots, \mathbf{x}_{i-1}, \mathbf{x}_{i+1}, \ldots, \mathbf{x}_{t-1}, \sum_{i=1}^t a_i \mathbf{x}_i) \\
\geq& \log|\mathcal{T}'| - (t-1)n\log q \\
\geq& \log|\bigcup_{y\in\mathcal{Y}'} \mathcal{T}_y| - 1 - (t-1)n\log q \\
\geq& tn\log q - 7\log\frac{1}{\varepsilon} - 14 - (9t/2+1)\log q - (t-1)n\log q \\
=& n\log q - 7\log\frac{1}{\varepsilon} - 14 - (9t/2+1)\log q \ ,
\end{aligned}$$

where we used the inequality (7). Additionally, considering the additional leakage from $\psi(\mathbf{x}_1, \ldots, \mathbf{x}_t)$, we get that

$$\begin{aligned}
&\widetilde{\mathbf{H}}_\infty(\mathbf{x}_i | \mathbf{x}_1, \ldots, \mathbf{x}_{i-1}, \mathbf{x}_{i+1}, \ldots, \mathbf{x}_{t-1}, \phi(f(\mathbf{x}_1, \ldots, \mathbf{x}_t)), \psi(\mathbf{x}_1, \ldots, \mathbf{x}_t)) \\
=& n\log q - 14\log\frac{1}{\varepsilon} - 28 - (9t+2)\log q - s \\
\geq& \log q + 2\log\frac{t}{\varepsilon} \ ,
\end{aligned}$$

[7]Since for $y \in \mathcal{Y}'$ we get $\log|\phi^{-1}(\phi(y))| \leq \log\left(\frac{2q^{tn}}{|\bigcup_{y\in\mathcal{Y}} \mathcal{T}'_y|} \cdot 2^s\right) \leq s + 14 + (9t/2+1)\log q + 7\log\frac{1}{\varepsilon}$.

where we used the fact that the length of $\psi(\mathbf{x}_1, \ldots, \mathbf{x}_t)$ is at most $s + 14 + (9t/2 + 1)\log q + 7 \log \frac{1}{\varepsilon}$, and also the bound on $n \log q$ as given in the lemma statement. Restating with ϕ, ψ replaced by the function f, we get the following.

$$\widetilde{\mathbf{H}}_\infty(\mathbf{x}_i | \mathbf{x}_1, \ldots, \mathbf{x}_{i-1}, \mathbf{x}_{i+1}, \ldots, \mathbf{x}_{t-1}, f(\mathbf{x}_1, \ldots, \mathbf{x}_t)) \geq \log q + 2 \log \frac{t}{\varepsilon} \ .$$

By the leftover hash lemma (Lemma 2), we have that

$$\begin{aligned} &\langle \mathbf{x}_1, \mathbf{s} \rangle, \langle \mathbf{x}_2, \mathbf{s} \rangle, \ldots, \langle \mathbf{x}_{t-1}, \mathbf{s} \rangle, f(\mathbf{x}_1, \ldots, \mathbf{x}_t), \mathbf{s} \\ \approx_{\varepsilon/t}\ & u_1, \langle \mathbf{x}_2, \mathbf{s} \rangle, \ldots, \langle \mathbf{x}_{t-1}, \mathbf{s} \rangle, f(\mathbf{x}_1, \ldots, \mathbf{x}_t), \mathbf{x} \ . \end{aligned}$$

Similarly, for $i = 2, 3, 4, \ldots, t-1$

$$\begin{aligned} &u_1, \ldots, u_{i-1}, \langle \mathbf{x}_i, \mathbf{s} \rangle, \langle \mathbf{x}_{i+1}, \mathbf{s} \rangle, \ldots, \langle \mathbf{x}_{t-1}, \mathbf{s} \rangle, f(\mathbf{x}_1, \ldots, \mathbf{x}_t), \mathbf{s} \\ \approx_{\varepsilon/t}\ & u_1, \ldots, u_{i-1}, u_i, \langle \mathbf{x}_{i+1}, \mathbf{s} \rangle, \ldots, \langle \mathbf{x}_{t-1}, \mathbf{s} \rangle, f(\mathbf{x}_1, \ldots, \mathbf{x}_t), \mathbf{s} \ . \end{aligned}$$

By the triangle inequality, we get that

$$\begin{aligned} &\langle \mathbf{x}_1, \mathbf{s} \rangle, \ldots, \langle \mathbf{x}_{t-1}, \mathbf{s} \rangle, f(\mathbf{x}_1, \ldots, \mathbf{x}_t), \mathbf{s} \\ \approx_{\varepsilon(t-1)/t}\ & u_1, \ldots, u_{t-1}, f(\mathbf{x}_1, \ldots, \mathbf{x}_t), \mathbf{s} \ . \end{aligned}$$

Let Z be the additional randomness needed to sample $f(\mathbf{x}_1, \ldots, \mathbf{x}_t), S$ given $\langle \sum_{i=1}^t a_i \mathbf{x}_i, \mathbf{s} \rangle$, i.e., for some σ, $f(\mathbf{x}_1, \ldots, \mathbf{x}_t), \mathbf{s} \equiv \sigma(Z, \langle \sum_{i=1}^t a_i \mathbf{x}_i, \mathbf{s} \rangle)$. By the leftover hash lemma (Lemma 2), we have that

$$\langle \sum_{i=1}^t a_i \mathbf{x}_i, \mathbf{s} \rangle, f(\mathbf{x}_1, \ldots, \mathbf{x}_t), \mathbf{s} \approx_{\varepsilon/t} u, \mathsf{Sim}(u, Z) \ .$$

Again by the triangle inequality, we have that

$$\begin{aligned} &\langle \mathbf{x}_1, \mathbf{s} \rangle, \ldots, \langle \mathbf{x}_{t-1}, \mathbf{s} \rangle, \langle \sum_{i=1}^t a_i \mathbf{x}_i, \mathbf{s} \rangle, f(\mathbf{x}_1, \ldots, \mathbf{x}_t), \mathbf{s} \\ \approx_\varepsilon\ & U_1, \ldots, U_{t-1}, U, \sigma(U, Z) \ . \end{aligned}$$

Writing $\langle \mathbf{x}_t, \mathbf{s} \rangle$ as $\frac{1}{a_t}\left(\langle \sum_{i=1}^t a_i \mathbf{x}_i, \mathbf{s} \rangle - \sum_{i=1}^{t-1} a_i \langle \mathbf{x}_i, \mathbf{s} \rangle \right)$ and applying Lemma 4, we have that

$$\begin{aligned} &\langle \mathbf{x}_1, \mathbf{s} \rangle, \ldots, \langle \mathbf{x}_t, \mathbf{s} \rangle, f(\mathbf{x}_1, \ldots, \mathbf{x}_t), \mathbf{s} \\ \approx_\varepsilon\ & u_1, \ldots, u_{t-1}, \frac{1}{a_t}(u - \sum_{i=1}^{t-1} a_i u_i), \mathsf{Sim}(\mathbf{s}, u, Z), \mathbf{s} \ . \end{aligned}$$

Notice that $u_t := \frac{1}{a_t}(u - \sum_{i=1}^{t-1} a_i u_i)$ is uniform in $\mathbb{F}_q$ and independent of $u_1, \ldots, u_{t-1}$. Rearranging, we have that $u = \sum_{i=1}^t a_i u_i$. Thus, we obtain

$$\langle \mathbf{x}_1, \mathbf{s} \rangle, \ldots, \langle \mathbf{x}_t, \mathbf{s} \rangle, f(\mathbf{x}_1, \ldots, \mathbf{x}_t), \mathbf{s} \approx_\varepsilon u_1, \ldots, u_{t-1}, u_t, \mathsf{Sim}(\mathbf{s}, \sum_{i=1}^t a_i u_i, Z), \mathbf{s} \ ,$$

as needed.

□

We are now ready to complete the proof of Theorem 5.

Proof (Proof of Theorem 5). From Lemma 1, there exists a set $\mathcal{T}_1$, simulator Sim_1 and $a_1^{(1)}, \ldots, a_t^{(1)}$ such that for $(\mathbf{x}_1^{(1)}, \ldots, \mathbf{x}_t^{(1)})$ distributed uniformly in $\mathcal{T}_1$,

$$\begin{aligned} &\mathbf{s}, f(\mathbf{x}_1^{(1)}, \ldots, \mathbf{x}_t^{(1)}), \langle \mathbf{x}_1^{(1)}, \mathbf{s} \rangle, \ldots, \langle \mathbf{x}_t^{(1)}, \mathbf{s} \rangle \\ &\approx_\varepsilon \sigma_1 \left(\sum_{i=1}^{t} a_i^{(1)} u_i \right), u_1, \ldots, u_t \ . \end{aligned}$$

Since $a_1^{(1)}, \ldots, a_t^{(1)}$ are fixed and *public* we can rewrite above as:

$$\begin{aligned} &\mathbf{s}, f(\mathbf{x}_1^{(1)}, \ldots, \mathbf{x}_t^{(1)}), \langle \mathbf{x}_1^{(1)}, \mathbf{s} \rangle, \ldots, \langle \mathbf{x}_t^{(1)}, \mathbf{s} \rangle, a_1^{(1)}, \ldots, a_t^{(1)} \\ &\approx_\varepsilon \sigma_1 \left(\sum_{i=1}^{t} a_i^{(1)} u_i \right), u_1, \ldots, u_t, a_1^{(1)}, \ldots, a_t^{(1)} \ . \end{aligned}$$

If $|\mathbb{F}_q^{tn} \setminus \mathcal{T}_1| \geq \varepsilon q^{tn}$, then we again apply Lemma 1 to obtain a set $\mathcal{T}_2$, simulator Sim_2 and $a_1^{(2)}, \ldots, a_t^{(2)}$ such that for $(\mathbf{x}_1^{(2)}, \ldots, \mathbf{x}_t^{(2)})$ distributed uniformly in $\mathcal{T}_2$,

$$\begin{aligned} &\mathbf{s}, f(\mathbf{x}_1^{(2)}, \ldots, \mathbf{x}_t^{(2)}), \langle \mathbf{x}_1^{(2)}, \mathbf{s} \rangle, \ldots, \langle \mathbf{x}_t^{(2)}, \mathbf{s} \rangle, a_1^{(2)}, \ldots, a_t^{(2)} \\ &\approx_\varepsilon \sigma_2 \left(\sum_{i=1}^{t} a_i^{(2)} u_i \right), u_1, \ldots, u_t, a_1^{(2)}, \ldots, a_t^{(2)} \ . \end{aligned}$$

We continue to obtain sets $\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_\ell$ by applying Lemma 1, until $|\mathcal{T}_\ell| < \varepsilon q^{tn}$. Let the random variable $(a_1, \ldots, a_t)$ and the simulator σ be the tuple $(a_1^{(j)}, \ldots, a_t^{(j)})$ and σ_j with probability proportional to the size of $\mathcal{T}_j$, i.e., $\frac{|\mathcal{T}_j|}{|\mathcal{T}|}$. Then, by Lemma 3, we obtain the desired result. □

Corollary 1. *Let q be a prime power, let n, t, s be positive integers and $\varepsilon > 0$ such that*

$$n \log q - (25t + 3) \log q - s - 2 \log t - 60 \geq 16 \log \frac{1}{\varepsilon'} \ .$$

Let $m_1, \ldots, m_t \in \mathbb{F}_q$. Let $\mathbf{s}$ be uniform in $\mathbb{F}_q^n$ and let $\mathbf{x}_1, \ldots, \mathbf{x}_t$ be sampled uniformly in $\mathbb{F}_q^n$ conditioned on the event that for all $i \in [t]$, $\langle \mathbf{x}_i, \mathbf{s} \rangle = m_i$. For any $f : \mathbb{F}_q^{tn} \to \{0,1\}^{n \log q + s}$, there exists a simulator Sim and random variables $a_1, \ldots, a_t \in \mathbb{F}_q$ such that

$$\begin{aligned} &\mathbf{s}, f(\mathbf{x}_1, \ldots, \mathbf{x}_t), a_1, \ldots, a_t \\ &\approx_{\varepsilon'} \mathbf{s}, \mathsf{Sim} \left(\mathbf{s}, a_1, \ldots, a_t, \sum_{i=1}^{t} a_i m_i \right), a_1, \ldots, a_t \end{aligned}$$

where $u_1, \ldots, u_t$ are uniform and independent random variables in $\mathbb{F}_q$, independent of $(a_1, \ldots, a_t)$.

Proof. Applying Lemma 5 to Theorem 5, and conditioning on $u_1 = m_1, \ldots, u_t = m_t$, we get that if

$$n \log q - (9t + 3) \log q - s - 2 \log t - 28 \geq 16 \log \frac{1}{\varepsilon} \ , \tag{8}$$

then

$$\begin{aligned} &\mathbf{s}, f(\mathbf{x}_1, \ldots, \mathbf{x}_t), a_1, \ldots, a_t \\ &\approx_{\varepsilon'} \mathbf{s}, \mathsf{Sim} \left(\mathbf{s}, a_1, \ldots, a_t, \sum_{i=1}^{t} a_i m_i \right), a_1, \ldots, a_t \ , \end{aligned}$$

where $\varepsilon' = 4\varepsilon q^t$, or in other words, $\varepsilon = \frac{\varepsilon'}{4q^t}$. Substituting ε in Equation (8) gives the desired result. □

8 Rate-1 SSP OT from DDH

In this section, we discuss the standard definition of rate-1 statistical sender-private oblivious transfer (rate-1 SSP OT) and then go over our construction using algebraic restriction codes. We start by providing the necessary cryptographic definitions.

8.1 Decisional Diffie-Hellman Assumption

These assumptions below are with regard to a group-generator scheme while most protocols just consider the group. This however, is just to make the notation in the protocol easier. Each protocol-participating party just chooses the group $\mathbb{G}$ according to the publicly known group-generator scheme G and security parameter λ and proceeds as detailed in the protocol.

We recall the definition of the decisional Diffie-Hellman assumption [DH76] (DDH).

Definition 7 (DDH). *Let* G *be a group-generator scheme, which on input* 1^λ *outputs* $(\mathbb{G}, p, g)$. *The decisional Diffie-Hellman assumption holds for group-generator scheme* G *if for all polynomial time adversaries* $\mathcal{A}$

$$\left|\mathsf{Pr}\left[\mathcal{A}(g, g^a, b, ab) = 1\right] - \mathsf{Pr}\left[\mathcal{A}(g, g^a, g^b, g^c) = 1\right]\right|$$

is negligible in λ *for* $(\mathbb{G}, p, g) \leftarrow \mathsf{G}(1^\lambda)$ *and uniformly random* $a, b, c \in \mathbb{Z}_p$

8.2 Public-Key Encryption Schemes

A public-key encryption scheme uses two keys, a public key pk and a secret key sk. We use the public key to encrypt messages, the result of which is called ciphertext. Without knowledge of the secret key, it is virtually impossible to calculate the message from the ciphertext. The secret key, however, enables the holder to reliably retrieve the message from the ciphertext.

Definition 8 (Public-Key Encryption). *The following algorithms describe a public-key encryption scheme:*

$\mathsf{KeyGen}(1^\lambda)$: *The key-generation algorithm takes the security parameter* λ *as input and outputs a key pair* $(\mathsf{pk}, \mathsf{sk})$.

$\mathsf{Enc}(\mathsf{pk}, m)$: *The encryption algorithm takes a public key* pk *and a message* m *as input and outputs a ciphertext* c.

$\mathsf{Dec}(\mathsf{sk}, c)$: *The decryption algorithm takes a secret key* sk *and a ciphertext* c *as input and outputs a message* m. *It rarely requires randomness.*

In the rest of the document, every encryption scheme will be public key. Therefore we will not mention it again.

Definition 9 (Correctness). *An encryption scheme* $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *is correct if for all message* m *and security parameters* λ

$$\mathsf{Pr}\left[m = \mathsf{Dec}(\mathsf{sk}, \mathsf{Enc}(\mathsf{pk}, m)) \middle| (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)\right] = 1$$

The most popular notion of security for encryption schemes is IND-CPA security.

Definition 10 (IND-CPA Security). *An encryption scheme* $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *is ind-cpa secure if for all adversary pairs* $(\mathcal{A}_1, \mathcal{A}_2)$

$$\mathsf{Pr}\left[b = b' \middle| \begin{array}{l} (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda) \\ (m_0, m_1, \sigma) \leftarrow \mathcal{A}_1(1^\lambda, \mathsf{pk}) \\ b \leftarrow_{\$} \{0, 1\} \\ b' \leftarrow \mathcal{A}_2(m_b, \sigma) \end{array} \right] - \frac{1}{2}$$

is negligible in λ

The rate is trying to capture the size comparison between a ciphertext and its corresponding plaintext.

Definition 11 (Rate). *An encryption scheme* (KeyGen, Enc, Dec) *has rate ρ if there exists a polynomial μ such that for all security parameters λ, possible outputs of* KeyGen(1^λ) *called* (pk, sk)*, and messages m with $|m| \geq \mu(\lambda)$*

$$\frac{|m|}{|\mathsf{Enc}(\mathsf{pk}, m)|} \geq \rho(\lambda)$$

We call an encryption scheme high rate if it has a rate greater than 1/2 and we call it rate-1 if for $\lambda \to \infty$ the rate $\rho(\lambda)$ approaches 1.

8.3 Homomorphic Encryption

In homomorphic encryption the decryption algorithm is a homomorphism. Certain changes on a ciphertext change the underlying plaintext in a structured way.

Definition 12 (Homomorphic Encryption). *These four algorithms describe a homomorphic encryption scheme:*

KeyGen(1^λ) : *The key-generation algorithm takes the security parameter λ as input and outputs a key pair* (pk, sk)*.*

Enc(pk, m) : *The encryption algorithm takes a public key* pk *and a message m as inputs and outputs a ciphertext c.*

Eval$(1^\lambda, \mathsf{pk}, f, c_1, ..., c_n)$: *The evaluation algorithm takes a security parameter λ, a public key* pk*, a string representation of a function f and n where n is the input size of f ciphertexts $c_1, \ldots, c_n$ as inputs and outputs a new ciphertext c.*

Dec(sk, c) : *The decryption algorithm takes a secret key* sk *and a ciphertext c as input and outputs a message m. It rarely requires randomness.*

Definition 13 (Homomorphic Correctness). *Let $\mathcal{F}$ be a set of functions and f be an arbitrary element of $\mathcal{F}$. An $\mathcal{F}$-homomorphic encryption scheme* (KeyGen, Enc, Eval, Dec) *is correct if* (KeyGen, Enc, Dec) *is a correct encryption scheme, and for all messages m, security parameters λ, and* (pk, sk) *from the support of* KeyGen(1^λ)

$$\Pr\left[f(m) = \mathsf{Dec}(\mathsf{sk}, \mathsf{Eval}(1^\lambda, \mathsf{pk}, f, \mathsf{Enc}(\mathsf{pk}, m)))\right] = 1$$

8.4 Oblivious Transfer

Two-round oblivious transfer is a protocol in which a receiver encodes a choice bit b and transmits it to a sender. The sender then responds to that transmission using its two messages m_0 and m_1. In the end the receiver learns m_b, but not m_{1-b} and the sender learns nothing.

Definition 14 (Oblivious Transfer). *A (string)* 1*-out-of-*2 *OT consists of three algorithms:* OT_1*,* OT_2*, and* OT_3*.*

$\mathsf{OT}_1(1^\lambda, b)$**:** *Takes as inputs the security parameter $\lambda \in \mathbb{N}$ and a choice bit $b \in \{0,1\}$ to produce a request ot_1 and a state st.*

$\mathsf{OT}_2(ot_1, (m_0, m_1))$**:** *Uses the request ot_1, and the two sender inputs $m_0, m_1 \in \{0,1\}^*$ of same length to create a response ot_2.*

$\mathsf{OT}_3(ot_2, st)$**:** *Calculates a result y from the state st and the response ot_2.*

We define correctness in the following.

Definition 15 (Correctness). *An OT is correct if for all security parameters λ, bits $b \in \{0,1\}$, and sender inputs $m_0, m_1 \in \{0,1\}^*$ the following holds:*

$$\Pr\left[y = s_b \;\middle|\; \begin{array}{l} ot_1, st \leftarrow \mathsf{OT}_1(1^\lambda, b) \\ ot_2 \leftarrow \mathsf{OT}_2(ot_1, (m_0, m_1)) \\ y \leftarrow \mathsf{OT}_3(ot_2, st) \end{array}\right] = 1.$$

As standard for 2-round OT, we require that the bit of the receiver is hidden in an indistinguishability sense.

Definition 16 (Receiver Security). *An OT is receiver secure if for all security parameters λ and PPT adversaries $\mathcal{A}$ the following holds:*

$$\left|\mathsf{Pr}\left[\mathcal{A}(ot_1)|ot_1, st \leftarrow \mathsf{OT}_1(1^\lambda, 0)\right] - \mathsf{Pr}\left[\mathcal{A}(ot_1)|ot_1, st \leftarrow \mathsf{OT}_1(1^\lambda, 1)\right]\right|$$

is negligible in λ.

We define (malicious) statistical sender privacy for 2-round OT.

Definition 17 (Statistical Sender Privacy). *An OT is statistically sender private if the exists a unbounded simulator Sim such that for all requests ot_1 and sender inputs $m_0, m_1 \in \{0,1\}^*$ the following holds:*

$$\Delta(\mathsf{OT}_2(ot_1, (m_0, m_1)); \mathsf{Sim}^f(1^\lambda, ot_1))$$

is negligible in λ with Sim having one-time access to an oracle $f : b \mapsto m_0 \cdot (1-b) + m_1 \cdot b$.

The (download) rate of an OT protocol captures how big the senders response is in comparison to the size of a message m_0.

Definition 18 (Rate). *An OT $(\mathsf{OT}_1, \mathsf{OT}_2, \mathsf{OT}_3)$ has rate ρ if there exists a polynomial μ such that for all security parameters λ,*

$$\frac{|m_0|}{|ot_2|} \geq \rho(\lambda)$$

for all choice bits b, message lengths $n > \mu(\lambda)$, sender inputs $m_0, m_1 \in \{0,1\}^n$, receiver outputs $(ot_1, st) \leftarrow \mathsf{OT}_1(1^\lambda, b)$ and sender outputs $ot_2 \leftarrow \mathsf{OT}_2(ot_1, (m_0, m_1))$

8.5 Packed ElGamal

A big component of our OT construction is the packed ElGamal encryption scheme, which we recall here for completeness. As discussed in the introduction, in the ElGamal encryption scheme [ElG84] public keys are of the form g, h and messages m are encrypted as $g^r, h^r \cdot m$. In the packed ElGamal scheme, the same *header* g^r is shared across several *payload* slots $h_i^r \cdot m_i$, effectively amortizing the cost of the header to encrypt an entire vector $\mathbf{m}$. Now let $\mathbb{G}$ be a cyclic group of prime order p, and let g be a generator of $\mathbb{G}$. In our description we will provide a decryption algorithm which takes as additional input a matrix $\mathbf{M} \in \mathbb{Z}_p^{m \times n}$, which is applied to the secret key before decryption. In this way, we achieve correctness for homomorphic operations across slots.

$\mathsf{KeyGen}(1^\lambda, n)$**:**
- Choose $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_p^n$ uniformly at random.
- Set $\mathbf{h} = g^{\mathbf{s}}$.
- Return secret key $\mathsf{sk} = \mathbf{s}$ and public key $\mathsf{pk} = \mathbf{h}$.

$\mathsf{Enc}(\mathsf{pk}, \mathbf{m} \in \{0,1\}^n)$**:**
- Parse $\mathsf{pk} = \mathbf{h} \in \mathbb{G}^n$.
- Choose $r \xleftarrow{\$} \mathbb{Z}_p$ uniformly at random.
- Return the ciphertext $\mathbf{c} = (g^r, \mathbf{h}^r \cdot g^{\mathbf{m}}) \in \mathbb{G} \times \mathbb{G}^n$.

For a matrix $\mathbf{X} = (\mathbf{x}_1, \dots, \mathbf{x}_m) \in \{0,1\}^{n \times m}$, we overload encryption and denote

$$\mathsf{Enc}(\mathsf{pk}, \mathbf{X}) = (\mathsf{Enc}(\mathsf{pk}, \mathbf{x}_1), \dots, \mathsf{Enc}(\mathsf{pk}, \mathbf{x}_m)).$$

$\mathsf{Dec}(\mathsf{sk}, \mathbf{M}, \mathbf{c})$**:**
- Parse $\mathsf{sk} = \mathbf{s}$.

  - Compute $\mathbf{s}' = \mathbf{M}\mathbf{s}$
  - Parse $\mathbf{c} = (c \in \mathbb{G}, \mathbf{e} \in \mathbb{G}^n)$.
  - Return $\mathsf{dlog}_g(\mathbf{e}/c^{\mathbf{s}'})$.

$\mathsf{Eval}_1(\mathsf{pk}, \mathbf{C}, \mathbf{a} \in \mathbb{Z}_p^m, \mathbf{b} \in \mathbb{Z}_p^n)$
  - Parse $\mathsf{pk} = \mathbf{h}$
  - Parse $\mathbf{C} = \mathbf{c}_1, \dots, \mathbf{c}_m$.
  - For all $i \in [m]$ parse $\mathbf{c}_i = (c_i, \mathbf{e}_i)$.
  - Choose $r \xleftarrow{\$} \mathbb{Z}_p$ uniformly at random.
  - Set $c = g^r \cdot \prod_{i=1}^m c_i^{a_i}$
  - Set $\mathbf{e} = \mathbf{h}^r (\prod_{i=1}^m \mathbf{e}_i^{a_i}) \cdot g^{\mathbf{b}}$
  - Return the ciphertext $\mathbf{c} = (c, \mathbf{e})$

$\mathsf{Eval}_2(\mathsf{pk}, \mathbf{c}, \mathbf{M} \in \mathbb{Z}_p^{m \times n})$
  - Parse $\mathbf{M} = (m_{ij})$
  - Parse $\mathbf{c} = (c, \mathbf{e})$ and $\mathbf{e} = (e_1, \dots, e_n)$
  - For all $i \in [m]$ compute $d_i = \prod_{j=1}^n e_j^{m_{ij}}$
  - Return the ciphertext $\mathbf{c}' = (c, \mathbf{d})$

For two ciphertexts $\mathbf{c}_1$ and $\mathbf{c}_2$ we overload Eval_1 and denote by $\mathsf{Eval}_1(\mathsf{pk}, \mathbf{c}_1, \mathbf{c}_2, -)$ the homomorphic computation of the difference of $\mathbf{c}_1$ and $\mathbf{c}_2$. Homomorphic correctness of this scheme follows routinely. To analyze the rate of this scheme, note that plaintexts $\mathbf{m} \in \mathbb{G}^n$ consist of n group elements, whereas ciphertexts consist of $n + 1$ group elements, i.e. there is an additive overhead of 1 group element and the rate of the scheme comes down to $1 - 1/n$. Thus the rate of the scheme approaches 1 for a growing n.

Lemma 14. *The packed ElGamal encryption scheme as described above is IND-CPA secure, given the DDH problem is hard for group $\mathbb{G}$.*

Proof. IND-CPA security of the packed ElGamal scheme follows tightly (in n) from the decisional Diffie Hellman assumption in a routine way: A DDH instance (g, h, g', h') can be rerandomized into a pair of vectors $\mathbf{h}$ and $\mathbf{f}$, such that $\mathbf{h}$ is distributed uniformly random in $\mathbb{G}^n$ and the following holds for $\mathbf{f}$. If (g', h') is of the form $r \cdot (g, h)$, then $\mathbf{f}$ is of the form $r \cdot \mathbf{h}$, whereas if (g', h') is uniformly random in $\mathbb{G}^2$, then $\mathbf{f}$ is uniformly random in $\mathbb{G}^n$. Given such an instance $(\mathbf{h}, \mathbf{f})$, a reduction can set $\mathsf{pk} = \mathbf{h}$ and $\mathbf{c} = \mathbf{f} \cdot (1, \mathbf{m})$. If $\mathbf{f}$ is of the form $\mathbf{f} = r \cdot \mathbf{h}$, then $\mathbf{c}$ is a correctly distributed ciphertext for the public key pk and the message $\mathbf{m}$. On the other hand, if $\mathbf{f}$ is uniformly random, then $\mathbf{c}$ is also uniformly random and independent of $\mathbf{m}$. It follows that an adversary with advantage ϵ against the IND-CPA security of packed ElGamal can be used to distinguish DDH with advantage ϵ.

Before presenting our construction we recall a useful pair of algorithms that allow us to compress ciphertexts for the packed ElGamal encryption scheme.

Lemma 15 ([BBD+20]). *There exists a pair of (expected) PPT algorithms* $(\mathsf{Shrink}, \mathsf{ShrinkDec})$ *such that if* $(c, \mathbf{e}) = \mathsf{Enc}(\mathsf{pk}, \mathbf{m})$ *be a packed ElGamal ciphertext encrypting a message* $\mathbf{m} \in \{0,1\}^n$.

- $\mathsf{Shrink}(c, \mathbf{e}) \to (\tilde{c}, K, b_1, \dots, b_n) \in \mathbb{G} \times \{0,1\}^{\lambda + n}$.
- $\Pr[\mathsf{ShrinkDec}(\mathsf{sk}, \mathsf{Shrink}(c, \mathbf{e})) = \mathbf{m}] = 1$.

Proof (Sketch). Let T be a polynomial in the security parameter and let $\mathsf{PRF} : \{0,1\}^\lambda \times \mathbb{G} \to \{0,1\}^\tau$, where $\tau \approx \log(\lambda)$, be a pseudorandom function. On input a ciphertext $(c, (\mathbf{e}_1, \dots, \mathbf{e}_n))$, the compression algorithm Shrink samples the key K for the PRF until the following two conditions are simultaneously satisfied: For all $i \in [n]$ it holds that

(1) $\mathsf{PRF}(K, \mathbf{e}_i / g) \neq 0$.
(2) There exists a $\delta_i \in [T - 1]$ such that $\mathsf{PRF}(K, \mathbf{e}_i \cdot g^{\delta_i}) = 0$.

The compressed ciphertext consists of $(c, K, \delta_1 \mod 2, \dots, \delta_n \mod 2)$ where δ_i is the smallest integer that satisfies condition (2).

The compressed decryption algorithm $\mathsf{ShrinkDec}$ finds, for every $i \in [n]$, the smallest γ_i such that $\mathsf{PRF}(K, c^{\mathbf{s}_i} \cdot g^{\gamma_i}) = 0$ by exhaustive search, where $\mathsf{sk} = (\mathbf{s}_1, \dots, \mathbf{s}_n)$. Finally it outputs $M_i = \delta_i \oplus \mathsf{LSB}(\gamma_i)$, where LSB denotes the least significant bit of an integer. Note that the scheme is correct with probability 1, since condition (1) ensures that there is no ambiguity in the decoding of the bit M_i. By setting the parameters appropriately, we can guarantee that K can always be found in polynomial time, except with negligible probability.

We can straightforwardly modify the algorithm $\mathsf{ShrinkDec}$ in the same way as we have modified the Dec algorithm above to support decryption of ciphertexts produced by Eval_2. Specifically, we modify it such that it takes as an additional input a matrix $\mathbf{M}$ and transforms the secret key sk before decrypting its input ciphertext.

8.6 Construction

We will now provide our construction of rate-1 SSP OT from the packed ElGamal scheme and an additional receiver-secure rate-1 OT. Specifically, let $(\mathsf{OT}_1, \mathsf{OT}_2, \mathsf{OT}_3)$ be a receiver-secure rate-1 OT protocol. We will also use the the packed ElGamal encryption scheme with ciphertext compression discussed above. Finally, let $\mathsf{AR} = (\mathsf{AR.Encode}, \mathsf{AR.Eval}, \mathsf{AR.Decode})$ be an AR-code with linear decoding, i.e. decoding of a codeword $\mathbf{y}$ proceeds by computing $\mathbf{R}_s\mathbf{y}$ for a matrix $\mathbf{R}_s \in \mathbb{Z}_p^{2n \times m}$ specified by a seed s.

Our OT protocol $(\mathsf{OT}_1^*, \mathsf{OT}_2^*, \mathsf{OT}_3^*)$ is given as follows. In the following we assume that the seed s is available to the receiver after the first message ot_1^* has been sent. Note that since the seed can be reused in an arbitrary number of parallel executions of the protocol, its size can be amortized and does therefore not affect the asymptotic rate of the protocol.

$\mathsf{OT}_1^*(1^\lambda, b)$:
- Compute $(ot_1, st_{\mathsf{OT}}) \leftarrow \mathsf{OT}_1(1^\lambda, b)$.
- Compute $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda, m)$.
- Set $\mathbf{A} = b \cdot \mathbf{I} \in \mathbb{Z}_p^{m \times m}$.
- Compute $\mathbf{C} = \mathsf{Enc}(\mathsf{pk}, \mathbf{A})$
- Return $ot_1^* = (ot_1, \mathsf{pk}, \mathbf{C})$ as the first message and set $(st_{\mathsf{OT}}, \mathsf{sk})$ to be the secret state.

$\mathsf{OT}_2^*(ot_1^*, (\mathbf{m}_0, \mathbf{m}_1))$:
- Parse $ot_1^* = (ot_1, \mathsf{pk}, \mathbf{C})$
- Parse $\mathbf{m}_0 \in \{0,1\}^n$ and $\mathbf{m}_1 \in \{0,1\}^n$.
- Sample two uniform vectors $\mathbf{r}_0 \xleftarrow{\$} \mathbb{Z}_p^n$ and $\mathbf{r}_1 \xleftarrow{\$} \mathbb{Z}_p^n$.
- Compute $\mathbf{x}_1 = \begin{pmatrix} \mathbf{m}_1 - \mathbf{m}_0 + \mathbf{r}_0 \\ \mathbf{m}_1 - \mathbf{m}_0 + \mathbf{r}_1 \end{pmatrix} \in \mathbb{Z}_p^{2n}$ and $\mathbf{x}_2 = \begin{pmatrix} \mathbf{m}_0 \\ \mathbf{m}_0 - \mathbf{r}_1 \end{pmatrix} \in \mathbb{Z}_p^{2n}$
- Compute $(\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2) \leftarrow \mathsf{AR.Encode}(s, \mathbf{x}_1, \mathbf{x}_2)$
- Compute $\mathbf{c} = \mathsf{Eval}_1(\mathsf{pk}, \mathbf{C}, \hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2)$
- Compute $\mathbf{c}^* = \mathsf{Eval}_2(\mathsf{pk}, \mathbf{c}, \mathbf{R}_s)$
- Compute $\tilde{\mathbf{c}} = \mathsf{Shrink}(\mathsf{pk}, \mathbf{c}^*)$
- Parse $\tilde{\mathbf{c}}$ as $\tilde{\mathbf{c}} = (\tilde{c}, \tilde{\mathbf{c}}_0, \tilde{\mathbf{c}}_1)$.
- Compute $ot_2 \leftarrow \mathsf{OT}_2(ot_1, (\tilde{\mathbf{c}}_0, \tilde{\mathbf{c}}_1))$.
- Output $ot_2^* \leftarrow (\tilde{c}, ot_2)$

$\mathsf{OT}_3^*(ot_2^*, st)$:
- Parse st as $(st_{\mathsf{OT}}, \mathsf{sk})$ and ot_2^* as $(\tilde{c}, ot_2)$.
- Compute $\tilde{\mathbf{c}}' = \mathsf{OT}_3(ot_2, st_{\mathsf{OT}})$
- Return $\mathsf{ShrinkDec}(\mathsf{sk}, \mathbf{R}_s, (\tilde{c}, \tilde{\mathbf{c}}'))$.

Correctness follows routinely from the correctness of the underlying primitives. First observe that in each of the two branches by homomorphic correctness of Eval_1 and Eval_2 that $\mathbf{c}^*$ encrypts

$$\begin{aligned}\mathbf{R}_s(b \cdot \hat{\mathbf{x}}_1 + \hat{\mathbf{x}}_2) &= \mathsf{AR.Decode}(s, \mathsf{AR.Eval}(b, \mathsf{AR.Encode}(s, \mathbf{x}_1, \mathbf{x}_2))) \\ &= b\mathbf{x}_1 + \mathbf{x}_2.\end{aligned}$$

Now observe that

$$\begin{aligned}\mathsf{ShrinkDec}(\mathsf{sk}, \mathbf{R}_s, (\tilde{c}, \tilde{\mathbf{c}}'))) &= \mathsf{ShrinkDec}(\mathsf{sk}, \mathbf{R}_s, (\tilde{c}, \tilde{\mathbf{c}}_b)) \\ &= \begin{cases}(\mathbf{m}_1 - \mathbf{m}_0 + \mathbf{r}_0) \cdot 0 + \mathbf{m}_0 = \mathbf{m}_0 & \text{if } b = 0 \\ (\mathbf{m}_1 - \mathbf{m}_0 + \mathbf{r}_1) \cdot 1 + \mathbf{m}_0 - \mathbf{r}_1 = \mathbf{m}_1 & \text{if } b = 1\end{cases}\end{aligned}$$

We proceed by showing the computational receiver privacy of the resulting OT protocol.

Theorem 6 (Receiver Privacy). *The scheme as described above is computationally receiver private, given that* $(\mathsf{OT}_1, \mathsf{OT}_2, \mathsf{OT}_3)$ *is a computationally receiver private OT and that the packed ElGamal scheme is IND-CPA secure.*

Proof. The proof consists in observing that the following distributions

$$(\mathsf{OT}_1(1^\lambda, 0), \mathsf{Enc}(\mathsf{pk}, 0 \cdot \mathbf{I})) \approx (\mathsf{OT}_1(1^\lambda, 1), \mathsf{Enc}(\mathsf{pk}, 0 \cdot \mathbf{I})) \approx (\mathsf{OT}_1(1^\lambda, 1), \mathsf{Enc}(\mathsf{pk}, 1 \cdot \mathbf{I}))$$

are computationally indistinguishable by the receiver privacy of the OT and IND-CPA security of the packed ElGamal scheme, respectively.

Instantiating the AR Code Finally, we show that our scheme satisfies statistical sender privacy. In the certified group setting, we can directly rely on the AR codes constructed in Section 6. In the uncertified group setting, we routinely obtain the required codes by concatenating the AR codes of Section 7 over an extension field of $\mathbb{Z}_p$ with the AR codes of Section 6 via Lemma 7.

Theorem 7 (Sender Privacy). *The scheme as described above is statistically sender private, given that the AR code* AR *restricts functions of the form* $h(\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2) = \mathsf{Eval}_1(\mathsf{pk}, \mathbf{C}, \hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2)$ *(for maliciously chosen* pk*,* $\mathbf{C}$*) to linear functions of the form* $f(\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2) = a\hat{\mathbf{x}}_1 + \hat{\mathbf{x}}_2$ *or* $f(\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2) = \hat{\mathbf{x}}_1$.

Proof. We will first provide the description of the (unbounded) simulator Sim. Sim uses the extractor $\mathcal{E}$ and the simulator $\mathcal{S}$ of the AR code AR as a subroutine. Now fix a maliciously chosen receiver message $ot_1^* = (ot_1, \mathsf{pk}, \mathbf{C})$. On input ot_1^*, Sim proceeds as follows. It first defines a function $h(\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2) = \mathsf{Eval}_1(\mathsf{pk}, \mathbf{C}, \hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2)$ and runs the extractor $\mathcal{E}$ on input h, which outputs a function f and auxiliary information aux. The simulator now distinguishes the following two cases.

1. The function f is of the form $f(\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2) = \hat{\mathbf{x}}_1$
2. The function f is of the form $f(\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2) = a \cdot \hat{\mathbf{x}}_1 + \hat{\mathbf{x}}_2$ for an $a \in \mathbb{Z}_p$.

In the first case, the simulator computes $\mathbf{c}$ by running $\mathcal{S}$ and on a uniformly sampled $\mathbf{r} \xleftarrow{\$} \mathbb{Z}_p^{2n}$. In the second case Sim proceeds as follows, where we distinguish 3 sub-cases depending on the value of $a \in \mathbb{Z}_p$.

– $a = 0$: In this case, the simulator queries the OT oracle on 0 to receive $\mathbf{m}_0$, and then computes $\mathbf{c} = \mathcal{S}(\mathsf{aux}, \begin{pmatrix}\mathbf{m}_0 \\ \mathbf{r}_1'\end{pmatrix})$ for a uniformly random $\mathbf{r}_1'$.

– $a = 1$: This case is similar as the previous one, except that the bit is flipped. More precisely, the simulator queries the OT oracle on 1 and receives $\mathbf{m}_1$, then computes $\mathbf{c} = \mathcal{S}(\mathsf{aux}, \begin{pmatrix}\mathbf{r}_0' \\ \mathbf{m}_1\end{pmatrix})$ for a uniformly random $\mathbf{r}_0'$.

– $a \notin \{0, 1\}$: $\mathbf{c}$ is computed running $\mathcal{S}$ and on a uniformly sampled $\mathbf{r} \xleftarrow{\$} \mathbb{Z}_p^{2n}$.

The remainder of the algorithm proceeds exactly as in the definition of OT_2^*. Note that in any of the above cases, the simulator queries the OT oracle at most once. Thus, all we need to argue is that the distribution of the simulated $\mathbf{c}$ is statistically close to the real one.

Our analysis distinguishes the same cases as Sim, i.e. whether the extracted f is of the form $f(\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2) = \hat{\mathbf{x}}_1$ or $f(\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2) = a \cdot \hat{\mathbf{x}}_1 + \hat{\mathbf{x}}_2$.

1. In the first case, when f is of the form $f(\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2) = \hat{\mathbf{x}}_1$, observe that that $\mathbf{x}_1 = \begin{pmatrix} \mathbf{m}_1 - \mathbf{m}_0 + \mathbf{r}_0 \\ \mathbf{m}_1 - \mathbf{m}_0 + \mathbf{r}_1 \end{pmatrix} = \mathbf{r}$ is uniformly random and independent of $\mathbf{m}_0, \mathbf{m}_1$, as $\mathbf{r}_0$ and $\mathbf{r}_1$ are uniformly random. Consequently, it holds by the security of AR that
$$\begin{aligned} (s, \mathbf{c}) &= (s, h(\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2)) \\ &\approx (s, \mathcal{S}(\mathsf{aux}, f(\mathbf{x}_1, \mathbf{x}_2))) \\ &\equiv (s, \mathcal{S}(\mathsf{aux}, \mathbf{x}_1)) \\ &= (s, \mathcal{S}(\mathsf{aux}, \mathbf{r})), \end{aligned}$$
as the information in OT_2^* can be computed from s and $\mathbf{c}$, we conclude that Sim faithfully simulates the sender message ot_2^*.
2. We now turn to analyzing the second case, where f is of the form $f(\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2) = a \cdot \hat{\mathbf{x}}_1 + \hat{\mathbf{x}}_2$. In the first two sub-cases, it holds that $a \in \{0, 1\}$. To see why in these sub-cases the output of the simulator is correctly distributed, first observe that for $a \in \{0, 1\}$ it holds that
$$f(\mathbf{x}_0, \mathbf{x}_1) = a \cdot \begin{pmatrix} \mathbf{m}_1 - \mathbf{m}_0 + \mathbf{r}_0 \\ \mathbf{m}_1 - \mathbf{m}_0 + \mathbf{r}_1 \end{pmatrix} + \begin{pmatrix} \mathbf{m}_0 \\ \mathbf{m}_0 - \mathbf{r}_1 \end{pmatrix} = \begin{cases} \begin{pmatrix} \mathbf{m}_0 \\ \mathbf{m}_0 - \mathbf{r}_1 \end{pmatrix} & \text{if } a = 0 \\ \\ \begin{pmatrix} \mathbf{m}_1 + \mathbf{r}_0 \\ \mathbf{m}_1 \end{pmatrix} & \text{if } a = 1 \end{cases}$$
Note that if $a = 0$, then $\mathbf{r}_1' = \mathbf{m}_0 - \mathbf{r}_1$ is distributed uniformly random. Likewise, if $a = 1$ then $\mathbf{r}_0' = \mathbf{m}_1 + \mathbf{r}_0$ is distributed uniformly random. Consequently, the value $\mathbf{c}$ computed by Sim has the correct distribution as it holds by the security of AR that
$$\begin{aligned} (s, \mathbf{c}) &= (s, h(\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2)) \\ &\approx (s, \mathcal{S}(\mathsf{aux}, f(\mathbf{x}_1, \mathbf{x}_2))). \end{aligned}$$
For the last sub-case ($a \notin \{0, 1\}$) note that
$$f(\mathbf{x}_0, \mathbf{x}_1) = a \cdot \begin{pmatrix} \mathbf{m}_1 - \mathbf{m}_0 + \mathbf{r}_0 \\ \mathbf{m}_1 - \mathbf{m}_0 + \mathbf{r}_1 \end{pmatrix} + \begin{pmatrix} \mathbf{m}_0 \\ \mathbf{m}_0 - \mathbf{r}_1 \end{pmatrix} = \begin{pmatrix} a\mathbf{m}_1 + (1-a)\mathbf{m}_0 \\ a\mathbf{m}_1 + (1-a)\mathbf{m}_0 \end{pmatrix} + \begin{pmatrix} a \cdot \mathbf{r}_0 \\ (1-a) \cdot \mathbf{r}_1 \end{pmatrix}.$$
Note that since $a \notin \{0, 1\}$, the term $\mathbf{r} = \begin{pmatrix} a \cdot \mathbf{r}_0 \\ (1-a) \cdot \mathbf{r}_1 \end{pmatrix}$ is distributed uniformly random. Consequently, it holds by the security of AR that
$$\begin{aligned} (s, \mathbf{c}) &= (s, h(\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2)) \\ &\approx (s, \mathcal{S}(\mathsf{aux}, f(\mathbf{x}_1, \mathbf{x}_2))) \\ &= (s, \mathcal{S}(\mathsf{aux}, \mathbf{r})). \end{aligned}$$
As above, we conclude that Sim faithfully simulates the sender message ot_2^*. This concludes the proof.

Rate-1. Finally we argue that the scheme achieves rate-1. In the calculation we only consider without loss of generality the size of the OT second message (i.e. the download rate), since the size of the first message can always be amortized to increase the rate arbitrarily [DGI+19]. By Lemma 15, we have that for $b \in \{0, 1\}$ it holds that $|\mathsf{Shrink}(\mathsf{pk}, \mathbf{c})| = \log(|\mathbb{G}|) + \lambda + n = 2\lambda + n$, which approaches n as n grows. Since the underlying OT scheme has rate-1, then the size of ot_2 asymptotically equals n.

9 Applications of Rate-1 SSP OT

In this section we show that our rate-1 SSP OT allows us to build PIR with server's statistical security and asymptotically-optimal communication. More generally, by plugging in our rate-1 SSP OT into the construction of [IP07], we obtain homomorphic encryption for branching programs, with (a) statistical branching-program privacy and (b) semi-compactness: the size of a homomorphically-evaluated ciphertext grows only with the depth, and not the size, of the branching program. For brevity, we will present and analyze the protocol for PIR, and the analysis for branching programs will be similar, referring the reader to [IP07].

Recall that PIR allows a client with an index $i \in [K]$ to retrieve the ith element in a database $(m_1, \dots, m_K)$, held by a server. The PIR protocol is given as $(\mathsf{PIR}_1, \mathsf{PIR}_2, \mathsf{PIR}_3)$, where PIR_1 and PIR_3 correspond to the two-phase algorithms of the client, and PIR_2 is run by the sever. We require computational client privacy: for any two indices i_1 and i_2: $\mathsf{pir}_1 \approx \mathsf{pir}_2$, where $(\mathsf{pir}_1, *) \leftarrow_\$ \mathsf{PIR}_1(1^\lambda, i_1)$ and $(\mathsf{pir}_2, *) \leftarrow_\$ \mathsf{PIR}_1(1^\lambda, i_2)$. We require statistical security for the server: there exists a (computationally-unbounded) extraction algorithm PSim such that for any pir_1 and any $\mathbf{m} = (m_1, \dots, m_K)$, the distribution of $(\mathsf{pir}_1, \mathsf{PIR}_2(\mathsf{pir}_1, \mathbf{m}))$ is statistically indistinguishable from $(\mathsf{pir}_1, \mathsf{Sim}^{\mathsf{O}}(\mathsf{pir}_1))$, where Sim has one-time access to an oracle O, which on an input index i returns m_i.

In the following we let $K = 2^k$. The following construction is from [IP07].

- $\mathsf{PIR}_1(1^\lambda, s \in [2^k])$: Parse $s = s_1 \dots s_k$. Return $\mathsf{pir}_1 := (\mathsf{otr}_1, \dots, \mathsf{otr}_k)$ and $\mathsf{st} := (\mathsf{stt}_1, \dots, \mathsf{stt}_k)$, where for $i \in [K]$: $(\mathsf{otr}_i, \mathsf{stt}_i) \leftarrow_\$ \mathsf{OT}_1(1^\lambda, s_i)$.
- $\mathsf{PIR}_2(\mathsf{pir}_1, (m_1, \dots, m_K))$: Parse $\mathsf{pir}_1 := (\mathsf{otr}_1, \dots, \mathsf{otr}_k)$. For $i \in [2^k]$, set $\mathsf{ots}_i^{(0)} := m_i$. For $w \in \{1, \dots, k\}$:
  1. For $i \in [2^{k-w}]$, let $\mathsf{ots}_i^{(w)} \leftarrow_\$ \mathsf{OT}_2(\mathsf{otr}_w, (\mathsf{ots}_{2i-1}^{(w-1)}, \mathsf{ots}_{2i}^{(w-1)}))$

  Return $\mathsf{ots}_1^{(k)}$.
- $\mathsf{PIR}_3(\mathsf{st}, \mathsf{pir}_2)$: Parse $\mathsf{st} := (\mathsf{stt}_1, \dots, \mathsf{stt}_k)$ and $\mathsf{pir}_2 := \mathsf{ots}^{(0)}$. For $i \in [k]$ let $\mathsf{ots}^{(i)} := \mathsf{OT}_1(\mathsf{stt}_{k-i+1}, \mathsf{ots}^{(i-1)})$. Return $\mathsf{ots}^{(k)}$.

Correctness and client's security follow immediately. To establish statistical server's security we define the following computationally-unbounded simulation algorithm PSim, built based on the OT's simulation algorithm OSim.

- $\mathsf{PSim}^{\mathsf{O}}(\mathsf{otr}_1, \dots, \mathsf{otr}_\mathsf{k})$: For $i \in [k]$ extract a bit b_i from otr_i using the OT's simulation algorithm OSim (by observing the bit query that $\mathsf{OSim}(\mathsf{otr}_\mathsf{i})$ makes). Let $s = b_k \cdots b_1$, and query $O(s)$ to get m. Let $\mathsf{ots}_s^{(0)} = m$, and for all $i \in [K] \setminus \{s\}$, set $\mathsf{ots}_i^{(0)}$ to an arbitrary value. Continue the procedure of PIR_2 described above based on these values.

The proof of statistical security for PSim follows from that of OSim using an inductive argument. We omit the details.

Server's communication complexity. For $r = r(\lambda)$, if the server has $K = 2^k$ messages each of r bits, then the server's communication complexity is $r + \mathsf{poly}(\mathsf{k}, \lambda)$ bits (where poly is a fixed polynomial), achieving rate-1 for large enough r.

References

ACLV19. Divesh Aggarwal, Kai-Min Chung, Han-Hsuan Lin, and Thomas Vidick. A quantum-proof non-malleable extractor - with application to privacy amplification against active quantum adversaries. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2019, Part II*, volume 11477 of *Lecture Notes in Computer Science*, pages 442–469, Darmstadt, Germany, May 19–23, 2019. Springer, Heidelberg, Germany. 17

ADL14. Divesh Aggarwal, Yevgeniy Dodis, and Shachar Lovett. Non-malleable codes from additive combinatorics. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 774–783, 2014. 1, 11, 12

AIR01. William Aiello, Yuval Ishai, and Omer Reingold. Priced oblivious transfer: How to sell digital goods. In Birgit Pfitzmann, editor, *Advances in Cryptology – EUROCRYPT 2001*, volume 2045 of *Lecture Notes in Computer Science*, pages 119–135, Innsbruck, Austria, May 6–10, 2001. Springer, Heidelberg, Germany. 5, 9

AO20. Divesh Aggarwal and Maciej Obremski. A constant rate non-malleable code in the split-state model. In *61st Annual Symposium on Foundations of Computer Science*, pages 1285–1294, Durham, NC, USA, November 16–19, 2020. IEEE Computer Society Press. 11

BBD$^+$20. Zvika Brakerski, Pedro Branco, Nico Döttling, Sanjam Garg, and Giulio Malavolta. Constant ciphertext-rate non-committing encryption from standard assumptions. In Rafael Pass and Krzysztof Pietrzak, editors, *TCC 2020: 18th Theory of Cryptography Conference, Part I*, volume 12550 of *Lecture Notes in Computer Science*, pages 58–87, Durham, NC, USA, November 16–19, 2020. Springer, Heidelberg, Germany. 27

BD18. Zvika Brakerski and Nico Döttling. Two-message statistically sender-private OT from LWE. In Amos Beimel and Stefan Dziembowski, editors, *TCC 2018: 16th Theory of Cryptography Conference, Part II*, volume 11240 of *Lecture Notes in Computer Science*, pages 370–390, Panaji, India, November 11–14, 2018. Springer, Heidelberg, Germany. 9

BDGM19. Zvika Brakerski, Nico Döttling, Sanjam Garg, and Giulio Malavolta. Leveraging linear decryption: Rate-1 fully-homomorphic encryption and time-lock puzzles. In Dennis Hofheinz and Alon Rosen, editors, *TCC 2019: 17th Theory of Cryptography Conference, Part II*, volume 11892 of *Lecture Notes in Computer Science*, pages 407–437, Nuremberg, Germany, December 1–5, 2019. Springer, Heidelberg, Germany. 9

BFJ$^+$20. Saikrishna Badrinarayanan, Rex Fernando, Aayush Jain, Dakshita Khurana, and Amit Sahai. Statistical ZAP arguments. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology – EUROCRYPT 2020, Part III*, volume 12107 of *Lecture Notes in Computer Science*, pages 642–667, Zagreb, Croatia, May 10–14, 2020. Springer, Heidelberg, Germany. 5

BGI$^+$17. Saikrishna Badrinarayanan, Sanjam Garg, Yuval Ishai, Amit Sahai, and Akshay Wadia. Two-message witness indistinguishability and secure computation in the plain model from new assumptions. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology – ASIACRYPT 2017, Part III*, volume 10626 of *Lecture Notes in Computer Science*, pages 275–303, Hong Kong, China, December 3–7, 2017. Springer, Heidelberg, Germany. 9

BV11. Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. In Rafail Ostrovsky, editor, *52nd Annual Symposium on Foundations of Computer Science*, pages 97–106, Palm Springs, CA, USA, October 22–25, 2011. IEEE Computer Society Press. 2

CG14. Mahdi Cheraghchi and Venkatesan Guruswami. Non-malleable coding against bit-wise and split-state tampering. In Yehuda Lindell, editor, *TCC 2014: 11th Theory of Cryptography Conference*, volume 8349 of *Lecture Notes in Computer Science*, pages 440–464, San Diego, CA, USA, February 24–26, 2014. Springer, Heidelberg, Germany. 1

DGI$^+$19. Nico Döttling, Sanjam Garg, Yuval Ishai, Giulio Malavolta, Tamer Mour, and Rafail Ostrovsky. Trapdoor hash functions and their applications. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019, Part III*, volume 11694 of *Lecture Notes in Computer Science*, pages 3–32, Santa Barbara, CA, USA, August 18–22, 2019. Springer, Heidelberg, Germany. 6, 8, 30

DH76. Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976. 24

DKO13. Stefan Dziembowski, Tomasz Kazana, and Maciej Obremski. Non-malleable codes from two-source extractors. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part II*, volume 8043 of *Lecture Notes in Computer Science*, pages 239–257, Santa Barbara, CA, USA, August 18–22, 2013. Springer, Heidelberg, Germany. 1, 11

DLWZ11. Yevgeniy Dodis, Xin Li, Trevor D. Wooley, and David Zuckerman. Privacy amplification and non-malleable extractors via character sums. In Rafail Ostrovsky, editor, *52nd Annual Symposium on Foundations of Computer Science*, pages 668–677, Palm Springs, CA, USA, October 22–25, 2011. IEEE Computer Society Press. 1

DPW10. Stefan Dziembowski, Krzysztof Pietrzak, and Daniel Wichs. Non-malleable codes. In Andrew Chi-Chih Yao, editor, *ICS 2010: 1st Innovations in Computer Science*, pages 434–452, Tsinghua University, Beijing, China, January 5–7, 2010. Tsinghua University Press. 1

DW09. Yevgeniy Dodis and Daniel Wichs. Non-malleable extractors and symmetric key cryptography from weak secrets. In Michael Mitzenmacher, editor, *41st Annual ACM Symposium on Theory of Computing*, pages 601–610, Bethesda, MD, USA, May 31 – June 2, 2009. ACM Press. 1

EGL82. 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 – CRYPTO'82*, pages 205–210, Santa Barbara, CA, USA, 1982. Plenum Press, New York, USA. 5

ElG84. Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In G. R. Blakley and David Chaum, editors, *Advances in Cryptology – CRYPTO'84*, volume 196 of *Lecture Notes in Computer Science*, pages 10–18, Santa Barbara, CA, USA, August 19–23, 1984. Springer, Heidelberg, Germany. 6, 26

Gen09. Craig Gentry. Fully homomorphic encryption using ideal lattices. In Michael Mitzenmacher, editor, *41st Annual ACM Symposium on Theory of Computing*, pages 169–178, Bethesda, MD, USA, May 31 – June 2, 2009. ACM Press. 2

GH19. Craig Gentry and Shai Halevi. Compressible FHE with applications to PIR. In Dennis Hofheinz and Alon Rosen, editors, *TCC 2019: 17th Theory of Cryptography Conference, Part II*, volume 11892 of *Lecture Notes in Computer Science*, pages 438–464, Nuremberg, Germany, December 1–5, 2019. Springer, Heidelberg, Germany. 9

GJJM20. Vipul Goyal, Abhishek Jain, Zhengzhong Jin, and Giulio Malavolta. Statistical zaps and new oblivious transfer protocols. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology – EUROCRYPT 2020, Part III*, volume 12107 of *Lecture Notes in Computer Science*, pages 668–699, Zagreb, Croatia, May 10–14, 2020. Springer, Heidelberg, Germany. 5

GO94. Oded Goldreich and Yair Oren. Definitions and properties of zero-knowledge proof systems. *J. Cryptol.*, 7(1):1–32, 1994. 5

GSW13. Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part I*, volume 8042 of *Lecture Notes in Computer Science*, pages 75–92, Santa Barbara, CA, USA, August 18–22, 2013. Springer, Heidelberg, Germany. 2

HK12. Shai Halevi and Yael Tauman Kalai. Smooth projective hashing and two-message oblivious transfer. *Journal of Cryptology*, 25(1):158–193, January 2012. 9

IP07. Yuval Ishai and Anat Paskin. Evaluating branching programs on encrypted data. In Salil P. Vadhan, editor, *TCC 2007: 4th Theory of Cryptography Conference*, volume 4392 of *Lecture Notes in Computer Science*, pages 575–594, Amsterdam, The Netherlands, February 21–24, 2007. Springer, Heidelberg, Germany. 6, 31

Kil88. Joe Kilian. Founding cryptography on oblivious transfer. In *20th Annual ACM Symposium on Theory of Computing*, pages 20–31, Chicago, IL, USA, May 2–4, 1988. ACM Press. 5

KO97. Eyal Kushilevitz and Rafail Ostrovsky. Replication is NOT needed: SINGLE database, computationally-private information retrieval. In *38th Annual Symposium on Foundations of Computer Science*, pages 364–373, Miami Beach, Florida, October 19–22, 1997. IEEE Computer Society Press. 6

Li12. Xin Li. Non-malleable extractors, two-source extractors and privacy amplification. In *53rd Annual Symposium on Foundations of Computer Science*, pages 688–697, New Brunswick, NJ, USA, October 20–23, 2012. IEEE Computer Society Press. 1

NP01. Moni Naor and Benny Pinkas. Efficient oblivious transfer protocols. In S. Rao Kosaraju, editor, *12th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 448–457, Washington, DC, USA, January 7–9, 2001. ACM-SIAM. 5, 9

OPP14. Rafail Ostrovsky, Anat Paskin-Cherniavsky, and Beni Paskin-Cherniavsky. Maliciously circuit-private FHE. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology – CRYPTO 2014, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 536–553, Santa Barbara, CA, USA, August 17–21, 2014. Springer, Heidelberg, Germany. 5

Pai99. Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In Jacques Stern, editor, *Advances in Cryptology – EUROCRYPT'99*, volume 1592 of *Lecture Notes in Computer Science*, pages 223–238, Prague, Czech Republic, May 2–6, 1999. Springer, Heidelberg, Germany. 2

Rab05. Michael O. Rabin. How to exchange secrets with oblivious transfer. Cryptology ePrint Archive, Report 2005/187, 2005. `http://eprint.iacr.org/2005/187`. 5

Vaz86. Umesh Vazirani. *Randomness, Adversaries and Computation.* PhD thesis, EECS, UC Berkeley, 1986. Ph.D. Thesis. 5

vLW01. Jacobus Hendricus van Lint and Richard Michael Wilson. *A Course in Combinatorics.* Cambridge University Press, Cambridge, U.K.; New York, 2001. 11

Yao82. Andrew Chi-Chih Yao. Theory and applications of trapdoor functions (extended abstract). In *23rd Annual Symposium on Foundations of Computer Science*, pages 80–91, Chicago, Illinois, November 3–5, 1982. IEEE Computer Society Press. 2

Yao86. Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *27th Annual Symposium on Foundations of Computer Science*, pages 162–167, Toronto, Ontario, Canada, October 27–29, 1986. IEEE Computer Society Press. 5