Batched Oblivious Transfer with Square-Root Communication

Yicheng Li[1], Claudio Orlandi[2], Lawrence Roy[3] and Yizhou Yao[4]

[1] Arizona State University, USA
[2] Aarhus University, Denmark
[3] IBM Research Zürich, Switzerland
[4] Shanghai Jiao Tong University, China

Abstract. Oblivious Transfer (OT) is a fundamental cryptographic primitive and a core building block for many multiparty cryptographic protocols. While existing OT extension techniques achieve excellent asymptotic efficiency for very large batches, their performance degrades when the total number of OTs is only moderate, since the cost of generating the required base OTs is no longer effectively amortized. In this work, we close this gap by presenting OT constructions that achieve square-root communication complexity for batched OT generation. Concretely, our most efficient protocol generates ℓ random OTs using $O(\lambda\sqrt{\ell})$ bits of communication, where λ is the security parameter.

Our constructions are inspired by recent advances in homomorphic secret sharing and techniques for distributed discrete logarithm computation, and explore complementary points in the design space. The first construction is based on the Damgård–Jurik cryptosystem and standard assumptions, at the cost of a one-time trusted setup. The second eliminates the need for trusted setup, relying instead on a power-DDH assumption over prime-order groups. For 128-bit security, our schemes require approximately 2.6 KiB and 1.25 KiB of communication, respectively, to generate 128 random OTs, including the hash key in the power-DDH construction. Our best construction outperforms existing OT extension techniques for batch sizes up to $\ell \leq 2^{15}$.

Keywords: Oblivious Transfer · Vector Oblivious Linear Evaluation

E-mail: lycxsh@gmail.com (Yicheng Li), orlandi@cs.au.dk (Claudio Orlandi), ldr709@gmail.com (Lawrence Roy), yaoyizhou0620@sjtu.edu.cn (Yizhou Yao)

This work is licensed under a "CC BY 4.0" license.
Date of this document: 2026-10-12.

Contents

1 Introduction **2**
1.1 Our Contributions . . . 3
1.2 Related Work . . . 4

2 Technical Overview **5**
2.1 From VOLE to Oblivious Transfer . . . 5
2.2 DDLog-Based VOLE Constructions . . . 6
2.3 CRT Packing with Damgård–Jurik . . . 8
2.4 Toeplitz Compression with Power-DDH . . . 9

3 Preliminaries **11**
3.1 Correlation Robustness . . . 11
3.2 Damgård–Jurik and Distributed Logarithms . . . 12

4 Half-Chosen OT Constructions **13**
4.1 OT from Power-DDH . . . 13
4.2 OT from the Damgård–Jurik cryptosystem . . . 17

5 Efficiency **22**

References **24**

1 Introduction

Oblivious Transfer (OT) is one of the most fundamental primitives in cryptography. In an OT protocol, a sender holds two strings $x_0, x_1 \in \{0,1\}^\lambda$, and a receiver learns one of the strings x_b without revealing the choice bit $b \in \{0,1\}$ to the sender. It is well known that OT implies public-key cryptography, and thus performing even a single OT is relatively expensive. Moreover, many interesting cryptographic applications (such as secure multiparty computation, private set-intersection, and more) require a large number of OT correlations. Thus, a long line of work has investigated and improved the cost of generating OT correlations in an amortized sense.

OT Extension (OTE) is one of the most successful approaches for reducing this cost. In an OTE protocol, a small number of OTs (say, n base OTs) are used to efficiently generate a much larger number of random OT correlations (say, ℓ extended OTs). These random OTs can then be transformed into chosen-input OTs using standard information-theoretic reductions, at the cost of transferring $(2\lambda+1)\ell$ additional bits.

Significant progress has been made in optimizing OT extension for large values of ℓ. Using only Minicrypt assumptions, SoftspokenOT [Roy22] shows that, starting from λ base OTs, an arbitrary number of extended OTs can be generated with an additional λ/k bits of communication and $O(2^k/k)$ computation per OT, for an integer parameter $2 \le k \le 10$. Under variants of the Learning Parity with Noise (LPN) assumption, the pseudorandom correlation generator (PCG) framework [BCG+19b, BCG+22] achieves an arbitrary polynomial stretch. Concretely, to generate ℓ extended OTs, the PCG-based protocol starts from $O\big(\lambda \log(\ell/\lambda)\big)$ base OTs, requires $O\big(\lambda^2 \log(\ell/\lambda)\big)$ bits of communication, and runs in nearly linear time in ℓ[1]. For example, generating 2^{23} extended OTs

[1]The running time is dominated by computing an LPN instance and $O(\ell)$ PRG calls; several works [BCG+22, RRT23] propose LPN variants based on linear codes with fast encoding, yielding an almost linear-time PCG.

with $\lambda = 128$ requires about 32 MiB of communication using SoftSpokenOT, whereas state-of-the-art PCG constructions [BCG+22, GYW+23] need only about 31 KiB.

Despite these strong results at scale, a substantial efficiency gap remains when the number of OTs is of modest size, for example $\ell \leq 2^{11}$. This regime is important both for practical applications and as an intermediate step in OT extension pipelines, where a moderate number of OTs are used as base OTs for further amplification. For instance, generating 2^{20} extended OTs using PCGs uses 2^{10} intermediate OTs. However, generating this many base OTs using SoftSpokenOT [Roy22] already incurs about 11 KiB of communication[2]. Since roughly 22 KiB of communication already suffices for PCGs [BCG+22] to generate 2^{16} random OTs, the cost of base OT generation becomes a dominant bottleneck. Consequently, while PCG-based approaches achieve near-optimal performance for very large ℓ, they offer limited savings in the small and medium-scale regime. See Figure 1 for a concrete comparison of state-of-the-art approaches.

Our motivation: In this work, we focus on the problem of efficiently generating a *moderate* number of random OTs, in a regime where the total number of OTs is large enough to matter in practice, but too small for existing OT extension techniques to effectively amortize the cost of base OTs (e.g., thousands to millions). This regime arises both in real-world applications and as an intermediate step in larger OT extension pipelines. Despite excellent asymptotic performance at very large scales, the state of the art leaves an efficiency gap in this setting.

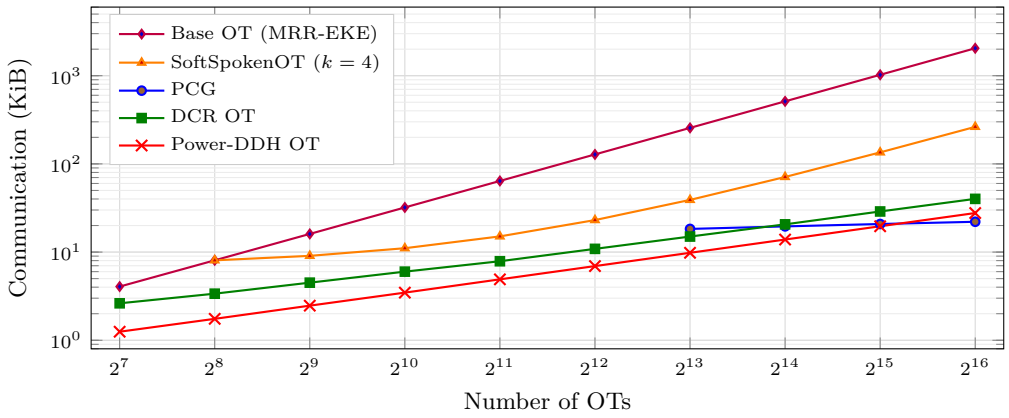


Figure 1: Communication in KiB for generating $\ell \in \{2^7, 2^8, \ldots, 2^{16}\}$ OTs, using the accounting in Table 1. The base-OT baseline uses the EKE/Even–Mansour variant of [MRR21]; these base OTs are also used by SoftSpokenOT ($k = 4$) [Roy22] and PCG [BCG+22, GYW+23]. Power-DDH costs include the initial hash key.

1.1 Our Contributions

We present two new protocols for generating random OTs with *sublinear* communication complexity. Specifically, our most efficient protocol generates ℓ random OTs using $O(\lambda\sqrt{\ell})$ communication, improving (in concrete terms) over the state of the art in the regime where the number of OTs is moderate. For typical parameters, our schemes achieve significantly smaller concrete communication costs than existing OT extension techniques.

At a high level, the two constructions explore complementary points in the design space. The first construction relies only on standard cryptographic assumptions, at the cost of a

[2]PCG-based constructions are typically instantiated with at least 2^{12} random OTs, as using smaller parameters may make the underlying LPN assumption insecure.

one-time trusted setup for generating a Damgård–Jurik modulus. The second construction removes the need for trusted setup, but assumes a stronger power-DDH assumption over prime-order groups. Together, they show that sublinear-communication OT generation can be achieved under different trust and assumption models.

Random OT from Damgård–Jurik. Let λ denote the security parameter, and let N be a secure RSA modulus used in the Damgård–Jurik cryptosystem.

Theorem 1 (informal)**.** *Assuming a secure correlation-robust hash function (Definition 1), a secure pseudorandom function, and the decisional composite residuosity (DCR) assumption, there exists a semi-honest secure protocol that generates ℓ random OTs with $O(\sqrt{\ell\lambda \log N})$ communication.*

This construction requires a one-time trusted setup to generate the modulus N, such that no party learns the corresponding secret key. After setup, the protocol consists of a single simultaneous round in which the parties exchange messages, making it well suited for asynchronous network settings.

Random OT from the Power-DDH assumption. Inspired by recent work on sublinear-communication vector oblivious linear evaluation (VOLE) [ARS24], we also present an OT construction based on a power-DDH assumption. Unlike prior work, which relies on Paillier groups or class groups to support distributed decryption, our construction only requires power-DDH over prime-order groups.

Theorem 2 (informal)**.** *Assuming a secure correlation-robust hash function (Definition 1) and the power-DDH assumption (Definition 3) over prime-order groups, there exists a semi-honest secure protocol that generates ℓ random OTs with $O(\lambda\sqrt{\ell})$ communication.*

Although we focus on random OT, both constructions in fact realize a stronger notion of *half-chosen OT*, where the sender's inputs are random while the receiver may choose its selection bits. This property allows our DCR and power-DDH protocols to be directly compiled into two-message and three-message standard OT, respectively; the latter also takes only two messages once the sender's hash key has been distributed.

From a quantitative perspective, our constructions offer improved concrete efficiency for a wide range of parameters. In particular, the DJ and power-DDH schemes achieve lower communication than the OT extension and base-OT baselines through batch sizes 2^{13} and 2^{15}, respectively. For example, with $\lambda = 128$, the Damgård–Jurik based construction requires approximately 2.6 KiB of communication to generate 128 random OTs, while the power-DDH based construction reduces this cost to 1.25 KiB, including transmission of the hash key. Concrete comparisons with prior work are given in Table 1, and a detailed efficiency analysis is provided in Section 5. As this paper focuses on realizing OT with concretely less communication, we leave malicious security as future work. We would like to remark that a major difficulty in extending to malicious security lies in extraction. Due to the succinctness of communication, it is non-trivial to design extraction procedures that work for arbitrary malformed messages. In principle one could enforce well-formedness of the protocol messages by requiring the parties to accompany them with sufficiently expressive succinct zero-knowledge proofs. However this might incur a significant overhead in concrete communication, in particular when generating a moderate number of OTs.

1.2 Related Work

OT extension protocols have been extensively studied since the seminal work of Beaver [Bea96]. IKNP [IKNP03] gave the practical construction of OT extension and has become the inspiration for most subsequent OT extension protocols. IKNP allows one to extend λ random OTs to an arbitrary polynomial number of random OTs with λ bits of communication per OT, assuming a correlation-robust hash function. A follow-up

work [KOS15] further achieves malicious security for IKNP with a small overhead. There is also a line of works [KK13, OOS17, PSS17] that focuses on extending base 1-out-of-2 OTs to many 1-out-of-n OTs with malicious security. Recently, SoftSpokenOT [Roy22] optimizes IKNP's per-OT communication to λ/k, at the cost of increasing computation by $2^{k-1}/k$, where k is an arbitrary integer.

In the past few years, the focus of research has shifted to building OT extension protocols from assumptions beyond Minicrypt. Pseudorandom correlation generators (PCGs) [BCG+19a, BCG+19b] allow one to extend $\lambda \log N$ OTs to N OTs with $O(\lambda^2 \log N)$ bits of communication for seed distribution, followed by silent local expansion, assuming the hardness of syndrome decoding problems. Here, "silent" means that OTs can be locally expanded from a short seed generated in a setup phase, with communication nearly independent of the number of extended OTs. The seed distribution can either be done by a trusted dealer, or by both parties engaging in an interactive protocol. As their efficiency bottleneck typically lies in computation, follow-up works [BCG+22, RRT23] instantiate PCGs with various linear codes that admit fast encoding or syndrome computation algorithms. Notably, by combining PCGs with local PRGs, [BCG+23] achieves constant computational overhead. Since both IKNP-style protocols and PCGs essentially need random base OTs, they inherently inherit the setup of the underlying OT protocol.

Pseudorandom correlation functions (PCFs) [BCG+20] generalize the notion of PCGs by allowing the generation of an unbounded number of random OTs. To date, PCFs for OT have been constructed under various assumptions, such as variable-density LPN [BCG+20], syndrome decoding of expand–accumulate codes [BCG+22], sparse LPN [BCM+25], and ring-LWR [HRK25]. To eliminate the need for an expensive setup for seed distribution, [OSY21] proposed the notion of *public-key* PCFs and gave constructions from QR/DCR. Nevertheless, a trusted public-key setup is required. Subsequent works [BCM+24, CDD+24] further developed an efficient framework based on constrained PRFs, and reported a throughput of 500,000 OTs per second [CDD+24], which is one and two orders of magnitude slower than PCG [RRT23] and SoftSpokenOT [Roy22], respectively.

2 Technical Overview

At a high level, both of our constructions follow a common template that has emerged in work on efficient oblivious transfer [IKNP03, BCG+19b] and related primitives. The starting point is to view random OT generation as a special case of producing correlated values, such as vector oblivious linear evaluations (VOLE), which can then be locally transformed into OT correlations using correlation-robust hashing. Achieving sublinear communication therefore reduces to designing succinct protocols for generating such correlated values. Our work builds on and refines techniques from prior constructions based on distributed discrete logarithms and homomorphic secret sharing [BGI16, OSY21, RS21, ARS24], while introducing new ways to pack and compress correlations so as to reduce communication in the regime of moderate batch sizes. In the following, we first recall the standard reduction from VOLE to OT, then review prior DDLog-based approaches, and finally describe our two protocols and the ideas that enable square-root communication.

2.1 From VOLE to Oblivious Transfer

A standard approach to generating random oblivious transfers is to first generate a related form of correlated randomness and then locally transform it into OT correlations. One particularly useful primitive for this purpose is *subfield vector oblivious linear evaluation* (VOLE). At a high level, subfield VOLE provides the parties with vectors that are linearly related by a secret offset. Each index of this correlation can be interpreted as a correlated OT instance, where all instances share the same offset. By applying a correlation-robust

hash function locally, this shared correlation can be removed, yielding independent random OTs. As a result, the problem of generating random OTs reduces to the problem of efficiently generating subfield VOLEs, a notion that we recall here: Party P_0 holds a vector $\mathbf{b} \in \mathbb{F}_2^\ell$ and a vector $\mathbf{z}_0 \in \mathbb{F}_{2^k}^\ell$, while party P_1 holds a scalar $\Delta \in \mathbb{F}_{2^k}$ and a vector $\mathbf{z}_1 \in \mathbb{F}_{2^k}^\ell$, such that

$$\mathbf{z}_0 = \mathbf{b} \cdot \Delta + \mathbf{z}_1.$$

Equivalently, for each index $j \in [\ell]$, party P_1 defines a pair $(\mathbf{z}_1[j], \mathbf{z}_1[j] \oplus \Delta)$, and party P_0 learns one of the two values according to the bit $\mathbf{b}[j]$, since

$$\mathbf{z}_0[j] = \mathbf{z}_1[j] \oplus (\mathbf{b}[j] \cdot \Delta).$$

This corresponds to a batch of correlated OTs whose sender inputs differ by a common Δ.

To obtain standard random OT correlations, the parties locally apply a correlation-robust hash function $H : \{0,1\}^* \to \{0,1\}^\lambda$. After hashing, each tuple

$$\big(\mathbf{b}[j], H(\mathbf{z}_0[j])\big) \quad \text{and} \quad \big(H(\mathbf{z}_1[j]), H(\mathbf{z}_1[j] \oplus \Delta)\big)$$

is computationally indistinguishable from an independently random OT instance, where P_0 holds (b, m_b) and P_1 holds (m_0, m_1). Consequently, generating ℓ random OTs reduces to generating a subfield VOLE of length ℓ.

Asymptotic efficiency. This reduction was introduced in [IKNP03], and has been widely used in OT extension. In particular, the LPN-based framework of [BCG+19a, BCG+19b] constructs subfield VOLEs with total communication complexity $O\big(\lambda^2 \log(\ell/\lambda)\big)$, which directly translates into the same asymptotic communication for generating ℓ random OTs via the above transformation. For sufficiently large ℓ, this yields extremely small amortized communication per OT, even below 1 bit per OT.

Concrete efficiency and limitations. Despite their excellent asymptotic behavior, these OT extension constructions incur a communication cost of at least λ^2, which dominates for small or moderate values of ℓ. Moreover, since the LPN parameters and the number of generated OTs ℓ are inherently linked, instantiating these protocols for fewer than 2^{12} OTs leads to LPN instances that no longer achieve 128-bit security, and is therefore considered deprecated. This motivates the search for alternative VOLE constructions with sublinear communication and better concrete efficiency in the moderate-scale regime, which is the focus of this work.

2.2 DDLog-Based VOLE Constructions

A modern approach to constructing succinct VOLE protocols is based on *distributed discrete logarithm* (DDLog) techniques. At a high level, DDLog allows two parties holding multiplicative shares of a group element to locally derive additive shares of the corresponding exponent. This enables the conversion of multiplicative correlations into linear correlations, which can then be used to construct VOLE. These techniques also apply more broadly to the succinct generation of general correlations using homomorphic secret sharing [BGI16, BCG+17], and, in particular, to succinct homomorphic secret sharing [ARS24].

We briefly recall the negligible-error DDLog mechanism for Paillier groups [OSY21, RS21, ADOS22]. Consider the Paillier group $\mathbb{Z}_{N^2}^*$, where $N = pq$ is an RSA modulus. Given group elements $g_0, g_1 \in \mathbb{Z}_{N^2}^*$ held by parties P_0 and P_1, respectively, such that

$$g_0 = (1+N)^{\Delta \cdot x} \cdot g_1 \bmod N^2,$$

the DDLog procedure allows the parties to locally compute values

$$z_0 \leftarrow \mathsf{DDLog}(g_0), \qquad z_1 \leftarrow \mathsf{DDLog}(g_1),$$

satisfying

$$z_0 = \Delta \cdot x + z_1 \pmod N.$$

That is, the multiplicative relation between g_0 and g_1 is converted into an additive sharing of the product $\Delta \cdot x$. Then a standard modulus switching / lifting trick [BKS19, OSY21] lets you remove the $\pmod N$ and get shares over the integers

$$z_0' = \Delta \cdot x + z_1',$$

which are correct except with probability $\frac{|\Delta \cdot x|}{N}$.

Using this primitive, [ARS24] shows how to compute many such products efficiently by reducing the computation of a vector–scalar product to a matrix multiplication. Concretely, a vector $\mathbf{x}$ of length $\ell = mn$ is reshaped into a matrix $X \in \{0,1\}^{m \times n}$, and the scalar multiplication $\mathbf{x} \cdot \Delta$ is expressed as the matrix product XY, where $Y = \Delta \cdot I_n$. Party P_0 encodes the rows of X, while party P_1 encodes the columns of Y, allowing the parties to derive additive shares of each entry $(XY)_{i,j}$ via DDLog.

Specifically, letting $g_0, g_1, \ldots, g_n \in \mathbb{Z}_{N^2}^*$ be public group elements, P_0 sends:

$$d_i := g_0^{v_i} \cdot \prod_{k=1}^{n} g_k^{x_{i,k}} \text{ for each row } i \in [m]$$

where v_i is a random mask, while P_1 sends for each column $j \in [n]$

$$c_j := \left(g_0^{r_j},\, g_1^{r_j} \cdot (1+N)^{y_{1,j}},\, \ldots,\, g_n^{r_j} \cdot (1+N)^{y_{n,j}}\right),$$

with r_j sampled at random. The parties can then locally compute multiplicative shares $h_{0,i,j}$ and $h_{1,i,j}$ of

$$\frac{h_{0,i,j}}{h_{1,i,j}} = \frac{c_{j,0}^{v_i} \cdot \prod_{k=1}^{n} c_{j,k}^{x_{i,k}}}{d_i^{r_j}} = \frac{(g_0^{r_j})^{v_i} \cdot \prod_{k=1}^{n} (g_k^{r_j} \cdot (1+N)^{y_{k,j}})^{x_{i,k}}}{(g_0^{v_i} \cdot \prod_{k=1}^{n} g_k^{x_{i,k}})^{r_j}}$$
$$= (1+N)^{\sum_{k=1}^{n} x_{i,k} y_{k,j}} = (1+N)^{(XY)_{i,j}},$$

so additive shares of $(XY)_{i,j}$ can be recovered from $h_{0,i,j}$ and $h_{1,i,j}$ using DDLog.

Asymptotic efficiency. Instantiating this approach with $m = \ell^{2/3}$ and $n = \ell^{1/3}$ yields total communication $O(\ell^{2/3} \cdot \log N)$ bits for generating a VOLE of length ℓ. Although this approach realizes a stronger notion of half-chosen VOLE that allows the parties to choose $(\Delta, \mathbf{x})$, essentially random VOLE suffices for OT and moreover the communication complexity is considerably larger than $O(\lambda^2 \log \ell / \lambda)$ of [BCG+19b].

Concrete efficiency and limitations. Despite their strong asymptotic performance, DDLog-based constructions incur large concrete overheads. The use of Paillier groups requires large moduli, and the communication includes $\Theta(n^2)$ group elements of size $\widetilde{\Theta}(\lambda^3)$ bits (to be secure against the general number field sieve [LJMP90]). For typical parameters, the communication per OT exceeds 128 bits for $\ell \leq 2^{18}$, offering no advantage over linear-communication OT extension schemes [MRR21, Roy22].[3]

[3]Note that [ARS24] also proposes an optimization that reduces the communication to $2^{13} \cdot \ell^{1/2}$ bits for P_0, by introducing n-ary power-DDH assumptions in Paillier groups. While this improves asymptotic efficiency, the concrete communication per OT remains above 128 bits for $\ell \leq 2^{12}$, and is therefore still unsuitable for generating a small or moderate number of OTs.

2.3 CRT Packing with Damgård–Jurik

We now describe our first construction for generating subfield VOLEs with sublinear communication, which is based on the Damgård–Jurik cryptosystem.

The starting point is the observation that the main bottleneck in prior DDLog-based constructions is the use of large Paillier moduli. While DDLog can also be instantiated over prime-order groups [BGI16, BGI17], such instantiations incur an inverse-polynomial correctness error, which is undesirable.

Packing via CRT and Damgård–Jurik. Our first idea is to amortize communication by packing multiple VOLE instances using the Chinese Remainder Theorem (CRT). Concretely, we aim to generate VOLE correlations over several prime moduli $p_1, \ldots, p_n$, each slightly larger than 2^λ, and then combine them via CRT. Since our ultimate goal is to obtain OT correlations, it is not necessary for all VOLE instances to share the same modulus; it suffices that the moduli have comparable bit length. However, correctness requires that N satisfies

$$N > \prod_{j=1}^{n} p_j =: M,$$

which severely limits the achievable packing rate when working over Paillier groups. To overcome this limitation, we move to the Damgård–Jurik cryptosystem, which generalizes Paillier by extending the plaintext space from $\mathbb{Z}_N$ to $\mathbb{Z}_{N^\zeta}$ and the ciphertext space from $\mathbb{Z}_{N^2}$ to $\mathbb{Z}_{N^{\zeta+1}}$, for an arbitrary integer $\zeta \geq 1$. Importantly, efficient DDLog procedures exist also over $\mathbb{Z}^*_{N^{\zeta+1}}$ [RS21].

Protocol description. Let $g \xleftarrow{\$} \mathbb{Z}^*_{N^{\zeta+1}}$ be public. Party P_1 samples $\Delta \xleftarrow{\$} [1, 2^\lambda]$ and $r \xleftarrow{\$} \mathbb{Z}_N$ and sends the Damgård–Jurik ciphertext[4]

$$c := g^{r \cdot N^\zeta} \cdot (1+N)^\Delta.$$

For each row index $i \in [m]$, party P_0 forms the CRT encoding $u_i \in [0, M)$ of its row (before masking, as described below) and sends

$$d_i := g^{u_i}.$$

The parties then locally compute

$$z_{0,i} \leftarrow \mathsf{DDLog}(c^{u_i}), \qquad z_{1,i} \leftarrow \mathsf{DDLog}(d_i^{r \cdot N^\zeta}).$$

By construction, for each i,

$$\frac{c^{u_i}}{d_i^{r \cdot N^\zeta}} = \frac{(g^{r \cdot N^\zeta})^{u_i} \cdot ((1+N)^\Delta)^{u_i}}{(g^{u_i})^{r \cdot N^\zeta}} = (1+N)^{\Delta \cdot u_i} \pmod{N^{\zeta+1}},$$

so correctness of DDLog implies

$$z_{0,i} = \Delta \cdot u_i + z_{1,i} \pmod{N^\zeta}.$$

The modulus removal trick [RS21] then gives

$$\underbrace{(z_{0,i} + \rho_i \bmod N^\zeta)}_{z'_{0,i}} = \Delta \cdot u_i + \underbrace{(z_{1,i} + \rho_i \bmod N^\zeta)}_{z'_{1,i}}$$

[4] For this overview, we use the plaintext encoding $x \mapsto (1+N)^x$ and the corresponding logarithm in DDLog, normalized by $\log((1+N)^x) = x \bmod N^\zeta$. The formal construction instead uses the Taylor-series exp/ log convention of Section 3.2.

over $\mathbb{Z}$ (without a modulus), except with probability $\frac{|\Delta \cdot u_i|}{N^\zeta}$ by Lemma 2, where $\rho_i \xleftarrow{\$} \mathbb{Z}_{N^\zeta}$ is a common random shift derived as $\mathsf{PRF}(\mathsf{sk}, i)$ for a public seed sk.

Applying CRT, this yields m VOLE instances over each prime p_j:

$$(z'_{0,i} \bmod p_j) = \Delta \cdot \underbrace{(u_i \bmod p_j)}_{x_{i,j}} + (z'_{1,i} \bmod p_j) \pmod{p_j}. \tag{1}$$

Controlling message size. At first sight, increasing ζ does not improve the packing rate, since the size of d_i appears to grow with ζ. Our key observation is that this is not necessary. For any $d_i \in \mathbb{Z}^*_{N^{\zeta+1}}$, there exists $s_i \in \mathbb{Z}_{N^\zeta}$ such that

$$d_i = (d_i \bmod N) \cdot (1+N)^{s_i} \bmod N^{\zeta+1}.$$

Thus, it suffices for P_0 to send

$$d'_i := g^{u_i} \bmod N,$$

since

$$\frac{c^{u_i}}{(d'_i)^{r \cdot N^\zeta}} = \frac{c^{u_i}}{(d_i \cdot (1+N)^{-s_i})^{r \cdot N^\zeta}} = (1+N)^{\Delta \cdot u_i + s_i r N^\zeta} \pmod{N^{\zeta+1}},$$

which still yields

$$z_{0,i} = \Delta \cdot u_i + z_{1,i} \pmod{N^\zeta}.$$

Crucially, the size of d'_i is now independent of ζ. This makes d'_i a compressing hash of u_i.

Privacy and correctness. To ensure privacy of P_0's inputs, we mask u_i by adding $r_i \cdot M$, where $r_i \xleftarrow{\$} [0, N)$. If we choose M coprime to $\varphi(N)$, the resulting value

$$d'_i := g^{u_i + r_i \cdot M} \bmod N$$

is statistically close to a distribution independent of u_i.

Finally, to enable conversion into OT correlations, we require that each

$$x_{i,j} := u_i \bmod p_j \in \{0, 1\}.$$

Recall that correctness of $z'_{0,i}$ and $z'_{1,i}$ over the integers (rather than modulo N^ζ) fails with probability $\frac{|\Delta \cdot u_i|}{N^\zeta}$. Thus, we need

$$\Delta \cdot \Big(u_i + r_i \cdot \prod_{j=1}^{n} p_j\Big) \ll N^\zeta.$$

We choose ζ to ensure that this condition holds. Finally, the ciphertext hides Δ, while correlation-robust hashing hides the unchosen outputs from the receiver. Together with the masking above, this gives privacy for both parties.

Efficiency. A detailed analysis is deferred to Section 5, but for 128-bit security, we achieve 2.6 KiB of communication to generate 128 random OTs, substantially improving over prior DDLog-based and base-OT-based approaches in the moderate-scale regime.

2.4 Toeplitz Compression with Power-DDH

We now present our second construction, which further reduces communication by eliminating both the use of Paillier-type groups and the need for distributed discrete logarithms. The key insight is that, for the purpose of generating OT correlations, it is not necessary to convert multiplicative sharings into additive ones. Instead, multiplicative correlations can be directly transformed into OT correlations using correlation-robust hashing. This allows us to work entirely in prime-order groups and to avoid DDLog altogether.

Multiplicative correlations and OT. Let $\mathbb{G}$ be a cyclic group of prime order q with generator g, and let H be a correlation-robust hash function. Suppose party P_0 holds a bit $b \in \{0,1\}$ and a group element $f_0 \in \mathbb{G}$, while party P_1 holds $\Delta \in \mathbb{Z}_q$ and $f_1 \in \mathbb{G}$ such that

$$f_0 = f_1 \cdot g^{b \cdot \Delta}.$$

Then P_0 can locally compute $(b, H(f_0))$, while P_1 can compute

$$\big(H(f_1),\, H(f_1 \cdot g^{\Delta})\big),$$

which forms a random OT correlation when the underlying random shares are generated securely: the sender learns nothing about the choice bit, and the receiver learns nothing about the unchosen output. Thus, generating random OTs reduces to securely producing many such multiplicative correlations.

Removing DDLogs. Recall the construction of [ARS24], reviewed in Section 2.2. The aim is to generate many correlations of shares of values of the form $b \cdot \Delta$. To achieve sublinear communication, the bit vector $\mathbf{b}$ of length ℓ is rearranged as a matrix $X \in \{0,1\}^{m \times n}$ with $\ell = mn$, so that the scalar-vector product $\Delta \cdot \mathbf{b}$ becomes the matrix product XY, where $Y = \Delta \cdot I_n$. The goal is to generate shares of each entry $(XY)_{i,j}$.

We now tweak this construction by switching to a prime-order group $\mathbb{G}$ instead of a Paillier group, and by skipping the DDLogs at the end. Let $g, g_0, g_1, \ldots, g_n \in \mathbb{G}$ be public group elements. Party P_0 sends, for each row $i \in [m]$,

$$d_i := g_0^{v_i} \cdot \prod_{k=1}^{n} g_k^{x_{i,k}},$$

where v_i is a random mask. Party P_1 sends, for each column $j \in [n]$,

$$c_j := \big(g_0^{w_j},\, g_1^{w_j} \cdot g^{y_{1,j}},\, \ldots,\, g_n^{w_j} \cdot g^{y_{n,j}}\big),$$

where w_j is sampled uniformly at random. The parties can then locally compute multiplicative shares $h_{0,i,j}$ and $h_{1,i,j}$ of

$$\frac{h_{0,i,j}}{h_{1,i,j}} = \frac{c_{j,0}^{v_i} \cdot \prod_{k=1}^{n} c_{j,k}^{x_{i,k}}}{d_i^{w_j}} = g^{\sum_{k=1}^{n} x_{i,k} y_{k,j}} = g^{(XY)_{i,j}},$$

analogously to the DDLog-based construction reviewed earlier. Combining these values allows the parties to derive multiplicative sharings of each $(XY)_{i,j}$ in $\mathbb{G}$.

Baseline efficiency. With parameters $m = n^2 = \ell^{2/3}$, this approach yields communication proportional to $O(\ell^{2/3})$ group elements. While this already avoids large moduli and DDLog, the resulting concrete efficiency is still not optimal for moderate batch sizes.

Compression via Power-DDH. To further reduce communication, we adapt a further compression technique based on Toeplitz matrices [GHO20, ARS24], using the power-DDH assumption. The idea is to correlate the group generators and random exponents so that all column encodings together can be represented using only $O(n)$ group elements. This correlation requires Y to be a Toeplitz matrix (i.e., it must be constant along diagonals), which is satisfied when $Y = \Delta \cdot I_n$.

Specifically, let $r, w \in \mathbb{Z}_q$ be random values held by P_1, and define

$$g_i := g^{r^i} \quad \text{for } i = 0, \ldots, n, \qquad w_j := w \cdot r^{n-j} \quad \text{for } j = 1, \ldots, n.$$

With these choices, the encodings sent by P_1 form a Toeplitz matrix whose entries depend on only $2n$ distinct group elements, as

$$c_{j,k} = g^{r^k w_j + y_{k,j}} = g^{w \cdot r^{n+k-j} + \Delta \cdot \delta_{k-j}} = c_{j+1,k+1},$$

where δ_{k-j} is the Kronecker delta (1 if $k-j=0$ and 0 otherwise). That is, $c_{j,k}$ only depends on $k-j \in [-n, n-1]$, and so is constant along diagonals. Together with $g_1, \ldots, g_n$ sent by P_1 and $d_1, \ldots, d_m$ digests from P_0, the above protocol transfers only $3n+m$ group elements and has a transparent setup. Moreover, P_1 can retain r and distribute the hash key $(g_1, \ldots, g_n)$ once for reuse with the same receiver, sampling all other coins afresh in each batch. Subsequent batches then communicate $2n+m$ group elements, in addition to the initial n-element hash key; no trusted setup is needed.

Efficiency. Choosing $m = 3n = \sqrt{3\ell}$, ignoring integer rounding, yields total communication $O(\sqrt{\ell})$ group elements. Instantiating $\mathbb{G}$ with a standard prime-order group (e.g., Curve25519 for 128-bit security), this results in 1.25 KiB of communication to generate 128 random OTs, including the hash key. This is the smallest concrete communication achieved among our constructions, and requires no trusted setup. Computational efficiency could be improved to linearithmic in ℓ using FFT techniques [GHO20], but we do not investigate this further for our Curve25519 instantiation.

3 Preliminaries

3.1 Correlation Robustness

In this paper we need a stronger notion of correlation robustness, allowing adversarially chosen base points and both positive and negative shifts by the same secret. Following [Roy22], we specialize tweakable correlation robustness to this setting, adapting the group domains and secret-key distributions to our two constructions. We write $H(\tau; z)$ for hashing z with public tweak τ.

Definition 1 (Correlation robustness with affine queries and tweaks)**.** Let the public parameters pp specify a finite scalar key space $\mathcal{K}$ that can be sampled uniformly in polynomial time, a tweak space $\mathcal{T}$, and a public, efficiently computable hash H with λ-bit outputs. For each $\tau \in \mathcal{T}$, they specify a finite cyclic group $\mathbb{G}_\tau$ with generator g_τ and efficient group operations. Write these groups additively. After generating pp, sample one secret $\Delta \xleftarrow{\$} \mathcal{K}$ and define

$$F_\Delta(\tau, a, A) := H\big(\tau; A + a\Delta g_\tau\big), \qquad \mathcal{Q} := \{(\tau, a, A) : \tau \in \mathcal{T},\ a \in \{-1, +1\},\ A \in \mathbb{G}_\tau\}.$$

We say that H is correlation robust for these parameters if $(F_\Delta)_\Delta$ is a pseudorandom function family: for every PPT distinguisher $\mathcal{D}$, the advantage

$$\big|\Pr\big[\mathcal{D}^{F_\Delta}(\mathsf{pp}) = 1\big] - \Pr\big[\mathcal{D}^{\mathcal{R}}(\mathsf{pp}) = 1\big]\big|$$

is negligible in λ, where $\mathcal{R} : \mathcal{Q} \to \{0,1\}^\lambda$ is a uniformly random function. Probabilities are over setup and key generation and the distinguisher's coins. Queries may be chosen adaptively, and H remains public.

The two instantiations. In both protocols, the tweak is $\tau = (i, j, b)$, where (i, j) identifies an OT and $b \in \{0, 1\}$ its branch.

- **Power-DDH.** Take $\mathcal{K} = \mathbb{Z}_p$, $\mathbb{G}_\tau = \mathbb{G}$, and $g_\tau = g$, where $\mathbb{G}$ has prime order p. Writing the group operation multiplicatively gives

$$F_\Delta((i, j, b), a, A) = H(i, j, b; A\, g^{a\Delta}).$$

- **DCR.** Take $\mathcal{K} = [1, 2^\lambda]$ and, for $\tau = (i, j, b)$, $\mathbb{G}_\tau = \mathbb{Z}_{p_j}$ and $g_\tau = 1$, where $p_j > 2^\lambda$. Then
$$F_\Delta((i, j, b), a, A) = H\big(i, j, b; (A + a\Delta) \bmod p_j\big).$$

Relation to prior work and plausibility. Related correlation-robustness notions and affine hashing are used in OT extension and PSI [KKRT16, Roy22, RR22]. The need to handle both signs of a shared secret also appears in the related-key-secure approach to OT extension of Applebaum, Harnik, and Ishai [AHI11]. We follow Roy's tweakable correlation-robustness definition [Roy22], using the trivial code and restricting the coefficients to $\{-1, +1\}$, with the group domains and key distributions adapted as above. This falsifiable assumption holds for both instantiations in the random-oracle model.

3.2 Damgård–Jurik and Distributed Logarithms

We recall Damgård–Jurik encryption [DJ01] and the distributed-logarithm map used in our construction. We use the Taylor-series convention of [RS21, Section 2.2].[5]

RSA moduli. For our DCR-based construction, $\mathsf{RSA.Gen}$ samples two distinct κ-bit safe primes $P = 2P' + 1$ and $Q = 2Q' + 1$, where P', Q' are also prime, and sets $N = PQ$. We assume DCR for this modulus distribution, as in [RS21, Section 6] and [ARS24, Figure 5].

Scheme 1: DJ encryption

Let $\zeta \geq 1$ be an integer parameter. For the modulus N generated by KeyGen, the plaintext space is $\mathbb{Z}_{N^\zeta}$ and the ciphertext space is $\mathbb{Z}^*_{N^{\zeta+1}}$. Define two maps:

$$\exp : \mathbb{Z}_{N^\zeta} \to 1 + N \cdot \mathbb{Z}_{N^{\zeta+1}}, \qquad \exp(x) := \sum_{k=0}^{\zeta} \frac{(Nx)^k}{k!} \bmod N^{\zeta+1}.$$

$$\log : 1 + N \cdot \mathbb{Z}_{N^{\zeta+1}} \to \mathbb{Z}_{N^\zeta}, \qquad \log(1 + Nx) := \sum_{k=1}^{\zeta} \frac{(-N)^{k-1} x^k}{k} \bmod N^\zeta.$$

The maps exp and log are efficiently computable inverse group isomorphisms.

$\mathsf{KeyGen}(1^\lambda)$:
1. Run $(N, P, Q) \leftarrow \mathsf{RSA.Gen}$.
2. Compute $\varphi := (P - 1)(Q - 1)$.
3. Output $(\mathsf{pk}, \mathsf{sk}) = (N, \varphi)$.

$\mathsf{Enc}(\mathsf{pk}, x)$:
1. Sample $r \xleftarrow{\$} \mathbb{Z}^*_{N^{\zeta+1}}$.
2. Compute and output
$$c := r^{N^\zeta} \cdot \exp(x) \bmod N^{\zeta+1}.$$

$\mathsf{Dec}(\mathsf{pk}, \mathsf{sk}, c)$:
1. Compute $\psi := \varphi^{-1} \bmod N^\zeta$.
2. Compute and output
$$x := \psi \cdot \log(c^\varphi \bmod N^{\zeta+1}) \bmod N^\zeta.$$

The security of the Damgård-Jurik scheme relies on the DCR Assumption.

Definition 2 (Decisional Composite Residuosity (DCR) Assumption)**.** We say that the DCR problem is hard, if the following distributions are computationally indistinguishable:

$$\left\{ (N, x) \mid \begin{array}{l} (N, P, Q) \leftarrow \mathsf{RSA.Gen} \\ x \xleftarrow{\$} \mathbb{Z}^*_{N^2} \end{array} \right\}, \left\{ (N, x^N \mod N^2) \mid \begin{array}{l} (N, P, Q) \leftarrow \mathsf{RSA.Gen} \\ x \xleftarrow{\$} \mathbb{Z}^*_{N^2} \end{array} \right\}$$

[5] Throughout this subsection, we assume $\zeta < \min(P, Q)$ so that the Taylor denominators are invertible, as in [RS21, Lemma 26]. All parameter choices considered here satisfy this condition.

Theorem 3. *Assuming the DCR assumption holds, DJ encryption is a CPA-secure linearly homomorphic encryption scheme for the parameters above.*

In particular, the DDLog algorithm in a DJ group $\mathbb{Z}^*_{N^{\zeta+1}}$ is given by

$$\mathsf{DDLog}(h) = \log\left(\frac{h}{h \mod N}\right).$$

Lemma 1. *For N and ζ as above and any $g_0, g_1 \in \mathbb{Z}^*_{N^{\zeta+1}}$ such that $g_0 = g_1 \cdot \exp(x) \mod N^{\zeta+1}$ for some $x \in \mathbb{Z}$, it holds that*

$$\mathsf{DDLog}(g_0) - \mathsf{DDLog}(g_1) = x \mod N^{\zeta}.$$

Typically, the DDLog algorithm will accompany with a re-randomization procedure to get subtractive sharings over $\mathbb{Z}$. This technique is yielded by the following lifting lemma.

Lemma 2 (Lifting Lemma [RS21])**.** *For all modulus $N > 1$ and subtractive sharings over $\mathbb{Z}_N$, i.e., $x_0 - x_1 \equiv x \mod N$ for some $x \in \mathbb{Z}$, we have the following*

$$\Pr\left[(x_0 + r \mod N) - (x_1 + r \mod N) = x \mid r \xleftarrow{\$} \mathbb{Z}_N\right] = \max\left(1 - \frac{|x|}{N}, 0\right).$$

Lemma 2 directly indicates that subtractive sharings of a small value over a large modulus can be efficiently lifted to $\mathbb{Z}$ by having the two parties add a common random shift.

4 Half-Chosen OT Constructions

We give half-chosen OT constructions from Power-DDH (Section 4.1) and the Damgård-Jurik cryptosystem (Section 4.2). The functionality below follows the batch endemic OT functionality of [MRR21, Fig. 2]. It lets the adversary supply the corrupted party's output strings, so that simulation does not require programming the hash function.

Functionality 2: $\mathcal{F}_{\mathsf{OT}}$ — Half-chosen OT

Parameters. Number of OTs ℓ and string length λ.

Input. Receiver P_0 supplies $\mathbf{x} = (x_1, \ldots, x_\ell) \in \{0,1\}^\ell$; The sender P_1 has no input.

Generate strings. For each $i \in [1, \ell]$, proceed according to the corruption case:

- **Corrupted sender.** Receive $S_{i,0}, S_{i,1} \in \{0,1\}^\lambda$ from the adversary $\mathcal{A}$.
- **Corrutped receiver.** Receive $S_{i,x_i} \in \{0,1\}^\lambda$ from $\mathcal{A}$ and sample $S_{i,1-x_i} \xleftarrow{\$} \{0,1\}^\lambda$.
- **Both parties honest.** Sample $S_{i,0}, S_{i,1} \xleftarrow{\$} \{0,1\}^\lambda$.

Output. Send $(S_{i,x_i})_{i\in[1,\ell]}$ to P_0 and $\big((S_{i,0}, S_{i,1})\big)_{i\in[1,\ell]}$ to P_1.

4.1 OT from Power-DDH

We use the following variant of the Power-DDH assumption from [ARS24, Definition 8]. Security of Power-DDH assumptions was proven in the generic group model [CNS07], and their arguments can be naturally extended to our generalization by generic group model master theorems.

Definition 3 ($2n$-ary Power-DDH Assumption)**.** Sample public parameters $\mathsf{pp} \leftarrow \mathsf{Gen}(1^\lambda)$ describing a group $\mathbb{G}$ of prime order p, a generator $g \leftarrow \mathsf{Sample}(1^\lambda, \mathsf{pp})$, and

$w, r, r_0, \ldots, r_{2n-1} \xleftarrow{\$} \mathbb{Z}_p$. The $2n$-ary Power-DDH assumption holds if the following two distributions are computationally indistinguishable:

$$\Big(\mathsf{pp}, \{g^{r^i}\}_{i\in[0,n]}, \{g^{w\cdot r^i}\}_{i\in[0,2n-1]}\Big),$$
$$\Big(\mathsf{pp}, \{g^{r^i}\}_{i\in[0,n]}, \{g^{w\cdot r_i}\}_{i\in[0,2n-1]}\Big).$$

Protocol 3: Half-chosen OT from Power-DDH

Parameters. Security parameter λ; integers $n, m \geq 1$; batch size $\ell = mn$. The public parameters are $\mathsf{crs} = (\mathbb{G}, p, g, H)$, where $\mathbb{G}$ has prime order p, g is a generator, and $H : \{0,1\}^* \to \{0,1\}^\lambda$.

Assumptions. $2n$-ary Power-DDH (Definition 3) and correlation robustness of H (Definition 1).

Input. Receiver P_0 holds $X = (x_{i,j}) \in \{0,1\}^{n\times m}$; sender P_1 has no private input.

Output. For all $i \in [1,n]$ and $j \in [1,m]$, P_1 gets $(S_{i,j,0}, S_{i,j,1}) \in (\{0,1\}^\lambda)^2$, and P_0 gets $S_{i,j,x_{i,j}}$.

Sender message ($\mathrm{P}_1 \to \mathrm{P}_0$).

Sample $w, r, \Delta \xleftarrow{\$} \mathbb{Z}_p$. Compute the compact Toeplitz representation $\widetilde{T}$ and the hash key hk:

$$\widetilde{T} := \Big(g^w, \ldots, g^{w\cdot r^{n-1}}, g^{w\cdot r^n+\Delta}, g^{w\cdot r^{n+1}}, \ldots, g^{w\cdot r^{2n-1}}\Big),$$
$$\mathsf{hk} := \Big(g^r, \ldots, g^{r^n}\Big).$$

Send $(\widetilde{T}, \mathsf{hk})$ to P_0.

Receiver message ($\mathrm{P}_0 \to \mathrm{P}_1$).

Parse $\widetilde{T} = (t_0, \ldots, t_{2n-1})$ and $\mathsf{hk} = (h_1, \ldots, h_n)$. Sample $v_1, \ldots, v_m \xleftarrow{\$} \mathbb{Z}_p$ and compute the digest vector $\mathsf{dgst} = (d_1, \ldots, d_m)$, where

$$d_j := g^{v_j} \cdot \prod_{k=1}^{n} h_k^{x_{k,j}} \qquad (j \in [1,m]).$$

Send dgst to P_1.

Receiver output (P_0).

Expand $\widetilde{T}$ into the Toeplitz matrix $T_{i,k} = t_{n-i+k}$ $(1 \leq i \leq n,\ 0 \leq k \leq n)$:

$$T = \begin{pmatrix} t_{n-1} & t_n & \cdots & t_{2n-1} \\ t_{n-2} & t_{n-1} & \cdots & t_{2n-2} \\ \vdots & \vdots & \ddots & \vdots \\ t_0 & t_1 & \cdots & t_n \end{pmatrix}.$$

For every $i \in [1,n]$ and $j \in [1,m]$, compute

$$z_{0,i,j} := T_{i,0}^{v_j} \cdot \prod_{k=1}^{n} T_{i,k}^{x_{k,j}},$$
$$S_{i,j,x_{i,j}} := H\Big(i, j, x_{i,j}; z_{0,i,j}\Big),$$

and output $S_{i,j,x_{i,j}}$.

Sender output (P_1).

Parse $\mathsf{dgst} = (d_1, \ldots, d_m)$. For every $i \in [1,n]$ and $j \in [1,m]$, compute

$$z_{1,i,j} := d_j^{w\cdot r^{n-i}},$$
$$S_{i,j,0} := H\Big(i, j, 0; z_{1,i,j}\Big),$$
$$S_{i,j,1} := H\Big(i, j, 1; z_{1,i,j} \cdot g^{\Delta}\Big),$$

and output $(S_{i,j,0}, S_{i,j,1})$.

Theorem 4. *Let $n, m \geq 1$ be polynomially bounded in λ, and let $\mathbb{G}$ be a group of prime order p with generator g, where $1/p$ is negligible in λ. Assume $2n$-ary Power-DDH*

(Definition 3) and correlation robustness of H (Definition 1). Then Protocol 3 realizes $\mathcal{F}_{\mathsf{OT}}$, with $\ell = nm$ and string length λ, against static semi-honest corruption.

Proof. We index the $\ell = nm$ OTs in $\mathcal{F}_{\mathsf{OT}}$ by (i, j), as in the protocol. For each corruption case, we give an explicit simulator and compare the joint distribution of the corrupted party's view and the honest party's outputs in the real and ideal executions. The view consists of the corrupted party's input, random coins, messages, and outputs.

Correctness. Since $T_{i,k} = g^{wr^{n-i+k}} \cdot g^{\Delta \mathbf{1}_{k=i}}$, we have

$$z_{0,i,j} = g^{wr^{n-i}v_j} \prod_{k=1}^{n} g^{wr^{n-i+k}x_{k,j}} \cdot g^{\Delta x_{i,j}} = d_j^{wr^{n-i}} \cdot g^{\Delta x_{i,j}} = z_{1,i,j} \cdot g^{\Delta x_{i,j}}. \tag{2}$$

Hence, for every i, j, the receiver obtains the string indexed by $x_{i,j}$ in the sender's output pair, proving correctness.

Corrupted sender. Recall that, when the sender is corrupted, $\mathcal{F}_{\mathsf{OT}}$ lets the simulator supply both strings for each OT. The simulator does not need the receiver's input.

Simulator $\mathcal{S}_1$: corrupted sender P_1

Input. Public parameters; no receiver choices.

1. Sample the sender coins $w, r, \Delta \xleftarrow{\$} \mathbb{Z}_p$. Compute its message $(\widetilde{T}, \mathsf{hk})$ as in the protocol.
2. Sample $d_1, \ldots, d_m \xleftarrow{\$} \mathbb{G}$ and use $\mathsf{dgst} = (d_1, \ldots, d_m)$ as the simulated receiver message.
3. Like in the protocol, for every i, j, compute $z_{1,i,j} = d_j^{wr^{n-i}}$, $S_{i,j,0} = H(i, j, 0; z_{1,i,j})$, and $S_{i,j,1} = H(i, j, 1; z_{1,i,j} g^{\Delta})$. Submit these ordered pairs to $\mathcal{F}_{\mathsf{OT}}$ on behalf of the corrupted sender. The functionality delivers the selected strings to the honest receiver using its input X.
4. Output the simulated view: the sender's coins (w, r, Δ), the sent $(\widetilde{T}, \mathsf{hk})$, the received dgst, and the output pairs.

Fix any receiver input X. We compare the following hybrids.

Hybrid 0: real execution.
This is the real execution of Protocol 3.

Hybrid 1: rewrite the receiver's outputs.
Set the honest receiver's output at position (i, j) to the string $S_{i,j,x_{i,j}}$ selected from the sender's output pair, instead of computing it from $z_{0,i,j}$. Leave all other computations unchanged. By correctness, the selected string equals $H(i, j, x_{i,j}; z_{0,i,j})$, so this hybrid is identical to Hybrid 0.

Hybrid 2: sample the digests uniformly.
Sample $d_1, \ldots, d_m \xleftarrow{\$} \mathbb{G}$ directly, and compute all outputs as in Hybrid 1.

Conditional on the sender's coins, the values $u_j := v_j + \sum_{k=1}^{n} r^k x_{k,j}$ in Hybrid 1 are independent and uniform in $\mathbb{Z}_p$. Hence the digests $d_j = g^{u_j}$ are independent uniform group elements, exactly as in Hybrid 2. After the output rewrite, the sender's view and both parties' outputs depend only on these digests, the sender's coins, and X. The joint distributions in Hybrids 1 and 2 are therefore identical.

Hybrid 2 is exactly the joint ideal distribution generated by $\mathcal{S}_1$ and $\mathcal{F}_{\mathsf{OT}}$. Thus indistinguishability holds unconditionally.

Corrupted receiver. Recall that, when the receiver is corrupted, the ideal functionality $\mathcal{F}_{\mathsf{OT}}$ lets the simulator supply the chosen strings and samples the unchosen strings uniformly.

Simulator $\mathcal{S}_0$: corrupted receiver P_0

Input. Receiver choices $X = (x_{i,j})$ and public parameters.

1. Sample $r \xleftarrow{\$} \mathbb{Z}_p$ and $t_0, \ldots, t_{2n-1} \xleftarrow{\$} \mathbb{G}$. Set $\mathsf{hk} = (g^r, \ldots, g^{r^n})$ and $\widetilde{T} = (t_0, \ldots, t_{2n-1})$. Use $(\widetilde{T}, \mathsf{hk})$ as the simulated sender message.
2. Sample the receiver coins $v_1, \ldots, v_m \xleftarrow{\$} \mathbb{Z}_p$. Like in the protocol, compute the digests $d_j = g^{v_j} \prod_{k=1}^{n} (g^{r^k})^{x_{k,j}}$, the group elements $z_{0,i,j} = t_{n-i}^{v_j} \prod_{k=1}^{n} t_{n-i+k}^{x_{k,j}}$, and the chosen strings $S_{i,j,x_{i,j}} = H(i, j, x_{i,j}; z_{0,i,j})$.
3. Submit X and these chosen strings to $\mathcal{F}_{\mathsf{OT}}$ on behalf of the corrupted receiver. The functionality samples the unchosen strings and delivers both strings to the honest sender.
4. Output the simulated view: X, the receiver's coins $(v_j)_j$, the received $(\widetilde{T}, \mathsf{hk})$, the sent dgst, and the chosen strings.

Fix any receiver input X and auxiliary input, independent of the honest party's fresh coins. We first rewrite the honest sender's outputs in terms of the receiver's hash inputs and Δ. This lets the Power-DDH reduction generate the *entire* joint distribution without knowing the exponents behind its challenge elements. We then randomize the compact Toeplitz encoding, removing its dependence on Δ, before invoking correlation robustness.

Hybrid 0: real execution.
This is the real execution of Protocol 3.

Hybrid 1: rewrite the sender's outputs.
For each i, j and $b \in \{0, 1\}$, set

$$S_{i,j,b} = H\big(i, j, b;\ z_{0,i,j} \cdot g^{(b - x_{i,j})\Delta}\big). \tag{3}$$

This is identical to Hybrid 0 by correctness.

Hybrid 2: randomize the unmasked Toeplitz entries.
Write $C_k = g^{w r^k}$ for $0 \le k < 2n$. Replace these values with $C_k = g^{w r_k}$, where $r_0, \ldots, r_{2n-1} \xleftarrow{\$} \mathbb{Z}_p$. Continue to send $t_k = C_k$ for $k \neq n$ and $t_n = C_n g^{\Delta}$, with $\Delta \xleftarrow{\$} \mathbb{Z}_p$. Keep $\mathsf{hk} = (g^r, \ldots, g^{r^n})$. Compute the receiver's view from these messages, and compute the sender's outputs by the formula of Hybrid 1.

Hybrids 1 and 2 are computationally indistinguishable by the $2n$-ary Power-DDH assumption (Definition 3). Given a challenge $(\mathsf{pp}, g, (h_k)_{k=1}^{n}, (C_k)_{k=0}^{2n-1})$, where $h_k = g^{r^k}$, the reduction samples Δ and the receiver coins and constructs the messages above, with $\mathsf{hk} = (h_1, \ldots, h_n)$. It computes the digests and the bases $z_{0,i,j}$ using the receiver's instructions, and all output strings using Eq.(3). It then gives the receiver's view and the sender's output pairs to the distinguisher and returns its decision. These computations require neither w nor r: the two challenge distributions give exactly Hybrids 1 and 2.

Hybrid 3: sample the transmitted Toeplitz entries uniformly.
Sample $t_0, \ldots, t_{2n-1} \xleftarrow{\$} \mathbb{G}$ directly, independently of Δ and hk. Retain the same receiver computations and sender-output formula.

Conditional on $w \neq 0$ in Hybrid 2, the values $(C_k)_k$ are independent uniform group elements, independent of hk. Multiplying C_n by g^{Δ} leaves this distribution unchanged, even conditional on Δ. Thus the joint distributions in Hybrids 2 and 3 have statistical distance at most $1/p$, the probability of $w = 0$.

Hybrid 4: randomize the unchosen strings.
Replace every $S_{i,j,1-x_{i,j}}$ with a fresh independent uniform λ-bit string. Leave the receiver's view and all chosen strings unchanged.

Apply Definition 1 with its Power-DDH instantiation. Given the public parameters and an oracle $\mathcal{O}$, which is either F_Δ for $\Delta \xleftarrow{\$} \mathbb{Z}_p$ or a uniformly random function $\mathcal{R}$, the reduction performs the first two steps of $\mathcal{S}_0$. This generates the receiver's view, the bases $z_{0,i,j}$, and the chosen strings exactly as in Hybrid 3, independently of the hidden key Δ. In particular, the chosen strings are computed by public evaluation of H, not by querying $\mathcal{O}$.

For each i, j, it queries $\mathcal{O}$ on

$$(\tau, a, A) = \big((i, j, 1 - x_{i,j}),\ 1 - 2x_{i,j},\ z_{0,i,j}\big)$$

and uses the answer as $S_{i,j,1-x_{i,j}}$. It gives the receiver's view and the completed sender output pairs to the distinguisher and returns its decision.

If $\mathcal{O} = F_\Delta$, the answer is

$$H\big(i, j, 1 - x_{i,j}; z_{0,i,j} g^{(1-2x_{i,j})\Delta}\big),$$

exactly the unchosen string in Hybrid 3. If $\mathcal{O} = \mathcal{R}$, the distinct tweaks make all queries distinct, so the answers are independent uniform strings, giving exactly Hybrid 4. The reduction therefore has exactly the advantage of distinguishing these hybrids. One oracle handles both signs with the same hidden key; the bases may be known to the receiver and correlated with each other.

Hybrid 4 is exactly the joint ideal distribution generated by $\mathcal{S}_0$ and $\mathcal{F}_{\mathsf{OT}}$. The receiver-corruption distinguishing advantage is at most the Power-DDH reduction's advantage, plus $1/p$, plus the correlation-robustness reduction's advantage on nm queries. □

Optimization: reusing the hash key. For fixed n, the sender can retain r and distribute hk only once for polynomially many batches with the same receiver. With all other randomness sampled afresh, this preserves semi-honest security under the same assumptions. Each batch then communicates $2n + m$ group elements, in addition to n group elements for the initial key distribution. The communication figures in Section 5 include this initial key distribution; reuse offers an additional saving for subsequent batches. Since dgst depends on hk but not on $\widetilde{T}$, once the hash key is known, dgst and $\widetilde{T}$ can be exchanged in one simultaneous round. For standard OT, the receiver sends dgst first, and the sender replies with $\widetilde{T}$ together with its inputs masked by the corresponding OT strings. This gives two-message standard OT after hash-key distribution.

4.2 OT from the Damgård–Jurik cryptosystem

Protocol 4 presents our half-chosen OT protocol based on DCR, whose security is established in Theorem 5. The intuition behind the construction is given in Section 2.3.

Protocol 4: Half-chosen OT from DCR

Parameters. Security parameter λ; integers $m, n \geq 1$; batch size $\ell = mn$. The public parameters are $\mathsf{crs} = (N, p_1, \ldots, p_n, \zeta, g, H, \mathsf{PRF}, \mathsf{sk})$, where N is sampled as in Section 3.2, and $p_1, \ldots, p_n$ are distinct primes satisfying $2^\lambda < p_j < 2^{\kappa-2}$. Let $M = \prod_{j=1}^{n} p_j$ and choose an integer $\zeta \geq 2$ such that

$$N^{\zeta-1} \geq m\, 2^{2\lambda} M.$$

Sample $g \xleftarrow{\$} \mathbb{Z}^*_{N^{\zeta+1}}$ and $\mathsf{sk} \xleftarrow{\$} \{0,1\}^\lambda$. H outputs λ-bit strings, and $\mathsf{PRF} : \{0,1\}^\lambda \times [1,m] \to \mathbb{Z}_{N^\zeta}$.

Assumptions. DCR (Definition 2), security of PRF, and correlation robustness of H (Definition 1).

Input. Receiver P_0 holds $X = (x_{i,j}) \in \{0,1\}^{m \times n}$; sender P_1 has no private input.

Output. For all $i \in [1,m]$ and $j \in [1,n]$, P_1 gets $(S_{i,j,0}, S_{i,j,1}) \in (\{0,1\}^\lambda)^2$, and P_0 gets $S_{i,j,x_{i,j}}$.

Sender message ($\mathrm{P}_1 \to \mathrm{P}_0$). ***(The two messages can be exchanged simultaneously.)***

Sample $\Delta \xleftarrow{\$} [1, 2^\lambda]$ and $r \xleftarrow{\$} [0, N)$. Compute and send to P_0 the ciphertext c

$$c := g^{r \cdot N^\zeta} \cdot \exp(\Delta) \bmod N^{\zeta+1}.$$

Receiver message ($\mathrm{P}_0 \to \mathrm{P}_1$).

For all $i \in [1,m]$, use CRT to compute the unique $u_i \in [0, M)$ s.t., $u_i \bmod p_j = x_{i,j}$ for all $j \in [1,n]$. Sample $r_1, \ldots, r_m \xleftarrow{\$} [0, N)$ and compute the digest $\mathsf{dgst} = (d_1, \ldots, d_m)$, with

$$d_i := g^{u_i + r_i M} \bmod N \qquad (i \in [1,m]).$$

Send dgst to P_1.

Receiver output (P_0).

For each row $i \in [1,m]$, compute

$$z_{0,i} := \left(\mathsf{DDLog}(c^{u_i + r_i M} \bmod N^{\zeta+1}) + \mathsf{PRF}(\mathsf{sk}, i)\right) \bmod N^\zeta.$$

For every $i \in [1,m]$ and $j \in [1,n]$, compute

$$z_{0,i,j} := z_{0,i} \bmod p_j,$$
$$S_{i,j,x_{i,j}} := H\left(i, j, x_{i,j}; z_{0,i,j}\right),$$

and output $S_{i,j,x_{i,j}}$.

Sender output (P_1).

Parse $\mathsf{dgst} = (d_1, \ldots, d_m)$. For each row $i \in [1,m]$, compute

$$z_{1,i} := \left(\mathsf{DDLog}(d_i^{r \cdot N^\zeta} \bmod N^{\zeta+1}) + \mathsf{PRF}(\mathsf{sk}, i)\right) \bmod N^\zeta.$$

For every $i \in [1,m]$ and $j \in [1,n]$, compute

$$z_{1,i,j} := z_{1,i} \bmod p_j,$$
$$S_{i,j,0} := H\left(i, j, 0; z_{1,i,j}\right),$$
$$S_{i,j,1} := H\left(i, j, 1; (z_{1,i,j} + \Delta) \bmod p_j\right),$$

and output $(S_{i,j,0}, S_{i,j,1})$.

Theorem 5. *Let m, n, ζ be polynomially bounded in λ. Assume DCR (Definition 2), security of PRF, and correlation robustness of H (Definition 1). If*

$$N^{\zeta-1} \geq m\, 2^{2\lambda} M,$$

then Protocol 4 realizes $\mathcal{F}_{\mathsf{OT}}$, for $\ell = mn$, string length λ, against static semi-honest corruption.

Proof. Fix the receiver's input X. We index the $\ell = mn$ OTs in $\mathcal{F}_{\mathsf{OT}}$ by (i,j), as in the protocol.

Correctness. Let $a_i = u_i + r_i M$. Since $d_i = g^{a_i} \bmod N$, we have $g^{a_i} = d_i \exp(s_i) \bmod N^{\zeta+1}$ for some $s_i \in \mathbb{Z}_{N^\zeta}$. Since $\exp(s_i)^{rN^\zeta} = \exp(rN^\zeta s_i) = 1$, we obtain

$$c^{a_i} = d_i^{rN^\zeta} \exp(\Delta a_i) \pmod{N^{\zeta+1}}.$$

The DDLog identity and addition of the common shift $\mathsf{PRF}(\mathsf{sk}, i)$ give

$$z_{0,i} - z_{1,i} = \Delta a_i \pmod{N^\zeta}. \tag{4}$$

For the correctness analysis, replace the shifts $\mathsf{PRF}(\mathsf{sk}, i)$ by independent uniform elements of $\mathbb{Z}_{N^\zeta}$. This changes the probability of a lifting failure by at most a negligible PRF advantage ϵ_{PRF}: the reduction samples the pre-shift values for the fixed input X, uses its oracle answers as the common shifts, and tests whether any integer equality $z_{0,i} - z_{1,i} = \Delta a_i$ fails. The pre-shift values do not depend on sk.

With uniform shifts, $z_{1,i}$ is uniform, so the lifting argument of Lemma 2 gives failure probability $\Delta a_i / N^\zeta$ for row i. Since $a_i < NM$ and $\Delta \le 2^\lambda$, a union bound gives

$$\Pr[\exists i : \; z_{0,i} - z_{1,i} \neq \Delta a_i \text{ over } \mathbb{Z}] \le \epsilon_{\mathrm{corr}} := \frac{m\, 2^\lambda M}{N^{\zeta-1}} + \epsilon_{\mathrm{PRF}} \le 2^{-\lambda} + \epsilon_{\mathrm{PRF}}. \tag{5}$$

Outside this failure event, reduction modulo each p_j gives

$$z_{0,i,j} - z_{1,i,j} = \Delta a_i = \Delta x_{i,j} \pmod{p_j}, \tag{6}$$

so the receiver obtains the string indexed by $x_{i,j}$ in the sender's output pair.

Corrupted sender. Recall that, when the sender is corrupted, $\mathcal{F}_{\mathsf{OT}}$ lets the simulator supply both strings for each OT. The simulator does not need the receiver's input.

Simulator $\mathcal{S}_1$: corrupted sender P_1

Input. Public parameters; no receiver choices.

1. Sample the sender coins $\Delta \xleftarrow{\$} [1, 2^\lambda]$ and $r \xleftarrow{\$} [0, N)$. Compute $c = g^{rN^\zeta} \exp(\Delta) \bmod N^{\zeta+1}$.
2. Sample $r'_1, \ldots, r'_m \xleftarrow{\$} [0, N)$ and set $d_i = g^{Mr'_i} \bmod N$. Use $\mathsf{dgst} = (d_1, \ldots, d_m)$ as the receiver message.
3. Like in the protocol, compute the sender's row values $z_{1,i}$, set $z_{1,i,j} = z_{1,i} \bmod p_j$, and compute $S_{i,j,0} = H(i, j, 0; z_{1,i,j})$ and $S_{i,j,1} = H(i, j, 1; (z_{1,i,j} + \Delta) \bmod p_j)$. Submit these ordered pairs to $\mathcal{F}_{\mathsf{OT}}$ on behalf of the corrupted sender. The functionality delivers the selected strings to the honest receiver using its input X.
4. Output the simulated view: the sender's coins (Δ, r), the sent c, the received dgst, and the output pairs.

We compare the following hybrids.

Hybrid 0: real execution.
This is the real execution of Protocol 4.

Hybrid 1: rewrite the receiver's outputs.
Set the honest receiver's output at position (i, j) to the string $S_{i,j,x_{i,j}}$ selected from the sender's output pair, instead of computing it from $z_{0,i,j}$. Leave all other computations unchanged. By correctness, the joint distributions in Hybrids 0 and 1 have statistical distance at most ϵ_{corr}.

Hybrid 2: remove the input from the digests.
Sample $r'_1, \ldots, r'_m \xleftarrow{\$} [0, N)$ and replace every digest $g^{u_i + Mr_i} \bmod N$ by $g^{Mr'_i} \bmod N$. Compute all outputs as in Hybrid 1.

Let $\bar{g} = g \bmod N$ and $\epsilon_N := 2^{2-\kappa}$. We have

$$\frac{N - \varphi(N)}{N} = \frac{P + Q - 1}{N} \leq \epsilon_N.$$

A uniform integer in $[0, N)$, reduced modulo $\varphi(N)$, is ϵ_N-close to uniform: conditional on being less than $\varphi(N)$ it is exactly uniform, and the remaining event has probability at most ϵ_N. Since $\varphi(N) = 4P'Q'$ and each p_j is odd and smaller than P' and Q', we have $\gcd(M, N\varphi(N)) = 1$. Fix the setup, input and sender coins. Each exponent $u_i + Mr_i$ modulo $\varphi(N)$ is ϵ_N-close to uniform, as is Mr'_i. Both corresponding powers are therefore ϵ_N-close to uniform in $\langle \bar{g} \rangle$, and within $2\epsilon_N$ of one another. The rows are independent, giving distance at most $2m\epsilon_N$. After the output rewrite, the sender's view and both parties' outputs depend only on its coins, the digests and X, so the same bound holds for the joint distributions. Thus indistinguishability of Hybrids 1 and 2 holds unconditionally.

Hybrid 2 is exactly the joint ideal distribution generated by $\mathcal{S}_1$ and $\mathcal{F}_{\mathsf{OT}}$. The sender-corruption distinguishing advantage is at most $\epsilon_{\text{corr}} + 2m\epsilon_N$, which is negligible by PRF security, the choice of ζ satisfying $N^{\zeta-1} \geq m\, 2^{2\lambda} M$, and the bound on ϵ_N above.

Corrupted receiver. Recall that, when the receiver is corrupted, the ideal functionality $\mathcal{F}_{\mathsf{OT}}$ lets the simulator supply the chosen strings and samples the unchosen strings uniformly.

Simulator $\mathcal{S}_0$: corrupted receiver P_0

Input. Receiver choices $X = (x_{i,j})$ and public parameters.

1. Sample $r' \xleftarrow{\$} [0, N)$. Set $c = g^{r' N^\zeta} \bmod N^{\zeta+1}$ (thus c encrypts zero).
2. Compute the CRT representatives u_i of the rows of X. Sample the receiver coins $r_1, \ldots, r_m \xleftarrow{\$} [0, N)$. Like in the protocol, compute the digests $d_i = g^{u_i + r_i M} \bmod N$ and row values $z_{0,i}$ using c and sk, set $z_{0,i,j} = z_{0,i} \bmod p_j$, and compute the chosen strings $S_{i,j,x_{i,j}} = H(i, j, x_{i,j}; z_{0,i,j})$.
3. Submit X and these chosen strings to $\mathcal{F}_{\mathsf{OT}}$ on behalf of the corrupted receiver. The functionality samples the unchosen strings and delivers both strings to the honest sender.
4. Output the receiver's view of this execution.

The main idea of the simulator is to encrypt zero and compute the receiver's view from this ciphertext. The hybrids below justify this replacement using DCR and the randomization of the unchosen strings using correlation robustness.

Hybrid 0: real execution.
This is the real execution of Protocol 4.

Hybrid 1: rewrite the sender's outputs.
Leave the receiver's view unchanged and, for each i, j and $b \in \{0, 1\}$, set

$$S_{i,j,b} = H\big(i, j, b; (z_{0,i,j} + (b - x_{i,j})\Delta) \bmod p_j\big). \tag{7}$$

By correctness, the joint distributions in Hybrids 0 and 1 have statistical distance at most ϵ_{corr}.

Hybrid 2: encrypt zero.
Replace $c = g^{r N^\zeta} \exp(\Delta)$ by $c = g^{r' N^\zeta}$, with fresh $r' \xleftarrow{\$} [0, N)$. Recompute the receiver's pre-hash values and strings from this ciphertext, and continue to use Eq.(7) for the sender strings. Keep the original independent $\Delta \xleftarrow{\$} [1, 2^\lambda]$ in that formula.

Since g is uniform in $\mathbb{Z}^*_{N^{\zeta+1}}$, $\bar{g}^2$ is uniform in the quadratic residues modulo N, a cyclic group of order $P'Q'$. It generates this group except with probability at most

$$\frac{1}{P'} + \frac{1}{Q'} \leq \epsilon_g := 2^{3-\kappa}.$$

Conditional on $\bar{g}^2$ being a generator, g^{2N^ζ} generates the square-randomizer subgroup

$$\mathcal{R} := \{h^{2N^\zeta} : h \in \mathbb{Z}^*_{N^{\zeta+1}}\}.$$

Indeed, raising to N^ζ kills the principal-unit subgroup and permutes the quadratic residues modulo N. A uniform exponent in $[0, \varphi(N))$ can be written as $2s+\beta$, where $s \xleftarrow{\$} [0, \varphi(N)/2)$ and $\beta \xleftarrow{\$} \{0,1\}$. Thus its corresponding mask is distributed as $g^{\beta N^\zeta} R$, where $R \xleftarrow{\$} \mathcal{R}$ is independent of β. By the exponent-sampling bound above, jointly with the public g, the protocol ciphertext $g^{rN^\zeta}\exp(\mu)$ is within $\epsilon_N + \epsilon_g$ of $g^{\beta N^\zeta} R \exp(\mu)$.

Given a DJ public modulus, the reduction samples the other public parameters as in the setup. For the fixed input X, it samples Δ and the receiver coins, and requests a standard DJ challenge C on messages $(\Delta/2, 0)$ in $\mathbb{Z}_{N^\zeta}$. It samples $\beta \xleftarrow{\$} \{0,1\}$ and sets $c = C^2 g^{\beta N^\zeta} \bmod N^{\zeta+1}$. Squaring C makes its randomizer uniform in $\mathcal{R}$ and doubles its plaintext, since $\exp(\mu/2)^2 = \exp(\mu)$, giving exactly the comparison distribution above for $\mu = \Delta$ or $\mu = 0$. The reduction computes the receiver's view as in the protocol, and evaluates the sender's outputs using Eq.(7). The ciphertext-distribution comparison incurs $\epsilon_N + \epsilon_g$ at each endpoint. Thus this transition costs at most $\epsilon_{\mathrm{DJ}} + 2(\epsilon_N + \epsilon_g)$, where ϵ_{DJ} is the resulting DJ distinguishing advantage, negligible under DCR.

Hybrid 3: randomize the unchosen strings.
Replace every $S_{i,j,1-x_{i,j}}$ by an independent uniform λ-bit string. Leave everything else as in Hybrid 2.

Apply Definition 1 with its DCR instantiation. Given the public parameters and an oracle $\mathcal{O}$, which is either F_Δ for $\Delta \xleftarrow{\$} [1, 2^\lambda]$ or a uniformly random function on the same domain, the reduction performs the first two steps of $\mathcal{S}_0$. This generates the zero-encrypting ciphertext, the receiver's view, and the bases $z_{0,i,j}$ exactly as in Hybrid 2, independently of the hidden key Δ. The chosen strings are computed by public evaluation of H, not by querying $\mathcal{O}$.

For each i, j, it queries $\mathcal{O}$ on

$$(\tau, a, A) = \big((i, j, 1 - x_{i,j}),\ 1 - 2x_{i,j},\ z_{0,i,j}\big)$$

and uses the answer as $S_{i,j,1-x_{i,j}}$. Here $\mathbb{G}_\tau = \mathbb{Z}_{p_j}$ and $g_\tau = 1$, as specified by the DCR instantiation. It gives the receiver's view and the completed sender output pairs to the distinguisher and returns its decision.

If $\mathcal{O} = F_\Delta$, the answer is

$$H\big(i, j, 1 - x_{i,j}; (z_{0,i,j} + (1 - 2x_{i,j})\Delta) \bmod p_j\big),$$

exactly the unchosen string in Hybrid 2. If $\mathcal{O}$ is a uniformly random function, the distinct tweaks make all mn queries distinct, so the answers are independent uniform strings, giving exactly Hybrid 3. The reduction's advantage ϵ_{CR} therefore equals the advantage of distinguishing these hybrids. One oracle handles both signs and all moduli with the same integer key Δ; its residues need not be uniform or independent, and the bases may be known to the receiver and correlated with each other.

Hybrid 3 is exactly the joint ideal distribution generated by $\mathcal{S}_0$ and $\mathcal{F}_{\mathsf{OT}}$. The receiver-corruption advantage is at most

$$\epsilon_{\mathrm{corr}} + \epsilon_{\mathrm{DJ}} + 2(\epsilon_N + \epsilon_g) + \epsilon_{\mathrm{CR}},$$

which is negligible under the stated assumptions and parameters. □

5 Efficiency

We compare communication costs with prior work at 128-bit security and analyze the computational complexity of our constructions.

Table 1: Protocol-level communication in KiB for direct OT generation (a) and for replacing base OTs in semi-honest extensions (b), using the parameters in Section 5. Costs are rounded to two decimal places. Power-DDH costs include the initial hash key. Bold marks the lowest cost in each row of (a), and within each Original/With ours pair in (b). "NaN" marks a parameter regime not instantiated in this comparison.

(a) Direct generation of ℓ OTs

ℓ	Base OT MRR-EKE	SoftSpokenOT $k=4$	SoftSpokenOT $k=8$	PCG	DCR OT	Power-DDH OT
2^7	4.06	NaN	NaN	NaN	2.63	**1.25**
2^9	16.06	9.06	8.56	NaN	4.50	**2.47**
2^{11}	64.06	15.06	11.56	NaN	7.88	**4.91**
2^{13}	256.06	39.06	23.56	18.31	15.00	**9.81**
2^{15}	1024.06	135.06	71.56	20.81	28.88	**19.63**
2^{17}	4096.06	519.06	263.56	**23.31**	56.25	39.22
2^{19}	16384.06	2055.06	1031.56	**25.81**	111.75	78.41

(b) Replacing base OTs in semi-honest extensions

	SoftSpokenOT ($k=4$)		SoftSpokenOT ($k=8$)		PCG	
ℓ	Original	With ours	Original	With ours	Original	With ours
2^9	9.06	**6.25**	8.56	**5.75**	NaN	NaN
2^{11}	15.06	**12.25**	11.56	**8.75**	NaN	NaN
2^{13}	39.06	**36.25**	23.56	**20.75**	18.31	**15.50**
2^{15}	135.06	**132.25**	71.56	**68.75**	20.81	**18.00**
2^{17}	519.06	**516.25**	263.56	**260.75**	23.31	**20.50**
2^{19}	2055.06	**2052.25**	1031.56	**1028.75**	25.81	**23.00**

Parameter setting and cost model. Throughout this section, we measure communication in terms of total bits exchanged between the parties, excluding reusable public setup. We report costs in KiB (1 KiB = 1024 bytes). All reported costs correspond to generating ℓ random OTs with 128-bit security. For Power-DDH OT, we include the n-element hash key in every batch, although it can be reused with the same receiver. Once the hash key has been distributed, each half-chosen OT batch takes one simultaneous round, and the conversion to standard OT takes two messages. No trusted setup is needed for this protocol.

Power-DDH OT communicates $3n+m$ group elements in $\mathbb{G}$ to generate $\ell = mn$ random OTs. Ignoring integer rounding, setting $3n = m = \sqrt{3\ell}$ minimizes this cost at $2\sqrt{3\ell} \cdot |\mathbb{G}|$ bits, where $|\mathbb{G}|$ denotes the bit length of a group-element encoding. For 128-bit security, we use the prime-order subgroup of Curve25519 [Ber06], with a 256-bit compressed encoding. For the concrete estimates, we minimize $3n + m$ over positive integers with $nm \geq \ell$, discarding any excess OTs. This gives 1.25 KiB for $\ell = 2^7$ and 4.91 KiB for $\ell = 2^{11}$.

For generating $\ell = mn$ OTs, DCR OT communicates one Damgård–Jurik ciphertext and m elements in $\mathbb{Z}_N$. Writing $B = \lceil \log_2 N \rceil$, fixed-length encodings give a total

communication of

$$(\zeta + 1 + m)B \quad \text{bits}, \qquad N^{\zeta-1} \geq m\, 2^{2\lambda} M, \qquad M = \prod_{j=1}^{n} p_j.$$

For the concrete estimates, we use a 3072-bit modulus N and distinct primes $2^{128} < p_j < 2^{128} + 2^{64}$. We minimize communication over integer parameters satisfying $nm \geq \ell$ and the security bounds. This gives 2.63 KiB for $\ell = 2^7$ and 7.88 KiB for $\ell = 2^{11}$.

Baseline accounting. The EKE/Even–Mansour variant of [MRR21, Table 1] costs $32(\ell + 2)$ bytes for ℓ base OTs, hence 4160 bytes for 128 base OTs. SoftSpokenOT costs $4160 + P_k + 16\ell/k$ bytes, where $P_4 = 3072$ and $P_8 = 3584$ bytes account for the PPRF setup [Roy22, Fig. 13]. For PCG [BCG+22], we use regular LPN with compression rate $1/4$ and noise weight 64, requiring $K = 64(\log_2 \ell - 4)$ intermediate correlated OTs. These are generated using SoftSpokenOT with $k = 4$. Half-Tree [GYW+23, Fig. 4], without its optional first-level compression, adds $16K$ bytes, giving a total of $4160 + 3072 + 4K + 16K = 7232 + 20K$ bytes.

Direct generation of random OTs. We first compare our protocols as standalone methods for generating ℓ random OTs. Table 1(a) reports the concrete communication costs of our constructions alongside prior approaches, including the EKE/Even–Mansour base-OT variant of [MRR21, Table 1], SoftSpokenOT [Roy22], and PCG-based protocols [BCG+22], where the latter two are originally designed for generating an arbitrary polynomial number of OTs. Our constructions communicate less than the base-OT and SoftSpokenOT baselines at every supported batch size in the table. Under this cost model, the largest power-of-two batch sizes at which DCR OT and Power-DDH OT beat the PCG baseline are 2^{13} and 2^{15}, respectively.

Replacing base OTs in existing OT extension protocols. Since SoftSpokenOT [Roy22] and PCG-based protocols [BCG+22] are fundamentally OT extension schemes, their overall efficiency is influenced by the cost of generating base OTs. Our constructions can therefore also be used as drop-in replacements for base OT generation within the semi-honest versions of these frameworks. In Table 1(b), Power-DDH OT replaces only the 128 base OTs at the bottom of each extension pipeline. In particular, the PCG variant retains SoftSpokenOT and its PPRF setup to generate the intermediate OTs with the common XOR correlation required by Half-Tree; our half-chosen OTs do not directly provide this correlation. Each replacement reduces the base-OT cost from 4160 to 1280 bytes, saving 2880 bytes (2.8125 KiB) per batch. The table reports the resulting absolute communication costs; the relative savings are largest at the smaller batch sizes.

Computational complexity. As for computational costs, Power-DDH OT is dominated by $O(\ell^{3/2})$ group operations and $O(\ell)$ exponentiations. The former could potentially be optimized to $O(\ell \log \ell)$ group operations by adapting the algorithm of [Lee86] for multiplying a vector by a Toeplitz matrix using the Fast Fourier Transform. We do not pursue this optimization here. The computational complexity of DCR OT is dominated by $O(\ell)$ exponentiations in $\mathbb{Z}_{N^{\zeta+1}}$, where ζ is typically at most 20 for $\ell \leq 2^{13}$.

Acknowledgments

The authors thank Pierre Meyer, Peter Scholl and the anonymous reviewers for inputs and suggestions.

This research was supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme under grant agreement number 101124977 (DECRYPSIS) and the Danish Independent Research Council under Grant-IDs DFF-3103-00077B (CryptoDigi).

During the preparation of the camera-ready version, we used AI tools to improve the exposition, clarify and check details of the security proofs, and review the efficiency analysis and comparisons with prior work. The authors take full responsibility for the final text and all technical claims.

References

[ADOS22] Damiano Abram, Ivan Damgård, Claudio Orlandi, and Peter Scholl. An algebraic framework for silent preprocessing with trustless setup and active security. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology - CRYPTO 2022*, volume 13510 of *LNCS*, pages 421–452. Springer, 2022. doi:10.1007/978-3-031-15985-5_15.

[AHI11] Benny Applebaum, Danny Harnik, and Yuval Ishai. Semantic security under related-key attacks and applications. In *Innovations in Computer Science - ICS 2011*, pages 45–60, 2011. URL: https://eprint.iacr.org/2010/544.

[ARS24] Damiano Abram, Lawrence Roy, and Peter Scholl. Succinct homomorphic secret sharing. In Marc Joye and Gregor Leander, editors, *Advances in Cryptology - EUROCRYPT 2024*, volume 14656 of *LNCS*, pages 301–330. Springer, 2024. doi:10.1007/978-3-031-58751-1_11.

[BCG+17] Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, and Michele Orrù. Homomorphic secret sharing: Optimizations and applications. In Bhavani Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*, pages 2105–2122. ACM, 2017. doi:10.1145/3133956.3134107.

[BCG+19a] Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, Peter Rindal, and Peter Scholl. Efficient two-round OT extension and silent non-interactive secure computation. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *CCS 2019*, pages 291–308. ACM, 2019. doi:10.1145/3319535.3354255.

[BCG+19b] Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, and Peter Scholl. Efficient pseudorandom correlation generators: Silent OT extension and more. In *CRYPTO 2019*, volume 11694 of *LNCS*, pages 489–518. Springer, 2019. URL: https://@doi.org/10.1007/978-3-030-26954-8_16, doi:10.1007/978-3-030-26954-8_16.

[BCG+20] Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, and Peter Scholl. Correlated pseudorandom functions from variable-density LPN. In Sandy Irani, editor, *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020*, pages 1069–1080. IEEE, 2020. doi:10.1109/FOCS46700.2020.00103.

[BCG+22] Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, Nicolas Resch, and Peter Scholl. Correlated pseudorandomness from expand-accumulate codes. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology - CRYPTO 2022 - 42nd Annual International Cryptology*

Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15-18, 2022, Proceedings, Part II, volume 13508 of *LNCS*, pages 603–633. Springer, 2022. doi:10.1007/978-3-031-15979-4_21.

[BCG+23] Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, Nicolas Resch, and Peter Scholl. Oblivious transfer with constant computational overhead. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology - EUROCRYPT 2023*, volume 14004 of *LNCS*, pages 271–302. Springer, 2023. doi:10.1007/978-3-031-30545-0_10.

[BCM+24] Dung Bui, Geoffroy Couteau, Pierre Meyer, Alain Passelègue, and Mahshid Riahinia. Fast public-key silent OT and more from constrained naor-reingold. In Marc Joye and Gregor Leander, editors, *Advances in Cryptology - EUROCRYPT 2024*, volume 14656 of *Lecture Notes in Computer Science*, pages 88–118. Springer, 2024. doi:10.1007/978-3-031-58751-1_4.

[BCM+25] Lennart Braun, Geoffroy Couteau, Kelsey Melissaris, Mahshid Riahinia, and Elahe Sadeghi. Fast pseudorandom correlation functions from sparse LPN. In Goichiro Hanaoka and Bo-Yin Yang, editors, *Advances in Cryptology - ASIACRYPT 2025*, volume 16251 of *Lecture Notes in Computer Science*, pages 439–473. Springer, 2025. doi:10.1007/978-981-95-5122-4_14.

[Bea96] Donald Beaver. Correlated pseudorandomness and the complexity of private computations. In Gary L. Miller, editor, *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, Philadelphia, Pennsylvania, USA, May 22-24, 1996*, pages 479–488. ACM, 1996. doi:10.1145/237814.237996.

[Ber06] Daniel J. Bernstein. Curve25519: New diffie-hellman speed records. In Moti Yung, Yevgeniy Dodis, Aggelos Kiayias, and Tal Malkin, editors, *Public Key Cryptography - PKC 2006*, volume 3958 of *LNCS*, pages 207–228. Springer, 2006. doi:10.1007/11745853_14.

[BGI16] Elette Boyle, Niv Gilboa, and Yuval Ishai. Breaking the circuit size barrier for secure computation under DDH. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology - CRYPTO 2016*, volume 9814 of *LNCS*, pages 509–539. Springer, 2016. doi:10.1007/978-3-662-53018-4_19.

[BGI17] Elette Boyle, Niv Gilboa, and Yuval Ishai. Group-based secure computation: Optimizing rounds, communication, and computation. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *Advances in Cryptology - EUROCRYPT 2017*, volume 10211 of *LNCS*, pages 163–193, 2017. doi:10.1007/978-3-319-56614-6_6.

[BKS19] Elette Boyle, Lisa Kohl, and Peter Scholl. Homomorphic secret sharing from lattices without FHE. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2019*, volume 11477 of *LNCS*, pages 3–33. Springer, 2019. doi:10.1007/978-3-030-17656-3_1.

[CDD+24] Geoffroy Couteau, Lalita Devadas, Srinivas Devadas, Alexander Koch, and Sacha Servan-Schreiber. Quietot: Lightweight oblivious transfer with a public-key setup. In Kai-Min Chung and Yu Sasaki, editors, *Advances in Cryptology - ASIACRYPT 2024*, volume 15485 of *LNCS*, pages 197–231. Springer, 2024. doi:10.1007/978-981-96-0888-1_7.

[CNS07] Jan Camenisch, Gregory Neven, and Abhi Shelat. Simulatable adaptive oblivious transfer. In *EUROCRYPT 2007*, pages 573–590. Springer, 2007.

[DJ01] Ivan Damgård and Mads Jurik. A generalisation, a simplification and some applications of paillier's probabilistic public-key system. In Kwangjo Kim, editor, *PKC 2001*, volume 1992 of *LNCS*, pages 119–136. Springer, 2001. doi:10.1007/3-540-44586-2_9.

[GHO20] Sanjam Garg, Mohammad Hajiabadi, and Rafail Ostrovsky. Efficient range-trapdoor functions and applications: Rate-1 OT and more. In Rafael Pass and Krzysztof Pietrzak, editors, *Theory of Cryptography - 18th International Conference, TCC 2020, Durham, NC, USA, November 16-19, 2020, Proceedings, Part I*, volume 12550 of *Lecture Notes in Computer Science*, pages 88–116. Springer, 2020. doi:10.1007/978-3-030-64375-1_4.

[GYW+23] Xiaojie Guo, Kang Yang, Xiao Wang, Wenhao Zhang, Xiang Xie, Jiang Zhang, and Zheli Liu. Half-tree: Halving the cost of tree expansion in COT and DPF. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology - EUROCRYPT 2023*, volume 14004 of *LNCS*, pages 330–362. Springer, 2023. doi:10.1007/978-3-031-30545-0_12.

[HRK25] Sebastian Hasler, Pascal Reisert, and Ralf Küsters. Pseudorandom correlation functions from ring-lwr. In Goichiro Hanaoka and Bo-Yin Yang, editors, *Advances in Cryptology - ASIACRYPT 2025*, volume 16251 of *Lecture Notes in Computer Science*, pages 401–438. Springer, 2025. doi:10.1007/978-981-95-5122-4_13.

[IKNP03] Yuval Ishai, Joe Kilian, Kobbi Nissim, and Erez Petrank. Extending oblivious transfers efficiently. In Dan Boneh, editor, *Advances in Cryptology - CRYPTO 2003*, volume 2729 of *LNCS*, pages 145–161. Springer, 2003. doi:10.1007/978-3-540-45146-4_9.

[KK13] Vladimir Kolesnikov and Ranjit Kumaresan. Improved OT extension for transferring short secrets. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology - CRYPTO 2013*, volume 8043 of *LNCS*, pages 54–70. Springer, 2013. doi:10.1007/978-3-642-40084-1_4.

[KKRT16] Vladimir Kolesnikov, Ranjit Kumaresan, Mike Rosulek, and Ni Trieu. Efficient batched oblivious PRF with applications to private set intersection. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2016. URL: https://eprint.iacr.org/2016/799, doi:10.1145/2976749.2978381.

[KOS15] Marcel Keller, Emmanuela Orsini, and Peter Scholl. Actively secure OT extension with optimal overhead. In Rosario Gennaro and Matthew Robshaw, editors, *Advances in Cryptology - CRYPTO 2015*, volume 9215 of *LNCS*, pages 724–741. Springer, 2015. doi:10.1007/978-3-662-47989-6_35.

[Lee86] David Lee. Fast multiplication of a recursive block toeplitz matrix by a vector and its application. *J. Complex.*, 2(4):295–305, 1986. doi:10.1016/0885-064X(86)90007-5.

[LJMP90] Arjen K. Lenstra, Hendrik W. Lenstra Jr., Mark S. Manasse, and John M. Pollard. The number field sieve. In Harriet Ortiz, editor, *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing, May 13-17, 1990, Baltimore, Maryland, USA*, pages 564–572. ACM, 1990. doi:10.1145/100216.100295.

[MRR21] Ian McQuoid, Mike Rosulek, and Lawrence Roy. Batching base oblivious transfers. In Mehdi Tibouchi and Huaxiong Wang, editors, *Advances in Cryptology - ASIACRYPT 2021*, volume 13092 of *LNCS*, pages 281–310. Springer, 2021. doi:10.1007/978-3-030-92078-4_10.

[OOS17] Michele Orrù, Emmanuela Orsini, and Peter Scholl. Actively secure 1-out-of-n OT extension with application to private set intersection. In Helena Handschuh, editor, *Topics in Cryptology - CT-RSA 2017 - The Cryptographers' Track at the RSA Conference 2017, San Francisco, CA, USA, February 14-17, 2017, Proceedings*, volume 10159 of *Lecture Notes in Computer Science*, pages 381–396. Springer, 2017. doi:10.1007/978-3-319-52153-4_22.

[OSY21] Claudio Orlandi, Peter Scholl, and Sophia Yakoubov. The rise of paillier: Homomorphic secret sharing and public-key silent OT. In Anne Canteaut and François-Xavier Standaert, editors, *Advances in Cryptology - EUROCRYPT 2021*, volume 12696 of *LNCS*, pages 678–708. Springer, 2021. doi:10.1007/978-3-030-77870-5_24.

[PSS17] Arpita Patra, Pratik Sarkar, and Ajith Suresh. Fast actively secure OT extension for short secrets. In *24th Annual Network and Distributed System Security Symposium, NDSS 2017, San Diego, California, USA, February 26 - March 1, 2017*. The Internet Society, 2017. URL: https://www.ndss-symposium.org/ndss2017/ndss-2017-programme/fast-actively-secure-ot-extension-for-short-secrets/.

[Roy22] Lawrence Roy. Softspokenot: Quieter OT extension from small-field silent VOLE in the minicrypt model. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology - CRYPTO 2022*, volume 13507 of *LNCS*, pages 657–687. Springer, 2022. doi:10.1007/978-3-031-15802-5_23.

[RR22] Srinivasan Raghuraman and Peter Rindal. Blazing fast PSI from improved OKVS and subfield VOLE. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2022. URL: https://eprint.iacr.org/2022/320, doi:10.1145/3548606.3560658.

[RRT23] Srinivasan Raghuraman, Peter Rindal, and Titouan Tanguy. Expand-convolute codes for pseudorandom correlation generators from LPN. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology - CRYPTO 2023 - 43rd Annual International Cryptology Conference, CRYPTO 2023, Santa Barbara, CA, USA, August 20-24, 2023, Proceedings, Part IV*, volume 14084 of *LNCS*, pages 602–632. Springer, 2023. doi:10.1007/978-3-031-38551-3_19.

[RS21] Lawrence Roy and Jaspal Singh. Large message homomorphic secret sharing from DCR and applications. In Tal Malkin and Chris Peikert, editors, *Advances in Cryptology - CRYPTO 2021*, volume 12827 of *LNCS*, pages 687–717. Springer, 2021. doi:10.1007/978-3-030-84252-9_23.