

# From OT to OLE with Subquadratic Communication\*

Jack Doerner

University of Virginia  
Charlottesville, VA, USA  
j@ckdoerner.net

Yuval Ishai

Technion – Israel Institute of Technology  
Haifa, Israel  
Amazon Web Services, Inc.  
Seattle, WA, USA  
yuvali@cs.technion.ac.il

Iftach Haitner

Stellar Development Foundation  
San Francisco, CA, USA  
Tel Aviv University  
Tel Aviv, Israel  
iftachh@gmail.com

Nikolaos Makriyannis

Fireblocks  
New York, NY, USA  
nikos@fireblocks.com

## Abstract

Oblivious Linear Evaluation (OLE) is an algebraic generalization of oblivious transfer (OT) that forms a critical part of a growing number of applications. An OLE protocol over a modulus  $q$  enables the *receiver* party to securely evaluate a line  $a \cdot X + b$  chosen by the *sender* party on a secret point  $x \in \mathbb{Z}_q$ . Motivated by the big efficiency gap between OLE and OT and by fast OT extension techniques, we revisit the question of *reducing* OLE to OT, aiming to improve the communication cost of known reductions.

We start by observing that the Chinese Remainder Theorem (CRT) can be combined with a prior protocol of Gilboa (Crypto '99) to reduce its communication cost from  $O(\ell^2)$  to  $\tilde{O}(\ell)$  bits, for  $\ell = \log q$ . Unfortunately, whereas Gilboa's protocol is secure against a semi-honest sender and a malicious receiver, a direct application of the CRT technique is only semi-honest secure (it is insecure against malicious receivers). Thus, we employ number-theoretic techniques to protect our CRT-based protocol against malicious receivers, while still retaining a concrete advantage over Gilboa's protocol (e.g.,  $10.2\times$  less communication for  $\ell = 256$ ). Furthermore, we obtain a fully malicious OLE-to-OT reduction by applying either information-theoretic techniques with moderate overhead, or RSA-based cryptographic techniques with very low overhead.

We demonstrate the usefulness of our results in the context of OLE applications, including a post-quantum oblivious pseudo-random function (OPRF) and distributed signatures. In particular, assuming pre-existing random OT correlations, we can use our malicious-receiver OLE protocol to realize (a single instance of) the power-residue based OPRF candidate with security against a

malicious client and a semi-honest server using only 1.14 KB of communication, a  $16\times$  improvement over the best previous protocol in this setting. Using our RSA-based fully malicious OLE protocol, we achieve a  $5\times$  communication improvement over previous OT and EC-based distributed ECDSA protocols. Compared to other ECDSA protocols (including ones that use Paillier and class groups), the communication gains are more modest, but come at only a fraction of the computational cost as we avoid all expensive group operations.

## CCS Concepts

• **Theory of computation** → **Cryptographic protocols; Communication complexity.**

## Keywords

Oblivious Linear Evaluation; Oblivious Transfer

### ACM Reference Format:

Jack Doerner, Iftach Haitner, Yuval Ishai, and Nikolaos Makriyannis. 2025. From OT to OLE with Subquadratic Communication. In *Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security (CCS '25)*, October 13–17, 2025, Taipei, Taiwan. ACM, New York, NY, USA, 15 pages. <https://doi.org/10.1145/3719027.3765225>

## 1 Introduction

*Oblivious Transfer (OT)* [25, 48] is a two-party protocol that enables the *receiver*  $R$  to select one of two messages (bits or strings) supplied by the *sender*  $S$ . *Oblivious Linear Evaluation (OLE)* [47] is a natural algebraic generalization of OT in which the sender supplies the secret coefficients  $a$  and  $b$  of a line  $a \cdot X + b$  (defined over some finite ring  $\mathcal{R}$ ), and the receiver learns the evaluation of this line on a chosen secret point  $x$ . OLE can be used to perform multiplication of secrets over  $\mathcal{R}$  in a manner analogous to the way in which OT is used to perform secure multiplication of boolean secrets (i.e., “secure AND”) [29, 42]. This makes OLE a useful building block in a large variety of applications, including threshold cryptography [20, 22, 30] and secure computation of general arithmetic circuits [38, 41]. In this paper, we focus on the rings  $\mathbb{Z}_q$  for “large” primes  $q \in \mathbb{N}$  that are exponential in the security parameter. Such rings are common in applications, typically with  $128 \leq \log q \leq 512$ .

\*This is an abbreviated version. The full version is available at <http://eprint.iacr.org>.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from [permissions@acm.org](mailto:permissions@acm.org).  
CCS '25, Taipei, Taiwan

© 2025 Copyright held by the owner/author(s). Publication rights licensed to ACM.  
ACM ISBN 979-8-4007-1525-9/2025/10  
<https://doi.org/10.1145/3719027.3765225>

The primary motivation behind this work is the observation that, despite substantial optimization efforts, existing OLE protocols lag far behind their OT counterparts. Highly efficient OLE protocols do exist in the *amortized* setting from lattice assumptions [3, 18, 19, 33, 39, 45] and error-correcting codes [27, 38, 47]. Still, these only achieve their efficiency guarantees when a *large batch* of OLE relations is computed at once—typically thousands or more. Additional techniques are known for the direct implementation of single OLE instances using homomorphic encryption schemes, such as Paillier’s encryption [47] or lattice-based encryption [17], but these also incur significant overhead. For applications that require only a small number of OLE instances, no highly efficient solution is known.

*OLE in the OT-hybrid model.* What is typically the most efficient approach to realizing a small number of OLEs is via *reduction* to OT. Specifically, this is because of the existence of highly efficient OT-extension protocols [2, 5–9, 15, 35, 40, 43, 45, 49, 52, 54]. Reductions from OLE to OT are commonly specified in the *OT-hybrid model*: the parties have access to an ideal oracle that implements multiple instances of OT.<sup>1</sup> When measuring communication, we consider by default a variant of the OT-hybrid model in which parties have access to correlated randomness consisting of *random* instances of OT, with random messages of the desired length and a random selection bit. These *ROT* correlations can be generated in an offline phase, before the inputs are known, and by default are not counted towards communication costs. This metric is motivated by “silent” OT extension techniques [7], including the recently introduced *public-key* silent OT, which requires little or no communication beyond a shared public key infrastructure (PKI) [9, 15]. Ordinary OT with  $\ell$ -bit chosen messages can be reduced generically to ROT using  $2\ell + 1$  bits of communication [4], but it is often possible to achieve better efficiency by designing protocols to use ROT correlations directly.

The feasibility of information-theoretic OLE in the OT-hybrid model is implied by general feasibility results for OT-based secure computation [29, 36, 37, 42]. Combined with known results on the asymptotic circuit complexity of modular multiplication [32, 50], OLE over an  $\ell$ -bit modulus can be implemented with  $O(\ell \log \ell)$  bits of communication in the OT-hybrid model, with perfect security against semi-honest parties. A similar result with  $2^{-\kappa_s}$ -statistical security against malicious parties and  $O(\ell \log \ell) + \text{poly}(\kappa_s, \log \ell)$  communication follows from [37, Theorem 2]. However, the OLE protocols that follow from these asymptotic results are concretely inefficient for realistic values of  $\ell$  and require a super-constant number of rounds.

The first concretely efficient OLE protocol was proposed by Gilboa [28], who gave a simple and round-optimal protocol for OLE over an  $\ell$ -bit modulus that requires  $\approx 2\ell^2$  bits of communication in the OT-hybrid model (with perfect security), or  $\approx \ell^2$  bits in the ROT-hybrid model. This protocol is natively secure against a malicious receiver, who might behave in arbitrary ways, but it is only secure against a *semi-honest* sender, who is guaranteed to follow

the protocol’s instructions. A subsequent line of work attempted to achieve security when either participant is malicious, while minimizing overhead. Keller et al. [41] (implicitly) achieved this with a communication overhead factor of roughly 6 relative to Gilboa’s baseline. Haitner et al. [31] reduced this overhead factor to less than 2, either by settling for a relaxed OLE functionality (which suffices for some applications) or by relying on computational security and the DDH assumption. Purely OT-based OLE protocols with overhead factors between 2 and 4 were given by Doerner et al. [20, 21, 22].

*Breaking the quadratic communication barrier?* All of the above works build on Gilboa’s protocol as a baseline and inherit its quadratic communication in the modulus length  $\ell$ . This raises the question: is it possible to build *subquadratic* and *practical* OLE protocols in the OT-hybrid model, even when settling for security against semi-honest parties? If so, can such improvements be extended to achieve security against malicious parties?

## 1.1 Our Contribution

We answer the above questions affirmatively, improving the communication costs of OLE in the OT-hybrid model. We present three types of new protocols: (1) a semi-honest secure protocol; (2) a protocol secure against a malicious receiver and a semi-honest sender; (3) fully secure protocols, i.e., secure against malicious behavior by either participant. All protocols have subquadratic (in fact, nearly linear) communication cost in the modulus length  $\ell$  and offer a concrete improvement even for small values of  $\ell$  that arise in applications; we provide cost estimates for practically-relevant parameter combinations in Section 1.2. In the remainder of this section, we give a brief and intuitive overview of the above protocols and some of their applications; a more detailed technical overview appears in Section 2.

*Semi-honest OLE.* We begin with the observation that the Chinese Remainder Theorem (CRT) can be used to improve the communication complexity of Gilboa’s protocol from  $O(\ell^2)$  to  $\tilde{O}(\ell)$  bits. Concretely, the typical improvement ranges from  $7\times$  (for  $\ell = 128$ ) to  $26\times$  (for  $\ell = 512$ ) when 40 bits of statistical security are required (i.e.,  $2^{-\kappa_s}$  distinguishing advantage for statistical parameter  $\kappa_s = 40$ ). At a high level, we apply a standard randomization technique to reduce the problem of OLE modulo  $q$  to the problem of OLE modulo a smooth integer  $n \approx q^2 \cdot 2^{\kappa_s}$ , where  $n$  is the product of a set  $\{n_i\}_{i \in [t]}$  of small, distinct prime moduli, and  $\kappa_s$  is a statistical security parameter. The parties apply Gilboa’s protocol over each modulus  $n_i$  individually, and the receiver uses the CRT to reconstruct the output modulo  $q$  from the intermediate outputs modulo each  $n_i$ . Using this approach, the communication grows quadratically in the length of each  $n_i$ . But if we choose  $n_i$  to be the  $t$  smallest primes, then it suffices to use  $t = O(\ell / \log \ell)$  primes  $n_i$  where each  $n_i \in O(\ell)$ , and the overall communication is  $O(\ell \log \ell)$ . (Here and in the following, we assume that  $\ell \geq \kappa_s$ .) Like Gilboa’s protocol, and unlike the generic approaches discussed above, this CRT-based protocol is minimally interactive: it requires only one message in each direction (sequentially), given random OT correlations.

<sup>1</sup>Using standard composition theorems for secure computation [10, 11], an OLE in the OT-hybrid model can be combined with efficient OT extension techniques to yield an OLE in the plain model. This can be done with little overhead, and thus does not constitute a concrete or asymptotic bottleneck.

*Malicious-receiver OLE.* While Gilboa’s protocol natively achieves security against a malicious receiver, the above CRT-based protocol is only secure against a semi-honest receiver; since  $n \gg q$ , a malicious receiver can choose its input  $x$  outside the expected range and thereby obtain information about the sender’s input that cannot be simulated via an ideal OLE call. Our second protocol matches Gilboa’s security properties while retaining almost the same efficiency advantage as our first protocol; the main tradeoff is that we must use a slightly larger  $n \approx q^2 \cdot 2^{5\kappa_s}$ . We eliminate the “out of range” attack by instructing the sender to perform a simple statistical consistency check of the information learned by the receiver. We present a method to perform this check with minimal communication overhead, and without increasing the number of rounds (keeping the number of rounds intact, requires the use of a random oracle). Specifically, the receiver reveals its input/output pair modulo a fixed prime, and the sender checks for consistency (to preserve the receiver’s security, it uses a slightly larger input in the semi-honest protocol). While the resulting protocol is simple, its analysis is subtle and relies on a variant of Thue’s lemma.

*Fully malicious OLE.* Our final protocol extends the above malicious-receiver protocol also to provide security against malicious senders. Our extension requires the receiver to perform a similar check to the one the sender did in the malicious-receiver protocol, now applied to the input  $a, b$  of the sender. Since a malicious sender has an additional attack vector, i.e., using inconsistent inputs to baseline Gilboa’s protocol, the consistency check uses a random prime modulus  $r$ , and the sender commits to its input  $a$  in advance, and when revealing  $a \bmod r$  it proves it is consistent with the committed value. We devise two methods for doing this *commit and modular reveal check*:

- (1) The first method is based information-theoretically upon OT via cut-and-choose techniques. As such, it can be founded on many assumptions and is plausibly post-quantum secure. Unfortunately, its bandwidth complexity grows with the product of  $\ell$  and the statistical parameter, and we do not know how to leverage the CRT to improve this asymptotic figure as we are able to elsewhere. While this variant does meaningfully inflate the concrete bandwidth cost of the overall protocol, it remains more bandwidth-efficient than prior works, both asymptotically and concretely.
- (2) The second method is based upon an efficient RSA-based integer commitment scheme, along with a simple sigma protocol. This method has a communication cost that is independent of  $\ell$  and is, concretely, far lower. Still, it cannot offer information-theoretic security in the OT-hybrid model and cannot plausibly be post-quantum secure.

Alternatively, we explore the case that the sender does *not* commit to  $a$  in advance. The resulting protocol is highly efficient, gaining 0.77 KB compared to the provable RSA-based protocol and avoiding the costly exponentiations in the RSA group. Furthermore, since it is purely based on OT, it allows for post-quantum security. The security of this variant reduces to a natural but apparently new number-theoretic conjecture (see full version) whose study may be of independent interest.

Under all three of the approaches we explore, our fully malicious protocol requires the size of  $n$  to satisfy  $n'/\log_2 n \approx q^2 \cdot 2^{12\kappa_s}$ ,

where  $n'$  is the product of all but the largest  $\kappa_s$  factors of  $n$ . For comparison, the other cases admit much smaller choices of  $n$ , and that term dominates the communication complexity.

*VOLE.* We extend our fully malicious scheme to length- $m$  VOLE (i.e.,  $m$  OLEs with the same receiver input). In the full version,<sup>2</sup> we present a new information-theoretic reduction of VOLE to OLE, realizing length- $m$  VOLE modulo a big prime  $q$  from  $m + 1$  instances of OLE modulo  $q$  and a short commitment (which can be realized by an additional OLE). This improves on a similar reduction from [24], which requires more than  $2m$  OLE instances. For short VOLEs, which are useful for some applications (see below), we present a direct generalization of our OLE protocol that beats this general reduction. In concrete terms, the total communication is  $1.5\times$  smaller at  $m = 2$  compared to the general reduction, tapering to about  $1.1\times$  at  $m = 10$ ; for larger  $m$ , the generic approach becomes more efficient.

*Applications.* We demonstrate the usefulness of our results in the context of several OLE applications. The first is a construction of a plausibly post-quantum *oblivious pseudorandom function* (OPRF) based on the power-residue PRF candidate [16]. An OPRF protocol enables a client to securely evaluate a PRF on a secret key held by a server. Yang et al. [55] show that when settling for security against a malicious client and a semi-honest server, which suffices for some use cases, the power-residue OPRF requires just a single OLE invocation, with a concrete modulus size of 384 bits. In this context, one can utilize our malicious-receiver OLE protocol, which, in the ROT-hybrid model, requires only 1.14 KB of communication over two rounds—a  $16\times$  improvement over Gilboa’s protocol. This application further motivates the design of post-quantum *public-key* silent OT protocols [9, 15] that minimize the ROT setup cost.

Second, we observe that recent concretely efficient threshold signing schemes for the ECDSA [22] and BBS+ signature schemes [23] are formulated in the hybrid model of a small number of OLE or short *Vector* OLE invocations,<sup>3</sup> and these invocations dominate the overall cost of threshold signing. We propose a simple vector extension of our OLE protocols to suit these applications. In the case that our purely-OT-based fully malicious protocol is used, we reduce the bandwidth consumption of the protocols in question by at least  $1.8\times$  and as much as  $3\times$  in the ROT-hybrid model,<sup>4</sup> while using only the general assumption of ROT correlations. (In the ROT-hybrid model, our conjecture-based variant improves communication to  $14\times$  in this model.) In the case that our integer-commitment-based fully malicious protocol is used, we reduce the bandwidth consumption of prior works by as much as  $5.5\times$  at the cost of introducing the strong RSA assumption.<sup>5</sup> Many prior works have constructed threshold ECDSA and BBS+ signing schemes under number-theoretic assumptions like DCR or class groups; however, these have universally employed homomorphic encryption schemes [1, 12–14, 26, 46] that

<sup>2</sup>The full version is available at <http://eprint.iacr.org>

<sup>3</sup>Concretely, two invocations per pair of parties, and [22] requires a vector length of two. In contrast, the other cited works do not require vectorization.

<sup>4</sup>These performance improvements are calculated relative to the OLE/VOLE protocols proposed in [22], and assume a parameterization that yields 128 bits of computational security and 40 bits of statistical security.

<sup>5</sup>This implies that the security of the overall protocol can ultimately be founded on a wide range of specific assumptions.



are far more computationally expensive than our integer commitments (see Remark 1.2).

## 1.2 Summary of Concrete Costs

In Table 1.1, we present the concrete communication costs for our protocols in the ROT-hybrid model, which is our key metric of interest. We provide further discussion of *other* costs in the full version. For comparison, we also provide communication costs under equivalent parameterizations for the protocols of Gilboa [28], Haitner et al. [31], and Doerner et al. [22]. All protocols are marked with the flavor of security they achieve: either semi-honest (SH), malicious-receiver (MR), or fully-malicious (FM). They are also marked with any computational assumptions or conjectures they require. Let OT+ Cnj stand for our most efficient fully malicious OLE protocol, which does not use the commit&prove functionality (but whose security only holds assuming our conjecture is true, for the right choice of parameters).

**Computational complexity.** The computational cost of our protocols is dominated by: (i) modular-arithmetic operations (this is the lion's share of the computation); (ii) sampling a random prime of size roughly  $\kappa_s$  (FM variants only); and (iii) RSA group operations (for the OT+RSA protocol). Regarding (i) and (ii), to illustrate: our FM protocols require two CRT encodings (one per party), one CRT decoding (for the receiver), one sampling of a random prime of size  $\kappa_s$  (receiver only), and three modular reductions (two by small primes of size  $\kappa_s$  and one by the target prime of size  $|q|$ ). The primes used in CRT encoding and decoding are of size  $O(\log \log q)$ —in practice, at most 10 bits long—so all operations in (i) and (ii) are extremely lightweight. Specifically, CRT encoding involves  $t$  modular reductions  $a \bmod p_i$  with small  $p_i$ , while CRT decoding computes  $\sum_{i=1}^t a_i c_i \bmod n$  with constants  $c_i$ , and small  $a_i$  (encoding and decoding each cost  $O(t^2)$  when the  $p_i$ 's are small enough). By comparison, a single exponentiation in an EC group of size  $q$  costs  $\omega(|q|^2)$ . Concretely, for  $|q| = 256$ , which is associated with  $t = 177$  (see appendix in full version), the entire modular-arithmetic cost of our protocol is much lower than a single exponentiation in a  $q$ -sized group. The cost of sampling a random small prime is also modest: our benchmarks<sup>6</sup> show about 100  $\mu$ s for an 80-bit prime and 700  $\mu$ s for a 256-bit prime. This is comparable to the cost of a few operations in Secp256k1 (the Bitcoin curve). Finally, the RSA-group operations in our OT+RSA protocol (three multi-exponentiations in total) dominate its computational cost by at least an order of magnitude. This is because full exponentiation in an RSA group is significantly more expensive than an EC group at the 128-bit security level.

**Remark 1.2.** Perhaps a more appropriate comparison for our OT+RSA protocol is with protocols based on number-theoretic assumptions such as DCR. Several DCR-based OLE protocols achieve communication of only a few KB, but their reliance on Paillier encryption makes them computationally expensive. Security-wise, a protocol from [34] realizes the OLE functionality, but it involves cut-and-choose techniques which makes it comparatively heavy in both computation and communication compared to more recent protocols. On the flip side, the recent protocols from [1, 12, 26, 46] have

not been shown to realize the OLE functionality, though several works could potentially be adapted to do so (with some modifications). Complexity-wise, ignoring the non-RSA costs of our protocol which are negligible in this comparison, our protocol is expected to be about an order of magnitude faster than these, since the commit-and-prove RSA-based protocol (essentially a subprotocol in those works) accounts for less than a tenth of the cost (their dominant overhead being Paillier encryption and decryption). For  $q = 256$ , the most optimized variant (from [1], which relies on comparatively strong assumptions) requires roughly 3KB of communication and 50ms of computation. Thus, our work achieves comparable—or better—communication at only a fraction of the computational cost. A similar comparison holds for class-group-based protocols [13, 14].

## Organization

A high-level overview of our techniques is given in Section 2. In Section 3, we review the necessary preliminaries. In Section 4, we describe our semi-honest and malicious-receiver OLE protocols. In the full version, we describe our fully-secure OLE protocol, we describe and prove the OT-based and RSA-based sub-protocols that our fully-secure OLE requires, and we prove several theorems and lemmas that we introduce introduced. In the applications section of the full version we describe several applications of our protocol and give concrete performance results in context, we report in-depth parameterizations and concrete performance results for our protocol when it stands alone. Furthermore, a generic reduction from VOLE to OLE is given.

## 2 Technical Overview

In this section, we provide a high-level overview of our techniques. For a modulus  $n \in \mathbb{N}$ , recall the definitions of the following functionalities:

- **Oblivious Transfer ( $\text{OT}_n$ ):** Takes two inputs  $(a_0, a_1) \in \mathbb{Z}_n^2$  from the sender  $S$  and a selection bit  $i \in \{0, 1\}$  from the receiver  $R$ . It returns  $a_i$  to  $R$ .
- **Oblivious Linear Evaluation ( $\text{OLE}_n$ ):** Takes two inputs  $(a, b) \in \mathbb{Z}_n^2$  from the sender  $S$  and an input  $x \in \mathbb{Z}_n$  from the receiver  $R$ . It returns  $ax + b \bmod n$  to  $R$ .
- **Vector Oblivious Linear Evaluation ( $\text{VOLE}_{m,n}$ ):** Takes two inputs  $(\mathbf{a}, \mathbf{b}) \in (\mathbb{Z}_n^m)^2$  from the sender  $S$  and an input  $x \in \mathbb{Z}_n$  from the receiver  $R$ . It returns  $\mathbf{a}x + \mathbf{b} \bmod n$  to  $R$ .

### 2.1 Gilboa's Protocol

Similar to many other works in this area [20, 22, 31, 41], our starting point is a protocol due to Gilboa [28], a simple malicious-receiver  $\text{OLE}_n$  protocol in the  $\text{OT}_n$ -hybrid model defined as follows. (In the following, all ring operations in  $\mathbb{Z}_n$  are reduced modulo  $n$ ).

#### Protocol 2.1 (Gilboa's OLE protocol).

Parties: Sender  $S$  and receiver  $R$ .

Parameters:  $n \in \mathbb{N}$ . Let  $\ell \leftarrow \lceil \log(n) \rceil$ .

Oracle:  $\text{OT}_n$ .

$S$ 's input:  $(a, b) \in \mathbb{Z}_n^2$ .

$R$ 's input:  $x \in \mathbb{Z}_n$ . Let  $x_0, \dots, x_{\ell-1}$  be the bit decomposition of  $x$  (i.e.,  $x = \sum_{j=0}^{\ell-1} x_j \cdot 2^j$ ).

<sup>6</sup>Conducted on a 13th Gen Intel(R) Core(TM) i7-1365U CPU using OpenSSL 3.0.13.

Protocol	Gilboa	Ours	Ours	Ours	Ours	Ours	HMRT	DKLs
Security	MR	SH	MR	FM	FM	FM	FM	FM
Assumptions	OT	OT	OT	OT + Cnj	OT + RSA	OT	OT + DDH	OT
$ q  = 32$	0.13	0.08	0.28	1.30	2.07	5.63	0.82	1.81
$ q  = 64$	0.51	0.14	0.34	1.33	2.10	5.66	1.96	2.92
$ q  = 128$	2.05	0.29	0.50	1.42	2.19	6.12	4.94	7.86
$ q  = 256$	8.19	0.58	0.80	1.77	2.54	7.78	13.98	24.77
$ q  = 384$	18.43	0.90	1.14	2.17	2.94	9.51	27.12	50.29
$ q  = 512$	32.77	1.24	1.48	2.55	3.32	11.23	44.35	84.32

**Table 1.1: Communication cost in KB, with statistical security parameter  $\kappa_s = 40$  and computational security parameter  $\kappa_c = 128$ . We denote by  $|q|$  the bit-length of the OLE modulus  $q$ . Our protocols achieve a concrete distinguishing advantage no greater than  $2^{-\kappa_s}$ , with no hidden constants. Communication is measured in the ROT-hybrid model; i.e., we assume pre-existing random OT correlations at zero cost.**

Operations:

- (1)  $S$ : Sample  $b_0, \dots, b_{\ell-2} \xleftarrow{R} \mathbb{Z}_n$  and set  $b_{\ell-1} \leftarrow b - \sum_{j=0}^{\ell-2} b_j$ .
- (2) For  $j = 0$  to  $\ell - 1$  (in parallel): The parties jointly call  $\text{OT}_n$ , with  $S$  using input  $(b_j, b_j + a \cdot 2^j)$  and  $R$  using input  $x_j$ . Let  $z_j$  denote  $R$ 's output.
- (3)  $R$ : Set  $y = \sum_{j=0}^{\ell-1} z_j$ .

Output:  $R$  outputs  $y$ . ( $S$  outputs nothing).

To see that the above protocol achieves security against malicious receivers, notice that there is no opportunity for  $R$  to cheat, as the only possible deviation involves selecting a different input, i.e., choosing the selection bits  $x_0, \dots, x_{\ell-1}$  wrongly, during the OT phase (this does not violate the protocol's security, as it is equivalent to running the protocol with a different input  $x'$ ). The protocol, however, is not secure against malicious senders.<sup>7</sup>

**2.1.1 Reducing Communication Complexity via CRT.** To achieve communication-efficient OT-based OLE, we first observe a more efficient OLE protocol for the case where  $n$  is a *smooth modulus*:  $n = \prod_i n_i$  for some small, distinct primes  $n_i$ . For such an  $n$ , the OLE can be performed as follows: (1) For each  $n_i$ , the parties perform  $\text{OLE}_{n_i}$  over their inputs modulo  $n_i$ . (2) The receiver reconstructs the output from the small outputs using the Chinese Remainder Theorem (CRT). This approach significantly reduces the communication complexity from  $\approx \log(n)^2$  to  $\approx \log(n) \cdot \log \log(n)$ , assuming that  $n$  is the product of, say, the first consecutive primes.

The above gives rise to the following more efficient OLE over a prime modulus  $q$ , assuming the parties have access to  $\text{OLE}_n$  for a smooth modulus  $n$  significantly larger than  $q$ . Specifically, if  $n > q^2 \cdot 2^{\kappa_s}$ , where  $\kappa_s$  denotes a statistical security parameter, then  $\text{OLE}_n$  can effectively be used as an OLE over the integers without any wraparound: the sender uses  $(a, b')$  as its input to  $\text{OLE}_n$ , where  $b'$  is sampled uniformly from the range  $[0, n - q^2)$  subject to  $b' = b \bmod q$ . This standard randomization technique ensures that almost no information about  $a$  or  $b$  modulo  $q$  is leaked beyond the output. Note that for the honest receiver who uses its prescribed

input  $x \in \mathbb{Z}_q$ , it holds that the value  $y' = ax + b' \bmod n$  it received from  $\text{OLE}_n$  does not wrap around  $n$ , since  $ax + b' < n$ , and it can reduce  $y'$  modulo  $q$  to obtain the desired value  $y = ax + b \bmod q$ . Asymptotically, the above protocol achieves  $O(\log(q) \cdot \log \log(q))$  communication complexity compared to  $\log(q)^2$  of Gilboa's protocol, a near-quadratic improvement. Unfortunately, unlike Gilboa's protocol, the CRT-based protocol is insecure against malicious receivers.

**2.1.1.1 The large-input attack.** Consider a malicious receiver  $\tilde{R}^*$  that uses  $x^* = \lfloor 2n/q \rfloor$  in the  $\text{OLE}_n$  call instead of its prescribed input. Since  $x^* \cdot a \geq n$  if and only if  $a > n/\lfloor 2n/q \rfloor \approx q/2$ , such a receiver causes the value  $ax^* + b$  to wrap around  $n$  based on the sender's input  $a$ , and this behavior cannot be simulated in the ideal model with access to an  $\text{OLE}_q$  functionality.<sup>8</sup>

## 2.2 Efficient OT-Based Malicious-Receiver OLE

To address the large-input attack, we instruct  $S$  to use *random* inputs in the  $\text{OLE}_n$ -call (of larger domains), and only *after* testing that  $R$  did not use a large input,  $S$  sends  $R$  the required information so it can retrieve the prescribed output. Specifically, we use the following protocol.

### Protocol 2.2 (Malicious-receiver $\text{OLE}_q$ ).

Additional parameter: Large enough prime  $p \neq q$  such that  $p \nmid n$ .

Operation:

- (1)  $S$  samples  $a' \xleftarrow{R} (s_a)$ , for sufficiently large  $s_a$ , and  $b' \xleftarrow{R} (n - s_a s_x)$ .
- (2)  $R$  sets  $x' \in [q \cdot p]$  such that  $x' = x \bmod q$  and  $x' = 0 \bmod p$ .

<sup>7</sup>Consider a malicious sender who uses input  $(0, 1)$  in the first OT call (i.e.,  $j = 0$ ) and input  $(0, 0)$  in the other calls, causing  $R$  to output  $x_0$ . For  $n \geq 3$ , this attack clearly cannot be emulated given access to (ideal) OLE since the receiver's input/output pairs  $\{(0, 0), (1, 1), (2, 0), \dots\}$  are not on the same line.

<sup>8</sup>Consider an honest sender that uses uniform  $(a, b) \xleftarrow{R} \mathbb{Z}_q^2$  as input, and let  $y$  be the output obtained by the malicious receiver  $\tilde{R}^*$ , using input  $x^*$ , at the end of the execution. Assume that after the execution of Protocol 2.1 ends, the parties call an ideal function  $\text{IsConsist}$  to verify consistency:  $\text{IsConsist}$  gets  $(a, b)$  from  $S$  and  $(x, y)$  from  $R$ , and returns true iff  $ax + b = y \bmod q$ . Instruct  $\tilde{R}^*$  to use  $(x^* \bmod q, y)$  as its input to  $\text{IsConsist}$ . Clearly,  $\text{IsConsist}$  accepts if  $a < n/x^* - 1$ , but rejects with high probability if  $x \geq n/x^*$ . When using an ideal OLE, on the other hand, the call to  $\text{IsConsist}$  either always accepts, if the malicious receiver is using the prescribed input and output, or rejects with very high probability (over  $S$ 's input). We conclude that Protocol 2.1 is insecure against the malicious receiver  $\tilde{R}^*$ .

- (3) The parties run  $\text{OLE}_n$  on inputs  $(a', b')$  and  $x'$ . Let  $y'$  denote  $R$ 's output.
- (4)  $R$  sends  $y_0 \leftarrow y' \bmod p$  to  $S$ .
- (5)  $S$ :
  - (a) Aborts if  $b' \neq y_0 \bmod p$ .
  - (b) Sends  $(\delta_a, \delta_b) \leftarrow (a - a', b - b') \bmod q$ .  
(The above ensures that the parties obtain the correct correlation if no cheat is detected.)
- (6)  $R$  outputs  $y \leftarrow y' + \delta_b + x \cdot \delta_a \bmod q$ .

Namely, the test  $S$  uses is  $b' \stackrel{?}{=} y_0 \bmod p$  for a large-enough prime  $p$ . Note that by taking  $n > q^2$ , the protocol is correct in an all-honest execution. It is also clear that the protocol is secure against semi-honest senders, i.e., the test leaks no information about  $x$ . Arguing security against malicious receivers is more challenging. Our strategy consists of proving the following regarding a malicious receiver  $R^*$  using input  $x^*$  in the  $\text{OLE}_n$  call:

- (1) If  $x^* \leq 2^{\kappa_s} \cdot q$ , then the protocol is simulateable (using oracle to  $\text{OLE}_q$ ).
- (2) Otherwise,  $R^*$  passes the test of Step 5a with only negligible probability.

Item 1 is easy to prove: for small  $x^*$ , the value of  $y' \leftarrow x^* \cdot a' + b' \bmod n$  leaks only  $\kappa_s$  bits of information about  $a'$ , intuitively, its most significant bits. Thus, by CRT, it leaks almost no information about  $a$ . However, as stated, Item 2 is *incorrect*: consider the malicious receiver  $R^*$  that uses  $x^* \leftarrow p \cdot 2^{-1} \bmod n$  (we assume for simplicity that  $n$  is an odd number), and after obtaining  $y'$ , it sends  $y_0 = y' \bmod p$ . It is easy to see that  $x^*$  is a very large input, and yet, the outcome  $R^*$  passes the test with probability  $1/2$ : if  $a'$  is even, then  $y' = pa/2 + b$  (as integers). Thus,  $y' \equiv b \bmod p$ , and the test passes.

While such a value of  $x^*$  contradicts Item 2, a closer look reveals that the obtained leakage is inconsequential to the protocol's security. Specifically, due to CRT, the parity of  $a'$  does not reveal any information about  $a' \bmod q$ , and thus the protocol is secure for such a maliciously chosen  $x^*$ . More generally, for  $x^* = c \cdot d^{-1} \bmod n$ , where both  $c$  and  $d$  are small (e.g.,  $c \leq 2^{\kappa_s} \cdot n/s_a$  and  $d \leq 2^{\kappa_s}$ ), hereafter *small*  $(c, d)$ , we show in the technical sections that using such  $x^*$  leaks at most the values of  $\lfloor ca'/n \rfloor$  and  $a' \bmod d$ . For a suitable choice of parameters, these values are completely independent of  $a' \bmod q$ .<sup>9</sup> Furthermore, given a small pair  $(c, d)$ , the above method allows us to simulate the interaction of a corrupted receiver  $R^*$  with the sender, using an oracle to  $\text{OLE}_q$ . This simulation locally emulates the leakage of  $\lfloor ca'/n \rfloor$  and  $a' \bmod d$ , both of which depend solely on ephemeral randomness and are independent of  $a' \bmod q$  and  $\delta_a$ , and thus independent of  $a$ . Given this observation, to complete the proof, we need to prove the following two facts:

- (1) There exists an efficient method to compute a small pair  $(c, d)$  from  $x^*$ , if such  $x^*$  exists.
- (2) If  $x^*$  admits no small pair, then the test in Step 5a of Protocol 2.2 passes with only negligible probability.

*Finding a suitable pair  $(c, d)$  via rational reconstruction.* Thankfully, computing small  $(c, d)$  from  $x^*$  corresponds to the classic *rational reconstruction problem*. Given  $z \in \mathbb{Z}_n$  and bounds  $r_c, r_d \in \mathbb{N}$  such that  $n \geq 2r_c r_d$  (which our bounds on small  $(c, d)$  satisfy), the

<sup>9</sup>Intuitively,  $\lfloor ca'/n \rfloor$  leaks the higher-order bits of  $a'$ , and  $a' \bmod d$  leaks the lower-order bits of  $a'$ , but both are independent of  $a' \bmod q$ .

task is to find  $(c, d)$  such that  $d \cdot z = c \bmod n$ ,  $d \in [r_d]$  and  $|c| \leq r_c$  where the ratio  $c/d \in \mathbb{Q}$  is unique when viewed as a rational number. When a solution exists, the rational reconstruction problem can be solved efficiently as a byproduct of the extended GCD algorithm.

*Unpredictability of modular multiplication.* Note that the simulatable inputs, i.e.,  $x^*$  for which a small  $(c, d)$  exists, consist of the elements in the following set:

$$\mathcal{X}_{n, s_a, \kappa_s} := \{x \in \mathbb{N} : \exists c \in \mathbb{Z}, d \in [2^{\kappa_s}] \text{ s.t. } |c| \leq n \cdot 2^{\kappa_s} / s_a \wedge d \cdot x = c \bmod n\}.$$

The following lemma (proven using the so-called Thue's lemma, see Lemma 3.8) implies that the test in Step 5a of Protocol 2.2 passes with only negligible probability.

**LEMMA 2.3 (MODULAR MULTIPLICATION UNPREDICTABILITY, INFORMAL).** *Let  $x^* \notin \mathcal{X}_{n, s_a, \kappa_s}$  and let  $p > 2^{\kappa_s}$  be a prime such that  $p \nmid n$ . Then, for any  $y' \in \mathbb{Z}_n$  and  $y_0 \in \mathbb{Z}_p$ :*

$$\Pr_{a', b'}[y_0 = b' \bmod p \mid y' = x^* \cdot a' + b' \bmod n] \leq 2^{-\kappa_s}.$$

## 2.3 Towards Fully Malicious OLE

The case of a malicious sender is more complex, as, besides using large inputs, a malicious sender has an additional attack vector. To see that, let us take a closer look at how the protocol secures against large-input attacks, for both parties, is implemented over the prime divisors of  $n$ .

**Protocol 2.4** (Preliminary fully malicious OLE protocol).

Additional parameters: Let  $n_1, \dots, n_t$  denote the different prime divisors of  $n$ .

- (1)  $R$  samples  $x' \xleftarrow{R} (s_x)$ , for sufficiently large  $s_x$ .
- (2)  $S$  samples  $a' \xleftarrow{R} (s_a)$ , for sufficiently large  $s_a$ , and  $b' \xleftarrow{R} (n - s_a s_x)$ .
- (3) For  $i = 1, \dots, t$  (in parallel):  
Run Gilboa's protocol for  $\text{OLE}_{n_i}$  on input  $(a', b') \bmod n_i$  for  $S$  and input  $x' \bmod n_i$  for  $R$ . Let  $y'_i$  denote  $R$ 's output.
- (4)  $R$  computes  $y' \bmod n$  via CRT-reconstruction using  $y'_1, \dots, y'_t$ .
- (5) The parties perform "consistency-tests" to detect large-input attacks. If no cheating is detected,  $R$  sends  $\delta_x = x - x' \bmod q$ , and  $S$  sends back  $(\delta_a, \delta_b) = (a - a', b' - \delta_x \cdot a' - b) \bmod q$ .
- (6)  $R$  outputs  $y \leftarrow y' - x \cdot \delta_a - \delta_b \bmod q$ .

We do not know how to analyze the security Protocol 2.4 against a malicious sender that induces corruption in the executions of the small factors OLE sub-protocols; consider the following selective failure attack: For  $i = 1, \dots, 10$ , in the first OT-call inside the implementation of  $\text{OLE}_{n_i}$ , the sender sets the input to  $(\delta_{i,0}, \perp)$  instead of  $(\delta_{i,0}, \delta_{i,0} + a' \bmod n_i)$ . Effectively, the sender is guessing that the parity bit of  $a' \bmod n_i$  is zero for  $i = 1, \dots, 10$ . Since  $R$ 's input is random, the probability that  $S$  guesses correctly is approximately  $1/1000$ . In this case, it becomes challenging to analyze the distribution of  $x' \bmod q$ , given the leakage of the parity bits of  $x' \bmod n_1, \dots, x' \bmod n_{10}$  (or some other arbitrary leakage pattern).

To overcome the above issue, our security analysis assumes the entire value  $x' \bmod n_i$  is leaked. This requires that the range of  $x'$ , denoted  $s_x$  in Protocol 2.4, is suitably increased. Additionally, the product of the largest  $\kappa_s$  primes must be sufficiently smaller than  $s_x$  to ensure that the leakage remains independent of  $x' \bmod q$ .<sup>10</sup>

**2.3.0.1 Input corruptions.** A more subtle attack consists of *true* input corruptions rather than selective failures. Namely, in the first OT-call inside the implementation of  $\text{OLE}_{n_i}$ , the sender sets the input to  $(\delta_{i,0}, \delta_{i,0} + a'' \bmod n_i)$  instead of  $(\delta_{i,0}, \delta_{i,0} + a' \bmod n_i)$  for some unrelated value  $a''$ . Here, we benefit from the following key observations:

- (1) For any  $\mathcal{I} \subseteq [t]$  such that  $\prod_{i \in \mathcal{I}} n_i \leq s_x / 2^{\kappa_s}$ , the receiver's inputs ( $x' \bmod n_i$ ) are statistically independent in each call to  $\text{OLE}_{n_i}$  for  $i \in \mathcal{I}$ . (This is guaranteed by the size of  $s_x$  and since  $x'$  is uniformly distributed in  $(s_x)$ .)
- (2) Input corruptions in a call to  $\text{OLE}_{n_i}$  make guessing the value of  $a_0 \cdot x' - y' \bmod n_i$ , for  $x' \xleftarrow{R} (s_x)$ , succeed with probability at most  $1/2$ , for any  $a_0 \in \mathbb{Z}_{n_i}$ .<sup>11</sup>

From the above items, we deduce that if the malicious sender cheats in  $\ell$  different  $\text{OLE}_{n_i}$  executions, then the probability of guessing  $a_0 x' - y' \bmod n$ , i.e.,  $\max_{z \in (n)} \{\Pr[z = a_0 x' - y' \bmod n]\}$ , is bounded above by  $2^{-\ell}$  (for any  $a_0 \leq n$ ). In particular, this implies that guessing the value of  $a_0 x' - y'$  *over the integers* is also bounded above by  $2^{-\ell}$  (the latter is strictly more challenging to prove than the former).

The above observation leads to the following potential solution for detecting input corruptions: have the attacker provide a guess  $(a_0, z) \in \mathbb{Z}_n^2$  and check whether  $z = a_0 x' - y'$  over the integers. Of course, revealing the honest  $(a', b')$  is out of the question, as it would leak the honest sender's secrets. However, we can leverage the consistency test, which, *when the input corruptions are independent of the prime  $p$* , essentially tests whether  $z = a_0 x' - y'$  over the integers. (For a high-entropy value  $w$ , guessing  $w \bmod p$  for a random prime  $p$  is only slightly easier than guessing a uniform value modulo  $p$ .) To ensure that the input corruptions are independent of the prime, we instruct the receiver to sample a fresh random prime and hand it to the sender for the consistency test *only after* the sender's inputs to the OTs have been recorded.

**2.3.0.2 Achieving full security via commit&modular-reveal.** The final piece of the puzzle is enforcing that the attacker's guess—specifically the value of  $a_0$ —is independent of the random prime. To achieve this, we use a “commit&modular-reveal” functionality that, on input  $v$  provided by the sender, reveals  $(w, c \bmod w)$ , for a uniformly sampled prime  $w$ , to the receiver. Combining this functionality with Protocol 2.4 yields a fully secure protocol.<sup>12</sup>

<sup>10</sup>The rationale for using the largest  $\kappa_s$  primes is that this represents the strongest attack strategy, and the attacker will be caught with high probability if they cheat in more than  $\kappa_s$  primes.

<sup>11</sup>Actually, this requires the receiver to randomly, and unbeknownst to the sender, permute the powers of two in the baseline Gilboa's protocol. We will not address this issue in this high-level overview.

<sup>12</sup>While enforcing using commit&modular-reveal to enforce independence of  $a_0$  is critical for our current security proof, we speculate that the protocol remains secure without it. See discussion in full version.

## 2.4 Vector OLE

There are generic reductions from vector OLE (VOLE) to OLE. To construct the  $\text{VOLE}_{m,n}$  protocol, however, the best such reduction costs  $(m+1)$  times the cost of the  $\text{OLE}_n$  protocol, see full version.<sup>13</sup> Yet, as in many other constructions, one can modify our OLE protocol to get a  $\text{VOLE}_{m,n}$  in the cost of  $m$  times the cost of our  $\text{OLE}_n$  protocol. The construction is straightforward for malicious-receiver VOLE. For a fully secure VOLE, the analysis requires some non-trivial work, and the construction induces some additional cost in communication. See details in the sections.

## 3 Preliminaries

### 3.1 Notation

We use calligraphic letters to denote sets, uppercase for parties and random variables, and lowercase for values and functions. All logarithms considered here are base 2. Unless stated explicitly otherwise, all ring operations of elements of  $\mathbb{Z}_n$  are reduced modulo  $n$ . Let  $\mathbb{N}$  denote the set of natural numbers. For  $a \in \mathbb{R}$ , let  $\text{abs}(a)$  denote its absolute value. For  $n \in \mathbb{N}$ , let  $[n] := \{1, \dots, n\}$ ,  $(n) := \{0, \dots, n\}$ , and let  $\mathbb{Z}_n = \mathbb{Z}/n\mathbb{Z}$  denote the ring of integers modulo  $n$ . For an algorithm, protocol, or functionality  $A$  that has a parameter  $p$ , let  $A_v$  denote  $A$  with parameter  $p$  set to  $v$ . We use  $\leftarrow$  to indicate deterministic assignment. For a vector  $\mathbf{a} = (a_1, \dots, a_\ell)$ , let  $\mathbf{a}_i$  denote its  $i^{\text{th}}$  entry, i.e.,  $a_i$ . For a set  $S \subseteq \mathbb{Z}$ , let  $\pm S := \{a, -a : a \in S\}$ .

### 3.2 Entropy and Random Variables

All distributions considered in this work are finite. For a distribution  $X$ , let  $x \xleftarrow{R} X$  denote sampling  $x$  according to  $X$ . Similarly, for a set  $X$ ,  $x \xleftarrow{R} X$  denotes sampling  $x$  uniformly from  $X$ . The support of a distribution  $X$  over a discrete set  $X$ , is defined by  $\text{Supp}(X) := \{x \in X : X(x) > 0\}$ . For two distributions/random variables  $A, B$ , let  $\text{SD}(A, B)$  denote their *statistical distance*. We use the standard notion of min-entropy and collision probability. To extend the notion of min-entropy to the conditional case, as common in this setting, we move from min-entropy to *predictability*, which in the non-conditional case is just the exponent of the min-entropy, i.e.,  $P_\infty(X) := 2^{-H_\infty(X)}$ . We also use the natural definition of conditional collision probability.

**DEFINITION 3.1 (CONDITIONAL PREDICTABILITY AND COLLISION PROBABILITY).** For random variables  $X$  and  $Y$ , let

$$P_\infty(X | Y) := \mathbb{E}_{y \leftarrow Y} \left[ 2^{-H_\infty(X|Y=y)} \right]$$

$$\text{CP}(X | Y) := \mathbb{E}_{y \leftarrow Y} [\text{CP}(X|Y=y)]$$

### 3.3 Primes and CRT

Let  $\mathcal{P}$  denote the set of all primes, and for  $\kappa \in \mathbb{N}$ , let  $\mathcal{P}_\kappa \subset \mathcal{P}$  denote the set of all  $\kappa$ -bit primes. We extensively use the Chinese Remainder Theorem.

**THEOREM 3.2 (CHINESE REMINDER THEOREM (CRT)).** For any  $\ell \in \mathbb{N}$ , there exists an efficiently computable function  $\text{crt} : (\mathbb{Z}^{\geq 0}, \mathbb{Z}^{\geq 0})^\ell \mapsto \mathbb{Z}^{\geq 0}$  such that for relatively prime integers  $\{n_1, \dots, n_\ell\}$  and any

<sup>13</sup>The full version is available at <http://eprint.iacr.org>



$(z_1, \dots, z_\ell) \in (n_1) \times \dots \times (n_\ell)$ , it holds that  $z \leftarrow \text{crt}((z_i, n_i)_{i \in [\ell]}) \in \left(\prod_{i \in [\ell]} n_i\right)$ , and  $z = z_i \bmod n_i$  for every  $i \in [\ell]$ .

It is easy to see that  $\text{crt}$  is an injective and onto function from  $(n_1) \times \dots \times (n_\ell)$  to  $(\prod_{i \in [\ell]} n_i)$ . We also use the following fact.

**FACT 3.3.** *Let  $q, p \in \mathbb{N}$  be relatively prime, Let  $S \subset \mathbb{Z}$  be an  $s$ -size set of consecutive numbers, let  $X$  and  $Q$  be uniformly distributed over  $S$  and  $\mathbb{Z}_q$ , respectively. Then, for any  $x \in \mathbb{Z}_p$ :*

$$\text{SD}(X|_{X=x \bmod p} \bmod q, Q) \leq (pq - 1)/s.$$

**PROOF.** Let  $s' \leftarrow \lfloor s/(pq) \rfloor \cdot pq$ , let  $S'$  be the first  $s'$  elements of  $S$  and let  $X' \xleftarrow{R} S'$ . Since  $pq \mid s'$ , by CRT we have  $X'|_{X'=x \bmod p} \bmod q$  is uniform over  $\mathbb{Z}_q$ . Since  $\text{SD}(X, X') \leq (pq - 1)/s$ , this concludes the proof.  $\square$

**3.3.1 Number of Primes.** For  $n \in \mathbb{N}$ , let  $\pi(n)$  denote the number of primes in  $[n]$ , and let  $\tau(\kappa) := |\{\mathcal{P}_\kappa\}|$ , i.e., number of primes whose bit length is exactly  $\kappa$ . The following fact states a lower and upper bound on  $\pi(n)$ .

**FACT 3.4 ([51]).** *For any  $n \geq 17$ :  $\frac{n}{\log n} < \pi(n) < 1.25506 \cdot \frac{n}{\log n}$ .*

**Fact 3.4** yields the following bound.

**PROPOSITION 3.5.** *For any  $\kappa \geq 6$ :  $\tau(\kappa) > 0.24 \cdot 2^\kappa / \kappa$ .*

**PROOF.** Clearly,  $\tau(\kappa) = \pi(2^\kappa) - \pi(2^{\kappa-1})$ . Hence, by Fact 3.4

$$\begin{aligned} \tau(\kappa) &\geq \frac{2^\kappa}{\kappa} - \frac{1.25506}{2} \cdot \frac{2^\kappa}{\kappa - 1} \\ &= \frac{2^\kappa}{\kappa} \cdot \left(1 - \frac{1.25506}{2} \cdot \frac{\kappa}{\kappa - 1}\right) > \frac{2^\kappa}{\kappa} \cdot 0.24. \end{aligned}$$

$\square$

We also make use of the following bound.

**PROPOSITION 3.6.** *Let  $\mathcal{P}_{s_w}$  be the set of all primes of length  $s_w \in \mathbb{N}$ , and let  $d_v \in \mathbb{N}$ . If  $s_w \geq 5$ , then  $\forall (v, v') \in (d_v)^2$  s.t.  $v \neq v'$ :*

$$\Pr_{w \leftarrow \mathcal{P}_{s_w}} [v \bmod w = v' \bmod w] < \frac{5 \log_2 d_v}{2^{s_w}}.$$

**PROOF.** Without loss of generality, assume  $v > v'$ , and then let  $\delta \leftarrow v - v'$ . We have  $\delta \in [d_v]$  and we know that  $v \bmod w = v' \bmod w \implies \delta \bmod w = 0$ . For any fixed  $\delta \in [d_v]$ , there can be no more than  $\log_2 d_v / (s_w - 1)$  distinct values of  $w$  such that  $\delta \bmod w = 0$ .

On the other hand, per Proposition 3.5, we have  $|\mathcal{P}_{s_w}| > 0.24 \cdot 2^{s_w} / s_w$  when  $s_w \geq 6$ . The probability that a value  $w$  is sampled such that  $v \bmod w = v' \bmod w$  is therefore no more than

$$\frac{\log_2 d_v / (s_w - 1)}{|\mathcal{P}_{s_w}|} < \frac{s_w \cdot \log_2 d_v}{0.24 \cdot (s_w - 1) \cdot 2^{s_w}}$$

and since  $s_w \geq 6$  the proposition follows.  $\square$

### 3.4 Thue's Lemma and Rational Reconstruction

Intuitively, the lemma below states that every  $x \in \mathbb{Z}_n$  can be written as a 'modular' ratio  $x = c \cdot d^{-1} \bmod n$  for  $d \leq r_d$  and  $c \leq r_c$ , as long as  $r_c \cdot r_d \geq n$ . Furthermore,  $c$  and  $d$  are efficiently computable.

**LEMMA 3.7 (EFFECTIVE THUE'S LEMMA).** *There exists a poly-time function  $\widetilde{\text{gcd}}$  that on input  $n, x, r_c, r_d \in \mathbb{N}$  such that  $r_c \leq n < r_c r_d$ , outputs  $c, d \in \mathbb{Z}$  such that:*

- (1)  $d \in [r_d - 1]$ ,  $\text{abs}(c) \leq r_c - 1$ ,
- (2)  $d \cdot x = c \bmod n$ , and
- (3)  $\widetilde{\text{gcd}}(d, \frac{dx-c}{n}) = 1$ .

**PROOF.** The above theorem, without the constraint on the  $\widetilde{\text{gcd}}$ , corresponds to the effective Thue's lemma ([53, Theorem 4.7]). To address the  $\widetilde{\text{gcd}}$  constraint  $\widetilde{\text{gcd}}(d, \lambda) = 1$  for  $\lambda \leftarrow \frac{dx-c}{n}$ , consider the parameters  $n, x, c, d$  as in the theorem statement, even if they do not satisfy  $\widetilde{\text{gcd}}(d, \lambda) = 1$ . Define  $\lambda' = \widetilde{\text{gcd}}(d, \lambda)$ ; then  $n, x, c/\lambda', d/\lambda', \lambda/\lambda'$  satisfy all conditions of the theorem, including the  $\widetilde{\text{gcd}}$  constraint. We mention that  $\widetilde{\text{gcd}}$  is essentially the extended GCD algorithm.  $\square$

The next lemma is somewhat of a strengthening of Lemma 3.7. Intuitively, it states if there exists  $x \in \mathbb{Z}_n$  such that  $x = c \cdot d^{-1} \bmod n$  for  $d \leq r_d$  and  $c \leq d \cdot r_c$ , for  $n \geq 2r_c r_d^2$ , then  $c$  and  $d$  are efficiently computable.

**LEMMA 3.8 (RATIONAL RECONSTRUCTION).** *There exists poly-time function  $\widetilde{\text{gcd}}$  such that, on input  $n, x, r_d \in \mathbb{N}$  and  $r_c \in \mathbb{R}^+$ , where  $n \geq 2r_c r_d^2$ ,  $\widetilde{\text{gcd}}$  outputs  $(c, d) \in \mathbb{Z}^2$  if such a pair exists that satisfies the following properties:*

- (1)  $d \in [r_d]$  and  $\text{abs}(c) \leq d \cdot r_c$ , and
- (2)  $d \cdot x = c \bmod n$ .

*If no such pair  $(c, d)$  exists, then  $\widetilde{\text{gcd}}$  outputs  $\perp$ .*

**PROOF.** The above follows from the rational reconstruction problem ([53, Theorem 4.9, p. 90]), which states that for  $n \geq 2r_c r_d^2$ ,  $\widetilde{\text{gcd}}$  outputs  $c, d \in \mathbb{Z}$  such that  $d \in [r_d]$  and  $|c| \leq r_c'$  and  $d \cdot x = c \bmod n$  and  $c/d$  is unique as a rational number if such a pair exists. Assume that such a pair exists for  $r_c' = r_c \cdot r_d$ , and that without loss of generality  $c$  and  $d$  are coprime. Either  $c \leq d \cdot r_c$ , in which case we are done. Otherwise, by the uniqueness of  $c/d$  as a rational number, no such pair exists. As in Lemma 3.7, we mention that  $\widetilde{\text{gcd}}$  is essentially the extended  $\widetilde{\text{gcd}}$  algorithm.  $\square$

### 3.5 Universal Composability

We prove the security of our protocols in the universal composability (UC) framework [11], and in all cases we assume that the adversary has the following restrictions and capabilities:

- **Security with *abort*:** After getting its output, the adversary can instruct the ideal functionality to send  $\perp$  instead of the actual output to any subset of the honest parties.
- **Static corruptions:** The adversary must choose which parties to corrupt before the interaction starts.

In the remainder of this work, "UC security" implies the above adversarial properties. When proving UC security, we utilize the following fact: for two-party, non-reactive functionalities, it suffices



to provide *non-rewinding* (also known as “straight-line”) simulators [44, Thm. 5].<sup>14</sup> That is, to prove security, it suffices to prove correctness, and to provide a PPT straight-line (i.e., non-rewinding) simulator for the case that either of the parties is corrupt (for all possible inputs of the honest party).

**DEFINITION 3.9 (STATISTICAL VS. COMPUTATIONAL SECURITY).** We say that a protocol  $UC$ -realizes a functionality with statistical [resp., computational] security  $\varepsilon$  if the protocol has a straight-line simulator such that the statistical [resp., computational] distance between the real and emulated executions is at most  $\varepsilon$ .

The security bound  $\varepsilon$  is typically a function of the security parameter given to the protocol. Where differentiation is important, we use  $\kappa_s$  to denote the parameter that determines statistical distance, and  $\kappa_c$  to denote the parameter that determines computational distance.

### 3.6 OT and OLE

We use the standard OT, Random OT (ROT), and Correlated OT (COT) functionalities. For the latter two, we use their so-called *corruptible* variant, which allows both parties, when corrupt, to determine their outputs. These weaker variants suffice for our applications, and in some settings easier to construct.

#### Functionality 3.10 (OT<sub>n</sub>). Functionality 3.11 (ROT<sub>n</sub>).

Parameters:  $n \in \mathbb{N}$ . Parameters:  $n \in \mathbb{N}$ .  
 S's input:  $(a_0, a_1) \in \mathbb{Z}_n^2$ . Corrupt parties' input:  
 R's input:  $i \in \{0, 1\}$ . A corrupt sender can provide  
 Output: Send  $a_i$  to R.  $a_0, a_1 \in \mathbb{Z}_q$ . A corrupt receiver  
 can provide  $i \in \{0, 1\}$  and  $a_i \in \mathbb{Z}_q$ .  
 Operation:  
 If  $(a_0, a_1)$  are set, sample  $i \xleftarrow{R} \{0, 1\}$ .  
 Else, if  $(i, a_1)$  are set, sample  $a_{1-i} \xleftarrow{R} \mathbb{Z}_q$ ;  
 else, sample  $(a_0, a_1) \xleftarrow{R} \mathbb{Z}_n^2$  and  $i \xleftarrow{R} \{0, 1\}$ .  
 Output:  
 Send  $(a_0, a_1)$  to S and  $(i, a_i)$  to R.

#### Functionality 3.12 (COT<sub>n</sub>).

Parameters:  $n \in \mathbb{N}$ .  
 S's input:  $\delta \in \mathbb{Z}_n$ .  
 R's input:  $i \in \{0, 1\}$ .  
 Operation: Sample  $a_0 \xleftarrow{R} \mathbb{Z}_n$  and let  
 $a_1 \leftarrow a_0 + \delta$ .  
 Output: Send  $a_0$  to S and  $a_i$  to R.

We utilize the standard OLE and Vector OLE (VOLE) functionalities over the ring  $\mathbb{Z}_n$ .

#### Functionality 3.13 (OLE<sub>n</sub>).

Parameters:  $n \in \mathbb{N}$ .  
 S's input:  $(a, b) \in \mathbb{Z}_n^2$ .  
 R's input:  $x \in \mathbb{Z}_n$ .  
 Output: Send  
 $y \leftarrow ax + b \bmod n$  to R.

#### Functionality 3.14 (VOLE<sub>m,n</sub>).

Parameters:  $m, n \in \mathbb{N}$ .  
 S's input:  $\mathbf{a}, \mathbf{b} \in \mathbb{Z}_n^m$ .  
 R's input:  $x \in \mathbb{Z}_n$ .  
 Operation: For each  $i \in [t]$ :  
 let  $y_i \leftarrow \mathbf{a}_i \cdot x + \mathbf{b}_i \bmod n$ .  
 Output: Send  $\mathbf{y} \leftarrow (y_1, \dots, y_m)$   
 to R.

**3.6.0.1 ROT to COT.** All protocols in this paper can be implemented in the COT-hybrid model or the OT-hybrid; in the hybrid models, there is no difference in cost, because nothing extra must be transmitted. However, it is typical to instantiate both OT and COT in the ROT-hybrid model, because doing so enables preprocessing and the use of efficient techniques for generating batches of pseudorandom OT instances.<sup>15</sup> While OT<sub>n</sub> can be implemented from ROT<sub>n</sub> using  $2 \log n + 1$  bits of additional communication, COT<sub>n</sub> requires only  $\log n + 1$  bits. We therefore *present* our protocols in the COT-hybrid model, and analyze their costs in the ROT-hybrid model, assuming that the following COT protocol in the ROT-hybrid model is used.

#### Protocol 3.15 (Implementing COT using ROT).

Parameters:  $n \in \mathbb{N}$ .  
 Oracle: ROT<sub>n</sub>.  
 S's input:  $\delta \in \mathbb{Z}_n$ .  
 R's input:  $i \in \{0, 1\}$ .  
 Operations:  
 (1) The parties jointly call ROT<sub>n</sub>. Let  $(a_0, a_1)$  denote S's output and  $(i', z)$  denote R's output.  
 (2) R: Send  $f \leftarrow i \oplus i'$  to R.  
 (3) S:  
 (a) If  $f = 1$ , swap the values of  $a_0$  and  $a_1$ .  
 (b) Send  $o \leftarrow a_1 - a_0 - \delta$  to R.  
 Output: S outputs  $a_0$ . R outputs  $z$  if  $i = 0$ , and  $z - o$  otherwise.

**THEOREM 3.16 (SECURITY OF PROTOCOL 3.15).** Protocol 3.15 perfectly UC-realizes COT<sub>n</sub> in the ROT<sub>n</sub>-hybrid model.

**PROOF SKETCH: Correctness.** By construction, S outputs a uniform element in  $\mathbb{Z}_q$ , as needed. By the swap done by S, it is effectively always the case that  $i = i'$ . Hence, R outputs  $a_0$  if  $i = 0$ , and  $a_0 + \delta$ , otherwise.

**Malicious sender.** The simulator emulates a random execution of the protocol. Let  $(a_0, a_1)$  be the value the emulated call to the ROT functionality returned, assume for concreteness that the emulated receiver sent  $f = 0$ , and let  $o$  be the value the sender sent to the receiver. The simulator calls the COT oracle with input  $(a_0, \delta \leftarrow a_1 - a_0 - o)$ . Clearly, the sender's view in both executions is the same, and by the setting of  $(a_0, \delta)$ , this is also the case when adding the receiver's output.

**Malicious receiver.** The simulator acts as follows:

- (1) Emulate a random execution of the protocol until the end of Step 2. Let  $(i', a_{i'})$  be the output the emulated call to

<sup>14</sup>[44, Thm. 5] requires an implicit *start-synchronization* phase: all parties are active in the first round. But for two-party functionalities, it is easy to see that start-synchronization is not needed (it does not affect the adversary's abilities).

<sup>15</sup>For example, Silent OT extension can generate  $m$  pseudorandom OT correlations with  $o(m)$  communication.

the **ROT** functionality returned to the receiver. Assume for concreteness that the receiver sent  $f = 0$ .

- (2) Call the **COT** oracle with input  $(i', a_0)$  if  $i' = 0$ , and with input  $i'$  otherwise. Let  $a'_i$  be the returned output.
- (3) If  $i' = 0$ , sample  $o \xleftarrow{R} z_q$ ; otherwise, let  $o \leftarrow a_1 - a'_1$ . Send  $o$  as the message  $S$  sends in Step 3.
- (4) Continue the emulation of the receiver til its end.

Assume  $i' = 0$ . In the real execution, the value of  $a_1$  is hidden from the receiver. Thus,  $o$  is uniformly distributed and independent of the rest of the receiver's view and the sender's input and output. Since this is also the case in the emulated execution, the emulation is perfect.

Assume  $i' = 1$ . In the real execution, it holds that  $a_1 - o = a_0 + \delta$ , and  $S$ 's output is  $a_0$ . In the emulated execution, it holds that  $a_1 - o = a'_0 + \delta$ , for  $a'_0$  being  $S$ 's output. Hence, given  $(a_1, o, \delta)$ , the outputs  $S$ 's in the two executions are the same. Finally, we note that given the rest of the receiver's view and  $\delta$ , the value of  $o$  in both executions is uniformly distributed. In the real execution, this holds since no information has leaked about  $a_0$ , and in the emulated execution, this holds since  $a'_1$  is chosen uniformly.

□

## 4 Semi-Honest and Malicious-Receiver OLE

We begin this section by recalling Gilboa's malicious-receiver OLE protocol [28]. In Section 4.2, we describe a semi-honest CRT-based OLE protocol, and in Section 4.2, we describe a CRT-based OLE protocol with security against malicious receivers. Both protocols are defined in the hybrid model of ideal OLE functionalities over *small* moduli;<sup>16</sup> these can be instantiated using Gilboa's protocol.

### 4.1 Gilboa's Malicious-Receiver OLE

We start by describing Gilboa's protocol using our notation.

**Protocol 4.1** (Gilboa's malicious-receiver OLE).

Parameters:  $n \in \mathbb{N}$ . Let  $\ell \leftarrow \lceil \log n \rceil$ .

Oracle: **OT** <sub>$n$</sub> .

$S$ 's input:  $a, b \in \mathbb{Z}_n$ .

$R$ 's input:  $x \in \mathbb{Z}_n$ . Let  $(x_0, \dots, x_{\ell-1}) \in \{0, 1\}^\ell$  be the bit-decomposition of  $x$ , i.e.,  $x = \sum_{i=0}^{\ell-1} x_i \cdot 2^i$ .

Operations:

- (1)  $S$ : Sample  $(b_0, \dots, b_{\ell-2}) \xleftarrow{R} \mathbb{Z}_n^{\ell-1}$  and let  $b_{\ell-1} = b - \sum_{i=0}^{\ell-2} b_i$ .
- (2) For all  $i \in (\ell - 1)$  (in parallel): the parties jointly call **OT** <sub>$n$</sub>  $((b_i, b_i + 2^i \cdot a \bmod n), x_i)$ . Let  $z_i$  denote  $R$ 's output.

Output.  $R$  outputs  $\sum_{i=0}^{\ell-1} z_i \bmod n$ .

Note that when  $S$  is honest, any choice bits  $x_i$  of (a possibly malicious)  $R$  define a valid input  $x = \sum_{i=0}^{\ell-1} x_i \cdot 2^i \bmod n$ , such that the OT outputs obtained by  $R$  form a random additive secret sharing of the correct OLE output  $ax + b$ . This implies the following theorem.

**THEOREM 4.2 (SECURITY OF PROTOCOL 4.1).** *For any  $n \in \mathbb{N}$ , Protocol 4.1 perfectly UC-realizes **OLE** <sub>$n$</sub>  against semi-honest senders and malicious receivers, in the **OT** <sub>$n$</sub> -hybrid model.*

<sup>16</sup>Here *small* means roughly logarithmic in the modulus of the main protocol.

**4.1.1 Using Correlated OT.** It is easy to convert Protocol 4.1 to work in the **COT**-hybrid model, which in turn can be cheaply implemented using random OT via Theorem 3.16. This follows from the observation that in all  $\ell$  invocations of OT except the last one, the sender's first input is uniformly random. The same protocol can also be proved secure under the "endemic" variant of COT, which allows a malicious  $S$  to choose its own output and compute the output of  $R$  accordingly. This flavor of COT captures the functionality realized by pseudorandom correlation generators for COT [7, 54]. We formally describe the COT-based OLE protocol below.

**Protocol 4.3** (Gilboa's malicious-receiver OLE, using correlated OT).

Parameters:  $n \in \mathbb{N}$ . Let  $\ell \leftarrow \lceil \log n \rceil$ .

Oracle: **COT** <sub>$n$</sub> .

$S$ 's input:  $a, b \in \mathbb{Z}_n$ .

$R$ 's input:  $x \in \mathbb{Z}_n$ . Let  $(x_0, \dots, x_{\ell-1}) \in \{0, 1\}^\ell$  be the bit-decomposition of  $x$ .

Operations:

- (1) For all  $i \in (\ell - 1)$  (in parallel): the parties jointly call **COT** <sub>$n$</sub> ,  $S$  using input  $2^i \cdot a$  and  $R$  using input  $x_i$ . Let  $b_i$  denote  $S$ 's output and  $z_i$  denote  $R$ 's output.
- (2)  $S$ : Send  $o \leftarrow b - \sum_{i=0}^{\ell-1} b_i$  to  $R$ .

Output.  $R$  outputs  $o + \sum_{i=0}^{\ell-1} z_i \bmod n$ .

The only difference between the view of  $R$  in Protocol 4.3 and Protocol 4.1 is that in Step 1 of Protocol 4.3  $S$  uses a *random* value of  $b_{\ell-1}$ . Correctness is maintained because  $S$  transmits  $o$ , which is the difference between the values of  $b_{\ell-1}$  in Protocol 4.3 and Protocol 4.1, and security is maintained because the receiver learns the same information from  $z_i + o$  in Protocol 4.3 as it learned from  $z_i$  alone in Protocol 4.1.

**THEOREM 4.4 (SECURITY OF PROTOCOL 4.3).** *For any  $n \in \mathbb{N}$ , Protocol 4.3 perfectly UC-realizes **OLE** <sub>$n$</sub>  against semi-honest senders and malicious receivers, in the **COT** <sub>$n$</sub> -hybrid model.*

**PROOF SKETCH:** Correctness holds by constructions. Security against a malicious receiver holds since by controlling its output, which is allowed by the **ROT** functionality, only adds a known offset to  $\sum b_i$ , compared to the case where it let the functionality choose its output, and this offset can then be reduced from the value of  $b$  sent by  $S$ . Security against a malicious sender holds since moving from OT to COT does not give the sender additional power, and the offset  $o$  it sends in Step 2 is equivalent to changing the value of  $b$ . □

### 4.2 CRT-based Semi-Honest OLE

We now describe a simple OLE protocol for *smooth* moduli; that is, moduli that are the product of many small primes. This protocol achieves perfect security in the hybrid model of OLE instances over each of the small primes that divide the main smooth modulus. Afterward, we demonstrate that our OLE over smooth moduli can be used to construct OLE over *any* modulus  $q$ , and the resulting protocol is more efficient than using Protocol 4.3 over  $q$  directly.

#### 4.2.1 Perfectly Secure OLE over Smooth Integers.

**Protocol 4.5** (Perfectly secure OLE over smooth integers).Parameters:  $\{n_i\}_{i \in [t]}$ . Let  $n \leftarrow \prod_i n_i$ .Oracles:  $\{\text{OLE}_{n_i}\}_{i \in [t]}$ . $S$ 's input:  $a, b \in \mathbb{Z}_n$ . $R$ 's input:  $x \in \mathbb{Z}_n$ .Operation. For all  $i \in [t]$  (in parallel): The parties jointly call  $\text{OLE}_{n_i}$ ,  $S$  using input  $(a_i, b_i) \leftarrow (a, b) \bmod n_i$  and  $R$  using input  $x_i \leftarrow x \bmod n_i$ . Let  $y_i$  denote the output of  $R$  for this call.Output.  $R$ : output  $\text{crt}((y_i, n_i)_{i \in [t]})$ .

**THEOREM 4.6** (SECURITY OF PROTOCOL 4.5). *Let  $\{n_i\}_{i \in [t]}$  be relatively prime, and let  $n \leftarrow \prod_i n_i$ . Protocol 4.5 with parameter  $\{n_i\}_{i \in [t]}$ , perfectly UC-realizes  $\text{OLE}_n$  in the  $\{\text{OLE}_{n_i}\}_{i \in [t]}$ -hybrid model (against malicious adversaries).*

**PROOF.** Correctness and security against semi-honest adversaries easily follow from CRT. Security against malicious parties follows correctness since the protocol gives no leeway for active adversaries to deviate from the correct execution.  $\square$

**4.2.2 Semi-Honest OLE over Arbitrary Integers.** The following protocol realizes  $\text{OLE}_q$  for an arbitrary integer  $q$  against semi-honest adversaries. The protocol uses an oracle to  $\text{OLE}_n$  for smooth  $n \gg q$ , which can be instantiated using Protocol 4.5. The reduction works by using the standard “statistical smudging” technique to reduce arithmetic modulo  $q$  to arithmetic over the integers, using  $n$  as an upper bound on the resulting integers to prevent wraparound. Unlike the previous protocol, however, here both parties have room to cheat; thus, we only get security against semi-honest adversaries.

**Protocol 4.7** (Semi-honest OLE over arbitrary integers).Parameters:  $1^{\kappa_s}$  and  $q, n \in \mathbb{N}$ .Oracle:  $\text{OLE}_n$ . $S$ 's input:  $a, b \in \mathbb{Z}_q$ . $R$ 's input:  $x \in \mathbb{Z}_q$ .

Operation:

- (1)  $S$ : Sample  $b' \xleftarrow{R} (n - q^2)$  conditioned on  $b' = b \bmod q$ .
- (2) The parties jointly call  $\text{OLE}_n$ ,  $S$  using input  $(a, b')$  and  $R$  using input  $x$ .  
Let  $y$  denote the output of  $R$  for this call.

Output:  $R$ : Output  $y \bmod q$ .

Namely, the parties use  $\text{OLE}_n$  for the receiver to compute  $y = ax + b \bmod n$ . Since  $ax + b < n$ , it holds that  $y = ax + b$ , and thus  $y = ax + b \bmod q$ .

**THEOREM 4.8** (SECURITY OF PROTOCOL 4.7). *For any  $q, n \in \mathbb{N}$  with  $n \geq q^2 \cdot 2^{\kappa_s}$ , Protocol 4.7 UC-realize Functionality 3.13 against semi-honest adversaries, in the  $\text{OLE}_n$ -hybrid model, with statistical security  $2^{-\kappa_s}$ .*

**PROOF.** Correctness and security against the semi-honest sender (who does not receive any messages) are straightforward. For the semi-honest receiver, consider the following simulator:

**Algorithm 4.9** (Simulator for the semi-honest receiver).Oracle:  $\text{OLE}_q$ .Input:  $x \in \mathbb{Z}_q$ .

Operation:

- (1) Call  $\text{OLE}_q$  with input  $x$ , let  $y_q$  be the result.
- (2) Sample  $y \xleftarrow{R} \{y_q, \dots, n\}$  conditioned on  $y = y_q \bmod q$ , and send it to  $R$  as the answer of the  $\text{OLE}_n$  call.

Fix the input  $(a, b)$  of  $S$  and the input  $x$  of  $R$ , let  $y \leftarrow ax + b$  and  $y_q \leftarrow ax + b \bmod q$ . Let  $B'$  be the value of  $b'$  sampled by  $S$  in the real execution, and let  $Y \leftarrow ax + B'$ , i.e., the output of  $\text{OLE}_n$  call. Note that  $Y$  is uniform over  $\{y, \dots, y + n - q^2 - q\}$  conditioned on  $Y = y \bmod q$ . In the simulated execution, the value of  $y$  is uniform over  $\{y_q, \dots, n\}$  conditioned on  $y = y_q \bmod q$ . Hence, the statistical distance between the two executions is bounded by  $(y - y_q)/n \leq q^2/n \leq 2^{-\kappa_s}$ .  $\square$

**4.3 CRT-based Malicious-Receiver OLE**

The CRT-based, malicious-receiver OLE protocol follows similar lines to the semi-honest protocol presented in Section 4.2, while immunizing it against the only effective attack a malicious receiver can apply: using a too large input  $x$  in the  $\text{OLE}_n$ -call. At a high level, we enforce the malicious receiver to use “small” input  $x$  by asking it to provide  $x_p, y'_p \in \mathbb{Z}_p$ , for large enough prime  $p$ , and the receiver verifies that  $a \cdot x'_p + b' = y'_p \bmod p$ . An honest receiver, using  $x \in \mathbb{Z}_q$ , is guaranteed to pass the test. For  $x$ 's not in  $\mathbb{Z}_q$ , we show that for some  $x$ 's, the probability that a malicious receiver passes the test is negligible. For the other  $x$ 's, for which it might pass the test, we prove that (although they are not in  $[q]$ ) there is a way to simulate the receiver's actions. The protocol is formally defined below.

**Protocol 4.10** (Malicious-receiver OLE).Parameters:  $1^{\kappa_s}$  and  $p, q, n \in \mathbb{N}$ .Derived parameters: Let  $s_x \leftarrow qp, r_d \leftarrow 2^{\kappa_s+8}, s_a \leftarrow \max\{\max\{q, p\} \cdot r_d \cdot 2^{2\kappa_s+8}, p^2 \cdot 2^{\kappa_s+2}\}$  and  $s_b \leftarrow n - s_x s_a - 1$ .Oracle:  $\text{OLE}_n$ . $S$ 's input:  $a, b \in \mathbb{Z}_q$ . $R$ 's input:  $x \in \mathbb{Z}_q$ .

Operation:

- (1)  $S$ : Sample  $a' \xleftarrow{R} (s_a)$  and  $b' \xleftarrow{R} (s_b)$ .
- (2)  $R$ : Let  $x' \leftarrow \text{crt}((x, q), (0, p))$ .<sup>a</sup>
- (3) The parties jointly call  $\text{OLE}_n$ ,  $S$  using input  $(a', b')$  and  $R$  using input  $x'$ . Let  $y'$  denote  $R$ 's output.
- (4)  $R$ : Send  $y'_p \leftarrow y' \bmod p$  to  $S$ .
- (5)  $S$ :
  - (a) Abort if  $y'_p \neq b' \bmod p$ .
  - (b) Send  $(\delta_a, \delta_b) \leftarrow (a - a', b - b') \bmod q$  to  $R$ .

Output:  $R$ : Output  $y \leftarrow y' + \delta_b + \delta_a \cdot x \bmod q$ .<sup>a</sup>Hence,  $x' = x \bmod q$  and  $x' = 0 \bmod p$ .

Note that, unlike Protocol 4.7, Protocol 4.10 is (also) parameterized by an integer  $p$  (which will later be set to be relatively prime to

$n$  and  $q$ ), and that the value of  $a$  used as inputs to  $\text{OLE}_n$  is randomized. As stated, Protocol 4.10 has three rounds. In the full version,<sup>17</sup> we present a variant of the protocol that has only a single round (two rounds, when taking into account that the instantiation of the  $\text{OLE}_n$  protocol takes two rounds).

**THEOREM 4.11 (SECURITY OF PROTOCOL 4.10).** *Let  $\kappa_s, q, p, n \in \mathbb{N}$ , and define  $s_x, s_a$  as in Protocol 4.10. Assume  $p, q, n$  are relatively prime to each other,  $p \geq 2^{\kappa_s+2}$ , and  $n \geq 2^{\kappa_s+3} \cdot s_x \cdot s_a$ , then Protocol 4.10 UC-realize  $\text{OLE}_q$  in the  $\text{OLE}_n$ -hybrid model, against semi-honest senders, with statistical security  $2^{-\kappa_s}$ .*

**PROOF.** Correctness is proved in Claim 4.12, security against semi-honest senders is proved in Claim 4.13, and security against malicious receivers is proved in Claim 4.15.  $\square$

In the following, we fix the parameters  $\kappa_s, q, p, n$  that match the requirements of the theorem, and let  $r_d, s_x, s_a, s_b$  be as derived in Protocol 4.10.

**CLAIM 4.12 (CORRECTNESS).** *Protocol 4.10 is perfectly correct.*

#### 4.3.1 Semi-Honest Senders.

**CLAIM 4.13 (SEMI-HONEST SENDERS).** *There exists a simulator in the  $\text{OLE}_q$ -hybrid model that perfectly emulates the execution of  $S$  in the  $\text{OLE}_n$ -hybrid model.*

**PROOF.** The simulator  $\text{Sim}$  for the semi-honest sender  $S$  is defined as follows:

##### Algorithm 4.14 ( $\text{Sim}$ ).

Oracle:  $\text{OLE}_q$ .

Input:  $(a, b) \in \mathbb{Z}_q^2$ .

Operation:

- (1) Call  $\text{OLE}_q$  with input  $(a, b)$ .
- (2) Start emulating  $S$ . Let  $(a', b')$  denote the input provided by  $S$  to  $\text{OLE}_n$ .
- (3) Send  $y'_p \leftarrow b' \bmod p$  to  $S$  as the message of  $R$  in Step 4.

Fix the input  $(a, b)$  of  $\text{Sim}$  and the random input  $(a', b')$   $S$  provides to  $\text{OLE}_n$ . In the real execution,  $y'_p = b' \bmod p$ , which is exactly its value  $y'_p$  in the simulated execution (given the same view  $a', b'$ ). Thus, the two views are identically distributed.  $\square$

#### 4.3.2 Malicious Receivers.

**CLAIM 4.15 (SECURITY AGAINST MALICIOUS RECEIVERS).** *For any PPT receiver for Protocol 4.10 in the  $\text{OLE}_n$ -hybrid model, there exists a PPT non-rewinding simulator in the  $\text{OLE}_q$ -hybrid model, such that for any input of the honest sender, the statistical distance between the real and simulated executions is at most  $2^{-\kappa_s}$ .*

To prove Claim 4.15, we distinguish between two types of (malicious) receivers. Receivers of the first type use “small” inputs to the  $\text{OLE}_n$ -call (the exact definition of “small” will be provided soon). We show that while the honest sender might not catch such receivers, they do not pose a danger to the protocol’s security. We complete the picture by proving that, with high probability, the sender aborts when interacting with receivers that use “large” inputs to the  $\text{OLE}_n$ -call.

<sup>17</sup>The full version is available at <http://eprint.iacr.org>

The above is formalized by providing a simulator for the variant of Protocol 4.10 resulting from replacing the functionality  $\text{OLE}_n$  with its variant  $\widetilde{\text{OLE}}_n$  that aborts on large receiver’s inputs, and leaks some additional information needed for the simulation in case it does not abort. Let  $\widetilde{\text{gcd}}$  be the poly-time computable function guaranteed by Lemma 3.8 (i.e., Thue’s lemma).

##### Functionality 4.16 ( $\widetilde{\text{OLE}}_n$ : Leaky OLE).

Parameters:  $n \in \mathbb{N}$ . Let  $r_c \leftarrow r_d \cdot n/s_a$ .

$S$ ’s input:  $(a, b) \in \mathbb{Z}_n^2$ .

$R$ ’s input:  $x \in \mathbb{Z}_n$ .

Operation:

- (1) Let  $(c, d) \leftarrow \widetilde{\text{gcd}}(n, x, r_d, r_c)$ .
- (2) Abort if  $(c, d) = \perp$ .
- (3) Let  $a_d \leftarrow a \bmod d$  and  $y \leftarrow c \cdot (a - a_d)/d + b$ .

Output: Send  $(a_d, y)$  to  $R$ .

**DEFINITION 4.17 (PROTOCOL  $\widetilde{\Pi}$ ).** *Let  $\widetilde{\Pi} = (\widetilde{S}, \widetilde{R})$  be the variant of Protocol 4.10 in which the call to  $\text{OLE}_n$  is replaced with a call to  $\widetilde{\text{OLE}}_n$ .<sup>18</sup>*

In Claim 4.18, we show that the leakage of  $\widetilde{\text{OLE}}_n$  on  $x$  is not harmful, and the call to  $\widetilde{\text{OLE}}_n$  done in  $\widetilde{\Pi}$  can be simulated using a call to  $\text{OLE}_q$ . In Claim 4.21, we complete the picture by showing that (for malicious receivers) the call to  $\text{OLE}_n$  done in Protocol 4.10 can be simulated using a call to  $\widetilde{\text{OLE}}_n$ .

##### 4.3.2.1 Emulating access to $\widetilde{\text{OLE}}_n$ using $\text{OLE}_q$ .

**CLAIM 4.18 ( $\widetilde{\text{OLE}}_n$  TO  $\text{OLE}_q$ ).** *For any PPT receiver  $\widetilde{R}^*$  for  $\widetilde{\Pi}$  in the  $\widetilde{\text{OLE}}_n$ -hybrid model, there exists a PPT non-rewind simulator  $\text{Sim}$  in the  $\text{OLE}_q$ -hybrid model, such that the statistical distance between the real and simulated executions is at most  $2^{-\kappa_s-1}$ .*

**PROOF.** Fix a malicious and without loss of generality deterministic receiver  $\widetilde{R}^*$ , and consider the following simulator  $\text{Sim}$ :

##### Algorithm 4.19 ( $\text{Sim}$ ).

Oracle:  $\text{OLE}_q$ .

Operation:

- (1) Emulate a random execution of  $(S(0, 0), \widetilde{R}^*)$  until the last step (including the call to  $\widetilde{\text{OLE}}_n$ ). Abort if the emulation does. Let  $x'$  be the input  $\widetilde{R}^*$  provided to  $\widetilde{\text{OLE}}_n$ . Let  $(a'_d, y')$  denote the value  $\widetilde{\text{OLE}}_n$  sent back to  $\widetilde{R}^*$ .
- (2) Let  $(c, d) \leftarrow \widetilde{\text{gcd}}(n, x', r_d, r_c)$ .
- (3) Call  $\text{OLE}_q$  with input  $x \leftarrow c \cdot d^{-1} \bmod q$ .<sup>a</sup> Let  $y$  denote the returned output.
- (4) Sample  $\delta_a \xleftarrow{R} \mathbb{Z}_q$  and set  $\delta_b \leftarrow y - y' + cd^{-1}(\delta_a + a'_d) \bmod q$ .  
Send  $(\delta_a, \delta_b)$  to  $\widetilde{R}^*$  as the message  $S$  sends in the last step of the protocol.

<sup>a</sup> $d^{-1} \bmod q$  exists since  $d < q$  (and thus  $\text{gcd}(d, q) = 1$ ).

<sup>18</sup>For honest  $R$ , the protocol is not well defined ( $R$  expect a single output from its call to  $\text{OLE}_n$ ). This is not an issue, however, since we only use this protocol with malicious receivers that are aware of the change in the hybrid model.



In the following, we refer to the execution induced by  $\tilde{R}^*$  in  $\widetilde{\text{OLE}}_n$ -hybrid model as the *real execution*, and the execution induced by  $\text{Sim}$  in the  $\text{OLE}_q$ -hybrid model as the *simulated execution*. Fix the input  $(a, b)$  of  $S$ . Since  $\tilde{R}^*$  is deterministic, the input  $x'$  it provides to  $\text{OLE}_n$  is also fixed. In the following, we omit these fixed values from  $\tilde{R}^*$ 's view. We assume without loss of generality that the call to  $\text{OLE}_n$  does not abort and thus  $d \cdot a' = c \bmod n$ ,  $d \in [r_d]$  and  $|c| < r_d n / s_a$ , for  $c, d$  being as computed by  $\widetilde{\text{OLE}}_n$ .

By construction, the partial views  $(y', a'_d)$  of  $\tilde{R}^*$  are identically distributed in both executions. We next argue that in the real execution, with high probability over  $(y', a'_d)$ , the distribution of  $\delta_a$  conditioned on  $(y', a'_d)$  is statistically close to uniform over  $\mathbb{Z}_q$ , which by construction corresponds to its distribution in the simulated execution. Let  $(A', B')$  be the input  $S$  used in the call to  $\widetilde{\text{OLE}}_n$ . Since,  $\delta_a = A' - a \bmod q$ , we prove this part by bounding the distance of  $A' \bmod q$  from uniform, conditioned on  $(y', a'_d)$ . In the following, we assume for concreteness that  $c \neq 0$ ; the proof for  $c = 0$  follows similar lines. Let  $A_0 \leftarrow (A' - A'_d)/d$ , and observe that conditioned on  $a'_d$ , the value of  $A_0$  is  $(d/s_a < 2^{-\kappa_s-4})$ -close to uniform over  $[\lfloor s_a/d \rfloor]$ . Since  $B'$  is uniform over  $(s_b)$  and since  $y' = A'x' + B \bmod n$ , the value of  $A_0$  conditioned on  $v'$  is  $2^{-\kappa_s-4}$ -close to uniform over

$$\mathcal{A} := \begin{cases} [\lceil (y' - s_b)/c \rceil, \lfloor y'/c \rfloor] \cap [\lfloor s_a/d \rfloor] & c > 0, \\ [\lceil y'/c \rceil, \lfloor (y' - s_b)/c \rfloor] \cap [\lfloor s_a/d \rfloor] & c < 0. \end{cases}$$

Note that  $\mathcal{A}$  is a determined by  $(a'_d, y')$ . We use the following claim (proven below).

$$\text{CLAIM 4.20. } \Pr_{(a'_d, y')} [|\mathcal{A}| < q \cdot 2^{\kappa_s+4}] \leq 2^{-\kappa_s-2}.$$

PROOF. See full version.  $\square$

In the following, we assume  $\mathcal{A}$  is large, i.e.,  $|\mathcal{A}| \geq qp \cdot 2^{\kappa_s+4}$ , and take care of the complementary case later. By Claim 4.20 and Fact 3.3, the value of  $A_0 \bmod q$  conditioned on  $(y', a'_d)$  is  $q/|\mathcal{A}| + 2^{-\kappa_s-4} \leq 2^{-\kappa_s-3}$  close to uniform over  $\mathbb{Z}_q$ . Since conditioned on  $(y', a'_d)$ , there is an injective mapping between  $A_0 \bmod q$  and  $A' \bmod q$ , the value of  $A' \bmod q$  conditioned on  $(y', a'_d)$  is (also)  $2^{-\kappa_s-3}$ -close to uniform over  $\mathbb{Z}_q$ .

We next analyze the distribution of  $A' \bmod q$  conditioned on  $(a'_d, y', y'_p, e)$  for  $e \leftarrow y'_p \stackrel{?}{=} B' \bmod p - R^*$ 's full view without  $(\delta_a, \delta_b)$ . We prove that there exists a function  $t$  such that

$$\Pr[e \neq t(a'_d, y', y'_p)] \leq 2^{-\kappa_s-3},$$

where the probability is over  $(A', B')$  conditioned on  $(a'_d, y')$ . Indeed, if  $x' = 0 \bmod p$ , then  $e = 1$  iff  $y'_p = y' \bmod p$ . If  $x' \neq 0 \bmod p$  then, since  $B' = y' - A' \cdot x' \bmod p$ , it holds that  $e = 1$  iff  $A' = (x')^{-1}(y' - y'_p) \bmod p$ . Applying the same argument for proving the closeness to uniformity of  $A' \bmod q$ , yields that  $A' \bmod p$  is  $2^{-\kappa_s-3}$  close to uniform given  $(a'_d, y')$ . Thus, the test passes in this case with probability at most  $2^{-\kappa_s-3}$ . Hence, a simply hybrid argument yields that  $A' \bmod q$  is  $(2 \cdot 2^{-\kappa_s-3} = 2^{-\kappa_s-2})$ -close to uniform over  $\mathbb{Z}_q$  conditioned on  $(a'_d, y', y'_p, e)$ . Taking into account the probability that  $\mathcal{A}$  is not large, we deduce that the value of  $(a'_d, y', y'_p, e, \delta_a)$  in the real and simulated executions are  $2^{-\kappa_s-1}$ -close.

We conclude the proof by showing that  $\delta_b$ , the missing part of the view, is a deterministic function of  $(v', \delta_a)$  (in both executions), and thus can be ignored. Indeed, in the real execution, recalling that  $a'_d = A \bmod d$ ,  $y' = c(A' - a'_d)/d + B'$ , and letting  $y \leftarrow acd^{-1} + b \bmod q$ , it holds that

$$\begin{aligned} \delta_b &= b - B' \\ &= b - (y' + cd^{-1}(A' - a'_d)) \\ &= y - acd^{-1} - (y' + cd^{-1}(A' - a'_d)) \\ &= y - y' + cd^{-1}(a'_d + a - A') \\ &= y - y' + cd^{-1}(\delta_a + a'_d) \bmod q, \end{aligned}$$

which is the value of  $\delta_b$  in the simulated execution.  $\square$

**4.3.2.2 Emulating access to  $\text{OLE}_n$  using  $\widetilde{\text{OLE}}_n$ .** To conclude the proof of the malicious receiver part, we show how to simulate the interaction of a malicious receiver  $R^*$  with access to  $\text{OLE}_n$  with  $S$  by a malicious receiver with access to  $\widetilde{\text{OLE}}_n$ . Specifically, we prove the following result

**CLAIM 4.21 ( $\text{OLE}_n$  TO  $\widetilde{\text{OLE}}_n$ ).** *For any malicious receiver  $R^*$  for Protocol 4.10 in the  $\text{OLE}_n$ -hybrid model, there exists a malicious receiver  $\tilde{R}^*$  for  $\tilde{\Pi}$  in the  $\widetilde{\text{OLE}}_n$ -hybrid model, such that the statistical distance between the real and simulated executions is at most  $2^{-\kappa_s-1}$ .*

Fix a malicious, without loss of generality deterministic,  $R^*$  for Protocol 4.10. We define a malicious receiver  $\tilde{R}^*$  for interacting with the honest sender  $\tilde{S}$  in  $\tilde{\Pi}$  as follows.

**Algorithm 4.22 ( $\tilde{R}^*$ ).**

Oracle:  $\widetilde{\text{OLE}}_n$ .

Operation: Interact with  $\tilde{S}$  as follows:

- (1) Start emulating  $R^*$ . Let  $x' \in \{0, \dots, n-1\}$  denote the input provided by  $R^*$  to  $\text{OLE}_n$ .
- (2) Call  $\text{OLE}_n$  with input  $x'$ . Let  $(a'_d, y')$  be the returned output.
- (3) Forward  $y' + a'_d \cdot x' \bmod n$  to  $R^*$  as the answer of the call to  $\text{OLE}_n$ .
- (4) Interact with the sender till the end of the protocol by forwarding its messages to  $R^*$  and sending  $R^*$ 's responses back to it.

The heart of the proof of Claim 4.21 is in the following lemma, proved in the full version.

**DEFINITION 4.23.** *For  $n, \ell, s \in \mathbb{N}$ , let  $\mathcal{X}_{n,s,\ell}$  be defined as*

$$\{x \in \mathbb{N} : \exists c \in \mathbb{Z}, d \in [\ell] \text{ s.t. } |c| \leq dn \cdot \ell/s \wedge d \cdot x = c \bmod n\}.$$

**LEMMA 4.24 (UNPREDICTABILITY OF MODULAR MULTIPLICATION).** *Let  $\ell, n, s \in \mathbb{N}$ , let  $x \in \mathbb{N} \setminus \mathcal{X}_{n,s,\ell}$ , let  $p \in \mathcal{P} \cap (2\ell, \infty)$  be such that  $p \nmid n$  and  $s/p > 16\ell^2$ , and let  $V \xleftarrow{R} [s]$  and  $W \xleftarrow{R} \mathbb{Z}_n$ . Then,*

$$\Pr(W \bmod p \mid x \cdot V + W \bmod n) \leq 42 \cdot \ell^{-1} + \frac{p^2}{s}.$$

**PROOF OF CLAIM 4.21.** In the following, we refer to the execution induced by  $R^*$  in  $\text{OLE}_n$ -hybrid model as the real execution, and the execution induced by  $\tilde{R}^*$  in the  $\widetilde{\text{OLE}}_n$ -hybrid model as the

simulated execution. Let  $x'$  be the value provided by  $R^*$  to  $\text{OLE}_n$ . By construction,  $\text{OLE}_n$  aborts on receiver's input  $x'$  iff  $x' \notin \mathcal{X}_{n,s_a,r_d}$ .

Let  $A' \xleftarrow{R} (s_a)$  and  $B' \xleftarrow{R} (s_b)$  denote the sender input in the real execution, and let  $Y' \leftarrow A' \cdot x' + B' \bmod n$  be the value  $R^*$  received from  $\text{OLE}_n$ . Note that the sender's test in Step 5a passes iff  $R^*$  sends  $y'_p$  such that  $y'_p = B' \bmod p$ . Given this notation, we have to upperbound  $\alpha \leftarrow P_\infty(B' \bmod p \mid Y')$ . Assume  $B'$  is uniform over  $\mathbb{Z}_n$  (and not over  $(s_b)$ ), then by Lemma 4.24 (letting  $s \leftarrow s_a$ ,  $x \leftarrow x'$ ,  $V \leftarrow A'$  and  $W \leftarrow B'$ ),

$$\alpha' \leftarrow P_\infty(B' \bmod p \mid Y') \leq 42 \cdot 2^{-r_d} + p^2/s_a \leq 2^{-\kappa_s-2} + 2^{-\kappa_s-3}.$$

When taking into account the real distribution of  $B'$ , we deduce that the probability that the test in Step 5a passes is bounded by  $\alpha \leq \alpha' + s_b/n \leq \alpha' + 2^{-\kappa_s-3} \leq 2^{-\kappa_s-1}$ .<sup>19</sup>

We conclude the proof by showing that the executions are *identical* if  $x' \in \mathcal{X}_{n,s_a,r_d}$ . Since in the simulated execution, the call to  $\text{OLE}_n$  does not abort on such  $x'$ ; the only potential difference might be in the answer of the call to  $\text{OLE}_n$ . Fix the random  $(a', b')$  sampled by  $S$ . In the simulated execution, the output of  $\text{OLE}_n$  is set to  $y' + a'_d \cdot x \bmod n$ , for  $a'_d \leftarrow a' \bmod d$  and  $y' \leftarrow (a' - a'_d) \cdot c/d + b'$ , for some  $c, d \in \mathbb{Z}$  such that  $d \cdot x' = c \bmod n$ . Compute

$$\begin{aligned} y' + a'_d \cdot x' &= (a' - a'_d) \cdot c/d + b' + a'_d \cdot x' \\ &= (a' - a'_d) \cdot d \cdot x' / d + b' + a'_d \cdot x' \\ &= (a' - a'_d) \cdot x' + b' + a'_d \cdot x' \\ &= a'x' + b' \bmod n, \end{aligned}$$

which is the output of  $\text{OLE}_n$  in the real execution. Thus, the two executions are identical for such  $x'$ .  $\square$

#### 4.3.2.3 Putting it together.

PROOF OF CLAIM 4.15. Immediately follows from Claims 4.18 and 4.21.  $\square$

**4.3.3 Vector OLE.** It is easy to extend the malicious-receiver OLE from Protocol 4.10 into a malicious-receiver  $\text{VOLE}_{q,m}$ , for any  $q, m \in \mathbb{N}$ , in the  $\text{VOLE}_{n,m}$ -hybrid model:

- (1) In Step 1,  $S$  samples  $a' \xleftarrow{R} (s_a)^m, b' \xleftarrow{R} (s_b)^m$ .
- (2) In Step 3, the parties call  $\text{VOLE}_{n,m}$ ,  $S$  using  $(a', b')$  and  $R$  using  $x$ .
- (3) The protocol continues, in parallel, for each of the  $m$  coordinates of the output.<sup>20</sup>

Since  $\text{VOLE}_{n,m}$  can be implemented using a straightforward variant of Protocol 4.1, with communication cost that is  $m$  times that of the  $\text{OLE}_n$  protocol, the communication cost of the above VOLE protocol is  $m$  times that of Protocol 4.10. It is also easy to verify that the resulting protocol, with the same choice of parameters as in Theorem 4.11, is secure against a semi-honest sender with security  $m \cdot 2^{-\kappa_s}$ .

<sup>19</sup>Indeed, let  $B''$  be the uniform distribution over  $\mathbb{Z}_n$  and  $Y'' \leftarrow A'x' + B''$ . The statistical distance between  $(B', Y')$  and  $(B'', Y'')$  is just the distance between  $B'$  and  $B''$  ( $Y'$  and  $Y''$  are the same random function of  $B'$  and  $B''$ , respectively), which is  $s_b/n$ . Since the success of predicting  $B'$  given  $Y'$  (by an arbitrary algorithm) is a randomized function of  $(B', Y')$ , by the data-processing property of statistical distance, one cannot do it much better (better than  $s_b/n$ ) than in guessing  $B''$  given  $Y''$ .

<sup>20</sup>A slight save can be achieved by batching the  $y'_p$ 's into a single element.

**Acknowledgments.** We thank Nico Döttling for helpful discussions on general constructions of VOLE from OLE, Hugo Krawczyk and Yibin Yang for helpful discussions on the OPRF application, Amir Shpilka for helpful discussions on the number-theoretic conjecture, and the anonymous referees for their suggestions. We also thank Michael Adjedj for running experimental benchmarks and for discussions on concrete performance. I. Haitner was supported by ISF grant 836/23. Y. Ishai was supported by ISF grant 3527/24, BSF grant 2022370, and ISF-NSFC grant 3127/23. J. Doerner was supported by the Azrieli Foundation and the Brown University Data Science Institute.

## References

- [1] Michael Adjedj, Constantin Blokh, Geoffroy Couteau, Antoine Joux, and Nikolaos Makriyannis. 2024. Two-Round 2PC ECDSA at the Cost of 1 OLE. *IACR Cryptol. ePrint Arch.* (2024), 1950. <https://eprint.iacr.org/2024/1950>
- [2] Gilad Asharov, Yehuda Lindell, Thomas Schneider, and Michael Zohner. 2015. More Efficient Oblivious Transfer Extensions with Security for Malicious Adversaries. In *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I*.
- [3] Carsten Baum, Daniel Escudero, Alberto Pedrouzo-Ulloa, Peter Scholl, and Juan Ramón Troncoso-Pastoriza. 2020. Efficient Protocols for Oblivious Linear Function Evaluation from Ring-LWE. In *Security and Cryptography for Networks (SCN)*. 130–149.
- [4] Donald Beaver. 1991. Efficient Multiparty Protocols Using Circuit Randomization. In *Advances in Cryptology - CRYPTO '91, 11th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1991, Proceedings (Lecture Notes in Computer Science)*, Joan Feigenbaum (Ed.).
- [5] Donald Beaver. 1996. Correlated Pseudorandomness and the Complexity of Private Computations. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, Philadelphia, Pennsylvania, USA, May 22-24, 1996*.
- [6] Maxime Bombar, Dung Bui, Geoffroy Couteau, Alain Couvreur, Clément Ducros, and Sacha Servan-Schreiber. 2024. FOLEAGE:  $\mathbb{F}_4$ -OLE-Based Multi-party Computation for Boolean Circuits. In *Advances in Cryptology - ASIACRYPT 2024 - 30th International Conference on the Theory and Application of Cryptology and Information Security, Kolkata, India, December 9-13, 2024, Proceedings, Part VI (Lecture Notes in Computer Science, Vol. 15489)*, Kai-Min Chung and Yu Sasaki (Eds.). Springer, 69–101. [doi:10.1007/978-981-96-0938-3\\_3](https://doi.org/10.1007/978-981-96-0938-3_3)
- [7] Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, and Peter Scholl. 2019. Efficient Pseudorandom Correlation Generators: Silent OT Extension and More. In *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part III*.
- [8] Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, and Peter Scholl. 2020. Correlated Pseudorandom Functions from Variable-Density LPN. In *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*.
- [9] Dung Bui, Geoffroy Couteau, Pierre Meyer, Alain Passelègue, and Mahshid Riahiinia. 2024. Fast Public-Key Silent OT and More from Constrained Naor-Reingold. In *Advances in Cryptology - EUROCRYPT 2024 - 43rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zurich, Switzerland, May 26-30, 2024, Proceedings, Part VI*.
- [10] Ran Canetti. 2000. Security and Composition of Multiparty Cryptographic Protocols. *Journal of Cryptology* 13, 1 (2000), 143–202.
- [11] Ran Canetti. 2001. Universally Composable Security: A New Paradigm for Cryptographic Protocols. In *Annual Symposium on Foundations of Computer Science (FOCS)*. 136–145.
- [12] Ran Canetti, Rosario Gennaro, Steven Goldfeder, Nikolaos Makriyannis, and Udi Peled. 2020. UC Non-Interactive, Proactive, Threshold ECDSA with Identifiable Aborts. In *Proceedings of the 27th ACM Conference on Computer and Communications Security (CCS)*.
- [13] Guilhem Castagnos, Dario Catalano, Fabien Laguillaumie, Federico Savasta, and Ida Tucker. 2019. Two-Party ECDSA from Hash Proof Systems and Efficient Instantiations. In *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part III (Lecture Notes in Computer Science, Vol. 11694)*, Alexandra Boldyreva and Daniele Micciancio (Eds.). Springer, 191–221. [doi:10.1007/978-3-030-26954-8\\_7](https://doi.org/10.1007/978-3-030-26954-8_7)
- [14] Guilhem Castagnos, Dario Catalano, Fabien Laguillaumie, Federico Savasta, and Ida Tucker. 2020. Bandwidth-Efficient Threshold EC-DSA. In *Proceedings of the 23rd International Conference on the Theory and Practice of Public-Key Cryptography (PKC), part II*.

- [15] Geoffroy Couteau, Lalita Devadas, Srinivas Devadas, Alexander Koch, and Sacha Servan-Schreiber. 2024. QuietOT: Lightweight Oblivious Transfer with a Public-Key Setup. In *Advances in Cryptology - ASIACRYPT 2024 - 30th International Conference on the Theory and Application of Cryptology and Information Security, Kolkata, India, December 9–13, 2024, Proceedings, Part II (Lecture Notes in Computer Science, Vol. 15485)*, Kai-Min Chung and Yu Sasaki (Eds.). Springer, 197–231. doi:10.1007/978-981-96-0888-1\_7
- [16] Ivan Damgard. 1988. On the Randomness of Legendre and Jacobi Sequences. In *Annual International Cryptology Conference (CRYPTO)*, Shafi Goldwasser (Ed.), 163–172.
- [17] Leo de Castro, Carmit Hazay, Yuval Ishai, Vinod Vaikuntanathan, and Muthuramakrishnan Venkatasubramanian. 2022. Asymptotically Quasi-Optimal Cryptography. In *Advances in Cryptology - EUROCRYPT 2022 - 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, May 30 – June 3, 2022, Proceedings, Part I (Lecture Notes in Computer Science, Vol. 13275)*, Orr Dunkelman and Stefan Dziembowski (Eds.). Springer, 303–334. doi:10.1007/978-3-031-06944-4\_11
- [18] Leo de Castro, Chiraag Juvekar, and Vinod Vaikuntanathan. 2021. Fast Vector Oblivious Linear Evaluation from Ring Learning with Errors. In *WAHC '21: Proceedings of the 9th on Workshop on Encrypted Computing & Applied Homomorphic Cryptography, Virtual Event, Korea, 15 November 2021*. WAHC@ACM, 29–41. doi:10.1145/3474366.3486928
- [19] Leo de Castro, Duhyeon Kim, Miran Kim, Keewoo Lee, Seonhong Min, and Yongsoo Song. 2024. More Efficient Lattice-based OLE from Circuit-private Linear HE with Polynomial Overhead. Cryptology ePrint Archive, Paper 2024/1534. <https://eprint.iacr.org/2024/1534>
- [20] Jack Doerner, Yashvanth Kondi, Eysa Lee, and abhi shelat. 2018. Secure Two-party Threshold ECDSA from ECDSA Assumptions. In *Proceedings of the 39th IEEE Symposium on Security and Privacy (S&P)*.
- [21] Jack Doerner, Yashvanth Kondi, Eysa Lee, and abhi shelat. 2019. Threshold ECDSA from ECDSA Assumptions: The Multiparty Case. In *2019 IEEE Symposium on Security and Privacy, SP 2019, San Francisco, CA, USA, May 19–23, 2019*. IEEE, 1051–1066. doi:10.1109/SP.2019.00024
- [22] Jack Doerner, Yashvanth Kondi, Eysa Lee, and abhi shelat. 2024. Threshold ECDSA in Three Rounds. In *Proceedings of the 45th IEEE Symposium on Security and Privacy (S&P)*.
- [23] Jack Doerner, Yashvanth Kondi, Eysa Lee, abhi shelat, and LaKyah Tyner. 2023. Threshold BBS+ Signatures for Distributed Anonymous Credential Issuance. In *Proceedings of the 44th IEEE Symposium on Security and Privacy (S&P)*.
- [24] Nico Döttling, Daniel Kraschewski, and Jörn Müller-Quade. 2012. Statistically Secure Linear-Rate Dimension Extension for Oblivious Affine Function Evaluation. In *Information Theoretic Security - 6th International Conference, ICITS 2012, Montreal, QC, Canada, August 15–17, 2012. Proceedings (Lecture Notes in Computer Science, Vol. 7412)*, Adam D. Smith (Ed.). Springer, 111–128. doi:10.1007/978-3-642-32284-6\_7
- [25] Shimon Even, Oded Goldreich, and Abraham Lempel. 1985. A Randomized Protocol for Signing Contracts. *Commun. ACM* 28, 6 (1985), 637–647.
- [26] Rosario Gennaro and Steven Goldfeder. 2018. Fast Multiparty Threshold ECDSA with Fast Trustless Setup. In *Proceedings of the 25th ACM Conference on Computer and Communications Security (CCS)*.
- [27] Satrajit Ghosh, Jesper Buus Nielsen, and Tobias Nilges. 2017. Maliciously Secure Oblivious Linear Function Evaluation with Constant Overhead. In *Annual International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT)*, 629–659.
- [28] Niv Gilboa. 1999. Two Party RSA Key Generation. In *Annual International Cryptology Conference (CRYPTO)*, 116–129.
- [29] Oded Goldreich, Silvio Micali, and Avi Wigderson. 1987. How to Play any Mental Game or A Completeness Theorem for Protocols with Honest Majority. In *Annual ACM Symposium on Theory of Computing (STOC)*, 218–229.
- [30] Iftach Haitner, Yehuda Lindell, Ariel Nof, and Samuel Ranellucci. 2023. Fast Secure Multiparty ECDSA with Practical Distributed Key Generation and Applications to Cryptocurrency Custody. Cryptology ePrint Archive, Paper 2018/987, Version 20230529:135032. <https://eprint.iacr.org/archive/2018/987/20230529:135032>
- [31] Iftach Haitner, Nikolaos Makriyannis, Samuel Ranellucci, and Eliad Tsfadia. 2022. Highly Efficient OT-Based Multiplication Protocols. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT)*, 180–209.
- [32] David Harvey and Joris van der Hoeven. 2022. Polynomial Multiplication over Finite Fields in Time  $\tilde{O}(n \log n)$ . *J. ACM* 69, 2 (2022), 12:1–12:40. doi:10.1145/3505584
- [33] Carmit Hazay, Yuval Ishai, Antonio Marcedone, and Muthuramakrishnan Venkatasubramanian. 2019. LevioSA: Lightweight Secure Arithmetic Computation. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS 2019, London, UK, November 11–15, 2019*, Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz (Eds.). ACM, 327–344. doi:10.1145/3319535.3354258
- [34] Carmit Hazay and Yehuda Lindell. 2009. Efficient Oblivious Polynomial Evaluation with Simulation-Based Security. Cryptology ePrint Archive, Paper 2009/459. <https://eprint.iacr.org/2009/459>
- [35] Yuval Ishai, Joe Kilian, Kobbi Nissim, and Erez Petrank. 2003. Extending Oblivious Transfers Efficiently. In *Advances in Cryptology - CRYPTO 2003, 23rd Annual International Cryptology Conference, Santa Barbara, California, USA, August 17–21, 2003, Proceedings*.
- [36] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, Manoj Prabhakaran, and Amit Sahai. 2011. Efficient Non-interactive Secure Computation. In *Advances in Cryptology - EUROCRYPT 2011 - 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15–19, 2011. Proceedings (Lecture Notes in Computer Science, Vol. 6632)*, Kenneth G. Paterson (Ed.). Springer, 406–425. doi:10.1007/978-3-642-20465-4\_23
- [37] Yuval Ishai, Manoj Prabhakaran, and Amit Sahai. 2008. Founding Cryptography on Oblivious Transfer - Efficiently. In *Advances in Cryptology - CRYPTO 2008, 28th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 17–21, 2008. Proceedings (Lecture Notes in Computer Science, Vol. 5157)*, David A. Wagner (Ed.). Springer, 572–591. doi:10.1007/978-3-540-85174-5\_32
- [38] Yuval Ishai, Manoj Prabhakaran, and Amit Sahai. 2009. Secure Arithmetic Computation with No Honest Majority. In *Theory of Cryptography (TCC)*, 294–314.
- [39] Chiraag Juvekar, Vinod Vaikuntanathan, and Anantha P. Chandrakasan. 2018. GAZELLE: A Low Latency Framework for Secure Neural Network Inference. In *27th USENIX Security Symposium, USENIX Security 2018, Baltimore, MD, USA, August 15–17, 2018*, William Enck and Adrienne Porter Felt (Eds.). USENIX Association, 1651–1669. <https://www.usenix.org/conference/usenixsecurity18/presentation/juvekar>
- [40] Marcel Keller, Emmanuela Orsini, and Peter Scholl. 2015. Actively Secure OT Extension with Optimal Overhead. In *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16–20, 2015, Proceedings, Part I*.
- [41] Marcel Keller, Emmanuela Orsini, and Peter Scholl. 2016. MASCOT: Faster Malicious Arithmetic Secure Computation with Oblivious Transfer. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 830–842.
- [42] Joe Kilian. 1988. Founding Cryptography on Oblivious Transfer. In *Annual ACM Symposium on Theory of Computing (STOC)*, 20–31.
- [43] Vladimir Kolesnikov, Stanislav Peceny, Srinivasan Raghuraman, and Peter Rindal. 2025. Stationary Syndrome Decoding for Improved PCGs. *IACR Cryptol. ePrint Arch.* (2025), 295. <https://eprint.iacr.org/2025/295>
- [44] Eyal Kushilevitz, Yehuda Lindell, and Tal Rabin. 2010. Information-Theoretically Secure Protocols and Security under Composition. *SIAM J. Comput.* 39, 5 (2010), 2090–2112.
- [45] Zhe Li, Chaoping Xing, Yizhou Yao, and Chen Yuan. 2025. Efficient Pseudorandom Correlation Generators for Any Finite Field. *IACR Cryptol. ePrint Arch.* (2025), 169. <https://eprint.iacr.org/2025/169> To appear in Eurocrypt 2025.
- [46] Yehuda Lindell. 2017. Fast Secure Two-Party ECDSA Signing. In *Advances in Cryptology - CRYPTO 2017, part II*.
- [47] Moni Naor and Benny Pinkas. 2006. Oblivious Polynomial Evaluation. *SIAM J. Comput.* 35, 5 (2006), 1254–1281.
- [48] M. O. Rabin. 1981. How to Exchange Secrets by Oblivious Transfer. TR-81, Harvard.
- [49] Srinivasan Raghuraman, Peter Rindal, and Titouan Tanguy. 2023. Expand-Convolute Codes for Pseudorandom Correlation Generators from LPN. In *Advances in Cryptology - CRYPTO 2023 - 43rd Annual International Cryptology Conference, CRYPTO 2023, Santa Barbara, CA, USA, August 20–24, 2023, Proceedings, Part IV*.
- [50] John H. Reif and Stephen R. Tate. 1990. Optimal Size Integer Division Circuits. *SIAM J. Comput.* 19, 5 (1990), 912–924. doi:10.1137/0219064
- [51] J. Barkley Rosser and Lowell Schoenfeld. 1962. Approximate formulas for some functions of prime numbers. *Illinois Journal of Mathematics* 6, 1 (1962), 64–94.
- [52] Lawrence Roy. 2022. SoftSpokenOT: Quieter OT Extension from Small-Field Silent VOLE in the Minicrypt Model. In *Advances in Cryptology - CRYPTO 2022 - 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15–18, 2022, Proceedings, Part I*.
- [53] Victor Shoup. 2006. *A computational introduction to number theory and algebra*. Cambridge University Press.
- [54] Kang Yang, Chenkai Weng, Xiao Lan, Jiang Zhang, and Xiao Wang. 2020. Ferret: Fast Extension for Correlated OT with Small Communication. In *CCS '20: 2020 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, USA, November 9–13, 2020*, Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna (Eds.). ACM, 1607–1626. doi:10.1145/3372297.3417276
- [55] Yibin Yang, Fabrice Benhamouda, Shai Halevi, Hugo Krawczyk, and Tal Rabin. 2025. Gold OPRF: Post-Quantum Oblivious Power-Residue PRF. In *IEEE Symposium on Security and Privacy, SP 2025, San Francisco, CA, USA, May 12–15, 2025*, Marina Blanton, William Enck, and Cristina Nita-Rotaru (Eds.). IEEE, 259–278. doi:10.1109/SP61157.2025.00116