A Simple Post-Quantum Oblivious Transfer Protocol from Mod-LWR

Shen Dong[1], Hongrui Cui[1], Kaiyi Zhang[1], Kang Yang[2], and Yu Yu[1]

[1] Shanghai Jiao Tong University
[2] State Key Laboratory of Cryptology

Abstract. Oblivious transfer (OT) is a fundamental cryptographic protocol that plays a crucial role in secure multi-party computation (MPC). Most practical OT protocols by, e.g., Naor and Pinkas (SODA'01) or Chou and Orlandi (Latincrypt'15), are based on Diffie-Hellman (DH)-like assumptions and not post-quantum secure. In contrast, many other components of MPC protocols, including garbled circuits and secret sharings, are post-quantum secure. The reliance on non-post-quantum OT protocols presents a significant security bottleneck with the advent of quantum computing. In this paper, we address this issue by constructing a simple, efficient OT protocol based on Saber, a Mod-LWR-based key exchange protocol. We implemented our OT protocol and conducted experiments to evaluate its performance. Our results show that our OT protocol significantly outperforms the state-of-the-art Kyber-based post-quantum OT protocol by Masny and Rindal (CCS'19) in terms of both computation and communication costs. Furthermore, the computation speed of our OT protocol is faster than the best-known DH-based OT protocol by Chou and Orlandi (Latincrypt'15), making it competitive to replace DH-based OT in the high-bandwidth network setting.

Keywords: Post-Quantum Cryptography · Oblivious Transfer

1 Introduction

Oblivious transfer (OT) is a fundamental cryptographic primitive in modern cryptography, as it implies secure multiparty computation [29]. Several flavors of OT exist and have been proven equivalent. One-out-of-two OT is the simplest form, defined as follows: a sender has two input messages m_0 and m_1, and a receiver has a choice bit x. The receiver is supposed to receive m_x and remains unknown about m_{1-x}, while the sender learns nothing about the choice.

Previously, there have been many efficient construction for Oblivious Transfer, such as Naor-Pinkas OT [37], Chou-Orlandi OT [18] etc., quite a lot of which based on Diffie-Hellman-like assumptions. But with the development of quantum computers, the confidentiality and integrity of some classic cryptography system may be compromised. Many other components of secure multi-party computation (MPC) protocols like garbled circuits and secret sharing are either post-quantum secure or information-theoretically secure. This makes OT

become a potential security bottleneck in the era of quantum computing. So it is important to construct some practical post-quantum OT to extend post-quantum secure multiparty protocols, which would be used as a crucial building block for designing post-quantum MPC protocols. The current best candidate is the Masny-Rindal OT [33] built from the Module-LWE-based key exchange protocol Kyber [48]. However, there is still space for more simplicity and speed in practice.

Therefore, in this work, we build an OT from Saber key exchange scheme [19], following the framework of Naor-Pinkas OT [37]. The goals of our construction are simplicity, efficiency, and post-quantum security.

We choose to base our OT protocol on Saber, because it remains secure against quantum computers and is one of the round 3 candidates of the NIST Post-Quantum Cryptography Standardization effort [39]. Furthermore, compared with its Mod-LWE-based competitor Kyber, the underlying Mod-LWR [5] assumption of Saber eliminates randomness in ciphertext generation, so Mod-LWR-based schemes naturally require smaller bandwidth and enable deterministic operations. Moreover, all integer moduli in Saber are powers of 2, contributing to the simplicity and facilitating constant-time implementations. In summary, Saber offers simplicity by design, and its implementation achieves both efficiency and flexibility.

In the Saber key exchange, the sender and receiver use information reconciliation c to help to get the same bits from the inner product of two Mod-LWR samples, in the form of $\left(\mathbf{s}'^T \cdot \lfloor \frac{p}{q}\mathbf{A} \cdot \mathbf{s} \rceil\right)$ and $\left(\mathbf{s}^T \cdot \lfloor \frac{p}{q}\mathbf{A}^T \cdot \mathbf{s}' \rceil\right)$. The reconciliation c contains higher bits of one inner product, which helps to recover the error introduced by rounding and can make the probability of the two keys disagreeing negligible. We tweak this key exchange into an OT protocol as shown in Fig. 1.

Like the Naor-Pinkas construction, the sender shares a seed with the receiver in advance to generate a shared matrix $\mathbf{A}$ and a random polynomial vector $\mathbf{r}$. Then the sender computes an Mod-LWR sample $\lfloor \mathbf{A} \cdot \mathbf{s} \rfloor_{q \to p}$, while the receiver also computes one $\lfloor \mathbf{A}^T \cdot \mathbf{s}' \rfloor_{q \to p}$ symmetrically. The receiver assigns the sample to a vector indexed by the input bit x and assigns the subtraction result with $\mathbf{r}$ to the other. Then, the one indexed by 0 is sent to the sender, who gets two $\mathbf{b}'$ vectors. By the pseudorandomness of the Mod-LWR samples, the sender cannot determine the value of x. The sender computes two vectors' reconciliation bits c_0, c_1, encrypts two messages m_0, m_1 respectively, and sends them with its own Mod-LWR sample $\mathbf{b}$. On the receiver's side, it can derive only one key from $\mathbf{s}', \mathbf{b}$ and decrypts one ciphertext c_x i.e. the one indexed by its input bit x, so the receiver gets exactly the message indexed by x, remaining unaware about the other message.

The correctness of this protocol can be shown by proving that c helps derive shared keys for both sides and checking key derivations. It has been proven in [19] and [7] that the reconciliation c can ensure the key agreement with only negligible failure probability. Intuitively, the distance between v' and v_x is close enough to be corrected by c and a rounding constant h_2.

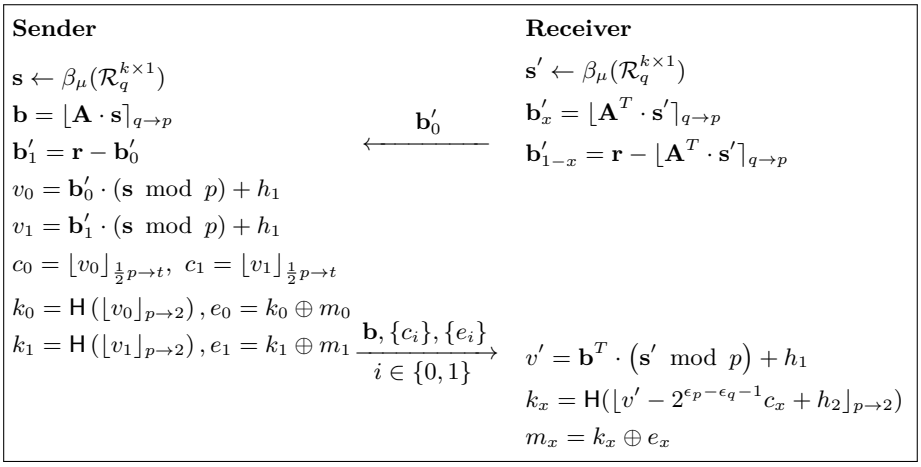

Fig. 1. Our Saber-based Oblivious Transfer protocol, where $\mathbf{A}, \mathbf{r}$ consists of the common reference string, H is a hash function modeled as random oracle, h_1, h_2, p, q are parameters, $x \in \{0,1\}$ is the chosen bit of the receiver, and m_0, m_1 are the messages to be sent by the sender.

For security, we show that by adapting the technique from [17], our protocol, augmented by consistency checking, achieves sender-sided simulation security and input extractability against a corrupted receiver. This notion of security has been shown to be sufficient to achieve UC security when combined with the KOS OT extension protocol [28]. Since the KOS protocol is later shown to have a flaw in its security proof, we adapt the argument in [17] to the state-of-the-art SoftSpoken OT extension protocol and show that the weakened security notion already suffices for SoftSpoken OT [47].

We implement our protocol and conduct experiments using the Saber parameter with 192-bit classical security. Compared with the implementation of Masny-Rindal OT [33] instantiated with Kyber768, our results demonstrate faster running time and smaller communication size. In the LAN setting (local loopback with high bandwidth), our implementation is even faster than classic OT protocols based on the DH-like assumption.

1.1 Related Work

Oblivious transfer (OT) was first proposed by Rabin in [45] and has been proved to be a fundamental component of secure multiparty computation [51,22,29,26]. OT has many variants in terms of functionality, including 1-out-of-2 OT, 1-out-of-N OT [38], and k-out-of-N OT [23]. In this work, we focus on the most fundamental 1-out-of-2 OT functionality.

Since it has been proved that public-key operations are essential for OT [24], Ishai et al. proposed the OT extension protocols where a small number of "base"

OTs can be a large number of output OTs with only symmetric-key primitives [25]. The OT extension protocol was subsequently improved in a number of works [30,3,4,28,40,41] with SoftSpokenOT being the state-of-the-art [47]. Notably, another line of work seek to reduce the communication of the extension process by relying on different variants of the Learning Parity with Noise (LPN) assumptions [11,10,50,8,12,9]. Both classes of extension methods do not rely on non-post-quantum assumptions and rely on a small number of base OT correlations, which motivates the design of efficient and post-quantum OT protocols.

As a fundamental component, many candidate constructions have been proposed (e.g. [6,37,1,27,31,18]) and some are still actively utilized in today's MPC libraries. Nevertheless, when considering post-quantum security, the choice is rather limited. In the following we review post-quantum OT protocols in the literature.

OT from Key Exchange. In [33], Masny and Rindal proposed a generic transformation from key exchange to OT and instantiated this transformation with the Mod-LWE-based Kyber scheme [48]. Subsequently, the random oracle programming technique in [33] is abstracted as "programmable once public functions" and utilized in subsequent works [34,35,36]. Unlike Masny-Rindal OT, in those protocols the sender sends the public key as the first message and can be re-used across different sessions.

Nevertheless, this change introduces stronger security requirements of the key exchange scheme, namely, even if the sender's first message is maliciously created, the receiver's response should be indistinguishable from uniform randomness. This property is dubbed with different names ("dense KEM" in [34] or "strong random response in [36]). When using Lattice-based assumptions like LWE, the modulus-to-noise ratio needs to be super-polynomial the information reconciliation cannot be simulated. Thus, OT protocols in [34,35,36] mainly focuses on DH-based encryption schemes which inherently has this property.

OT from Dual-mode Encryption. A line of work [42,44,15,14] is based on dual-mode cryptosystems, which builds on lattice trapdoor [21]. To the best of our knowledge, lattice trapdoor can only be instantiated from plain LWE [46] and thus OT protocols in this class are less efficient than those based on Mod-LWE [33] or Mod-LWR as in this work.

Theoretical Constructions. In [20] the authors proposed a series of transformation that boosts an "elementary" OT protocol with relatively weak security guarantee to one that is UC-secure. They then presented a candidate elementary OT from the post-quantum LPN assumption. Nevertheless, the transformation process involves non-black-box use of cryptographic primitives (e.g. evaluating PKE within GC). Therefore, we believe that the practical performance of this OT is less efficient than OTs built directly from key exchange.

In summary, the Mesny-Rindal protocol [33] instantiated with Kyber PKE [48] represents the state-of-the-art among post-quantum OT protocols in terms of

practical performance. This protocol also serves as the baseline for post-quantum OT in our experiment of Section 6.

1.2 Our Contribution

In this paper, we propose an oblivious transfer protocol from the Saber key exchange protocol, which mimics the classical Naor-Pinkas construction [37] based on Diffie-Hellman key exchange. Using the technique of [17], we argue that with an additional round of consistency check our protocol suffices for the secure composition with the SoftSpoken OT extension [47]. Finally, we implement our protocol and benchmark its performance with the classical OT protocols as well as existing OT protocols with post-quantum security. The experimental results demonstrate that our performance is superior to that of our existing counterparts in the LAN environment. We highlight the advantages of this work as follows.

- Ease of implementation with post-quantum security. Our protocol is easy to implement (less than 300 lines of code in C++), which is a virtue of widely-used classical OT protocols (e.g. [37,18]). In particular, the choice of Mod-LWR assumption allows us to introduce noise via deterministic rounding rather than sampling them from a distribution (e.g. discrete Gaussian), which helps to simplify the design and implementation of the OT protocol.
- Fast running time compared with both classical OT and post-quantum OT. Our protocol outperforms Kyber-based Masny-Rindal OT in both computation and communication costs. It also achieves similar running time in LAN settings compared to OT based on the classical DH-like assumption, however, it incurs a slightly larger communication size

In conclusion, our work provides a simple, efficient and secure way of constructing post-quantum OT and OT extension protocols.

2 Preliminaries

In this section, we define our protocol's notations and the UC security definitions. We also briefly introduce the Mod-LWR assumption and the Saber key exchange protocol.

2.1 Notations

We use $\lambda \in \mathbb{N}$ to denote the security parameter. We use "PPT" to abbreviate "probabilistic, polynomial time" and use $\mathsf{negl}(\lambda)$ to denote a function that shrinks faster than any inverse polynomial of λ. We borrow the notations and parameters from Saber [19]. We denote the ring of integers modulo an integer q as $\mathbb{Z}_q$. We then define polynomial ring $\mathcal{R}_q = \mathbb{Z}_q[X]$ and the quotient ring $\mathcal{R}_q[X]/(X^n + 1)$. In Saber, n is a fixed power of 2 (they choose $n = 256$). We also inherit the notation extension: for a ring $\mathcal{R}$, $\mathcal{R}^{\ell \times k}$ denotes the ring of $\ell \times k$-matrices over $\mathcal{R}$, and the $\pmod p$ operator is extended to polynomials in $\mathcal{R}^q$

and matrices over $\mathcal{R}^q$ by performing modulo operation coefficient-wise. $\beta_\mu(\mathbb{Z}_q)$ (resp. $\beta_\mu(\mathcal{R}_q)$) denotes a distribution over $\mathbb{Z}_q$ (resp. $\mathcal{R}_q$) where each vector element (resp. polynomial coefficient) follows a centered binomial distribution with parameter μ and corresponding standard deviation $\sigma = \sqrt{\frac{\mu}{2}}$. We use boldface letters to denote matrices and vectors. For a probability distribution $\mathcal{X}$ on $\mathbb{Z}_q$, $\mathbf{X} \leftarrow \mathcal{X}(\mathcal{R}_q^{\ell \times k})$ denotes sampling the matrix $\mathbf{X} \in \mathcal{R}^{\ell \times k}$ whose each coefficient is respectively sampled from $\mathcal{X}$.

We denote the rounding function by $\lfloor \cdot \rceil_{q \to p} : \mathbb{Z}_q \to \mathbb{Z}_p$ or $\lfloor \frac{p}{q} \cdot \rceil$ which maps $x \in \mathbb{Z}_q$ into the index of the interval that x belongs to in $\mathbb{Z}_p$. We denote the flooring function by $\lfloor \cdot \rfloor_{q \to p} : \mathbb{Z}_q \to \mathbb{Z}_p$ which maps $x \in \mathbb{Z}_q$ to $\lfloor \frac{q}{p} x \rfloor$ in $\mathbb{Z}_p$. By definition, we can transform this rounding computation into flooring: $\forall x \in \mathbb{Z}_q$, $\lfloor x \rceil_{q \to p} = \lfloor x + \frac{q}{2p} \rfloor_{q \to p}$.

We recall the computationally indistinguishable notion as follows.

Definition 1. *Two distribution ensembles $\mathcal{X} = \{\mathcal{X}_\lambda\}_{\lambda \in \mathbb{N}}$ and $\mathcal{Y} = \{\mathcal{Y}_\lambda\}_{\lambda \in \mathbb{N}}$ are computationally indistinguishable (denoted as $\mathcal{X} \overset{c}{\approx} \mathcal{Y}$) if for any PPT distinguisher $\mathcal{D}$, we have*

$$|\Pr[\mathcal{D}(\mathcal{X}_\lambda) = 1] - \Pr[\mathcal{D}(\mathcal{Y}_\lambda) = 1]| = \mathsf{negl}(\lambda).$$

We define the syntax and correctness of two-message OT protocols as follows.

Definition 2. *A two-message OT protocol is defined by four PPT algorithms $(\mathsf{Setup}, \mathsf{OT}_1, \mathsf{OT}_2, \mathsf{OT}_3)$ where the setup algorithm Setup prepares for the oblivious transfer, for example generates some common random strings as seed for further operand generation.*

- $\mathsf{OT}_1(1^\lambda, \mathsf{crs}, x)$*: Run by the receiver, taking* crs *and receiver's input bit x as input, outputting the receiver's message* otr *and a secret state* st.
- $\mathsf{OT}_2(1^\lambda, \mathsf{crs}, (m_0, m_1), \mathsf{otr})$*: Run by the sender, taking the sender's input messages m_0 and m_1 and receiver's first round message* otr *as input, outputting sender's second round answer* ots.
- $\mathsf{OT}_3(1^\lambda, x, \mathsf{ots}, \mathsf{st})$*: Run by the receiver to evaluate m_x from the interaction above.*

And we also say that a two-round OT scheme is correct, if with probability $1 - \mathsf{negl}(\lambda)$ *the following hold. For every choice bit $x \in \{0, 1\}$ of the reciever and for any input messages m_0 and m_1 of the sender, and for any* $\mathsf{otr}, \mathsf{st} \in \mathsf{OT}_1(1^\lambda, \mathsf{crs}, x)$ *and* $\mathsf{ots} \in \mathsf{OT}_2(1^\lambda, \mathsf{crs}, (m_0, m_1), \mathsf{otr})$*, we have* $\mathsf{OT}_3(1^\lambda, x, \mathsf{ots}, \mathsf{st}) = m_x$.

2.2 LWR and Mod-LWR Problems

The learning with rounding (LWR) problem is introduced by Banerjee et al. [5] and is a "derandomization" technique for the learning with errors (LWE) problem which generates the error terms deterministically. According to [5] and some other works, there exists a reduction from the LWE problem to the LWR problem. Instead of using a random error term to generate "noisy" inner products,

the LWR problem generates deterministic error by scaling and rounding coefficients modulo q to modulo p with $p < q$. We define the LWR and Mod-LWR assumptions in Definition 3 and Definition 4.

Definition 3. *Let $N, m, p, q, \mu \in \mathbb{N}$ be parameters where $p < q$. The LWR problem states that for $\mathbf{A} \leftarrow \mathbb{Z}_q^{N \times m}$, $\mathbf{s} \leftarrow \beta_\mu(\mathbb{Z}_q^{m \times 1})$ and $\mathbf{u} \leftarrow \mathbb{Z}_p^{N \times 1}$,*

$$\left(\mathbf{A}, \lfloor \frac{p}{q} \left(\mathbf{A}^T \mathbf{s}\right) \rceil_{q \to p}\right) \stackrel{c}{\approx} (\mathbf{A}, \mathbf{u})$$

Definition 4. *Let $N, k, p, q, \mu \in \mathbb{N}$ and $\mathcal{R}$ be parameters where $p < q$. The Mod-LWR problem states that for $\mathbf{A} \leftarrow \mathcal{R}_q^{N \times k}$, $\mathbf{s} \leftarrow \beta_\mu(\mathcal{R}_q^{k \times 1})$ and $\mathbf{u} \leftarrow \mathcal{R}_p^{N \times 1}$,*

$$\left(\mathbf{A}, \lfloor \frac{p}{q} \left(\mathbf{A}^T \mathbf{s}\right) \rceil_{q \to p}\right) \stackrel{c}{\approx} (\mathbf{A}, \mathbf{u})$$

The parameters for Saber key exchange are chosen as $\mathcal{R}_p = \mathbb{Z}_p[x]/(x^n + 1)$, $\mathcal{R}_q = \mathbb{Z}_q[x]/(x^n + 1)$, $t = 2^{\epsilon_t}$, $p = 2^{\epsilon_p}$, $q = 2^{\epsilon_q}$ in which $\epsilon_t, \epsilon_p, \epsilon_q \in \mathbb{N}$, and $0 < \epsilon_t + 1 < \epsilon_p < \epsilon_q$, β_μ is binomial distribution.

As mentioned in the preliminaries, the rounding operation can be done with flooring. So, we define relevant constants as follows. $h_1 = \sum_{i=0}^{n-1} \frac{q}{2p} X^i \in \mathcal{R}_q$, $h_2 = \sum_{i=0}^{n-1} \left(\frac{p}{4} - \frac{p}{4t}\right) X^i \in \mathcal{R}_p$ (also in $\mathcal{R}_q$), and $\mathbf{h} = (h_1, \ldots, h_1) \in R_q^{k \times 1}$. It can be verified that $\forall x \in \mathcal{R}_q, \lfloor x \rceil_{q \to p} = \lfloor x + h_1 \rfloor_{q \to p}$. Because the parameters are all powers of 2, flooring is equivalent to a part selection of bits, as shown below.

$$\forall x \in \mathbb{Z}_q,\ \lfloor x \rceil_{q \to p} = \lfloor x + \frac{q}{2p} \rfloor_{2^{\epsilon_q} \to 2^{\epsilon_p}} = \left(x + \frac{q}{2p}\right) \gg (\epsilon_q - \epsilon_p)\ \&\ (2^{\epsilon_p} - 1)$$

In [2], it is shown that the LWR problem with *leaky secrets* can be reduced to the standard LWE problem provided sufficient min-entropy remains in the secret. Moreover, in [32], it is proved that Mod-LWR over cyclotomic rings can be reduced to the ring-LWE problem. We utilize this property when arguing the sender's input privacy in Section 4.2.

2.3 The Saber Key Exchange Protocol

We recall the Mod-LWR-based Saber key exchange protocol [19]. In this protocol, the sender and receiver share public matrix $\mathbf{A}$, and they both have access to a random oracle H. The sender samples a secret $\mathbf{s}$ from the binomial distribution β_μ of polynomial module $\mathcal{R}_q$, computes the rounding $\mathbf{b} = \lfloor \frac{p}{q} \mathbf{A} \cdot \mathbf{s} \rceil$ and sends it to the receiver. Symmetrically the receiver samples a private $\mathbf{s}'$ and computes $\mathbf{b}' = \lfloor \frac{p}{q} \mathbf{A}^T \cdot \mathbf{s}' \rceil$ and sends $\mathbf{b}'$ with a reconciliation information c. Now both parties can compute a shared key: the sender can compute the key from $c, \mathbf{s}$ and $\mathbf{b}'$; the receiver computes from $\mathbf{b}$ and $\mathbf{s}'$.

The two communicating parties sometimes fail to agree on the same key, but using the additional reconciliation data c, we can ensure that this failure probability is negligibly small.

The details of the Saber key exchange are shown in Fig. 2. Then, we can feed the common secret $k = k' \in \mathcal{R}_2$ into H to derive the session keys.

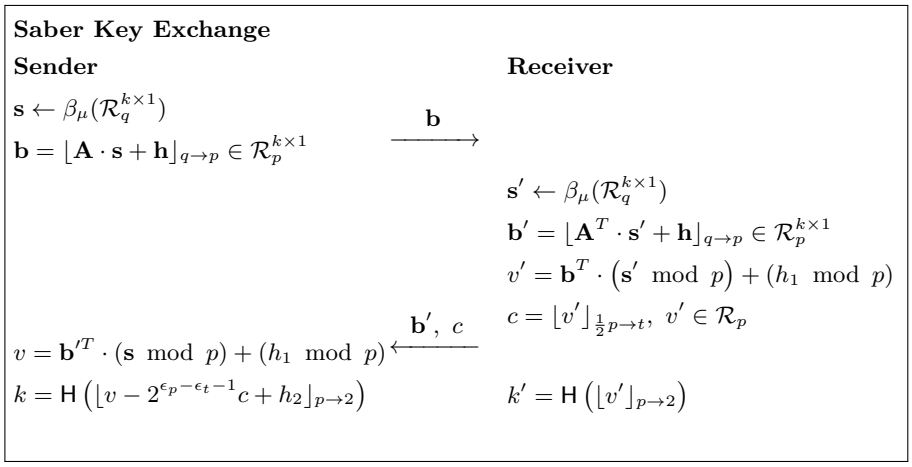

Fig. 2. The Saber key exchange protocol. Here $\mathbf{A} \in \mathcal{R}_q^{k\times k}$ is a public random matrix and $\mathsf{H} : \mathcal{R}_2^{k\times 1} \to \{0,1\}^\lambda$ is a random oracle.

2.4 Security Definitions

We argue the security of our protocol in the Universal Composition (UC) framework [16] and allow the corrupted parties to deviate arbitrarily from the protocol specification. This framework lays a solid foundation for designing and analyzing protocols in arbitrary malicious environments.

In the UC framework, each party is identified by a unique identity pair $(\mathsf{pid}, \mathsf{sid})$, where pid is the Party ID and sid is the Session ID. Parties with the same code and the same sid are said to be a part of the same protocol session. For simplicity, we omit the pid and sid in the protocol description.

The UC framework is based on the "simulation paradigm", in which we analyze the protocols in the real and ideal worlds, respectively. In the ideal world, there is a functionality $\mathcal{F}$ communicating with all honest parties and an adversary Sim. The corrupted parties are controlled by the Sim. In the real world, however, the parties communicate and interact with each other to execute the protocol. The corrupted parties are controlled by an adversary $\mathcal{A}$.

Additionally, UC security captures arbitrary concurrently running protocols using the notion of environment $\mathcal{Z}$, which determines the inputs to parties and sees the outputs generated by those parties. The environment $\mathcal{Z}$ can also communicate with the simulator Sim or the adversary $\mathcal{A}$. We denote the output of $\mathcal{Z}$ in the ideal world as $\mathsf{IDEAL}_{\mathcal{F},\mathsf{Sim},\mathcal{Z}}$ while the output of $\mathcal{Z}$ in the real world is denoted as $\mathsf{REAL}_{\Pi,\mathcal{A},\mathcal{Z}}$. We define a protocol to be UC-secure in Definition 5.

Definition 5. *We say a protocol Π UC-realizes the functionality $\mathcal{F}$, if for any PPT environment $\mathcal{Z}$ and any PPT adversary $\mathcal{A}$, there exists a PPT simulator* Sim *s.t.* $\mathsf{IDEAL}_{\mathcal{F},\mathsf{Sim},\mathcal{Z}} \overset{c}{\approx} \mathsf{REAL}_{\Pi,\mathcal{A},\mathcal{Z}}$.

3 The Saber-based OT Protocol

In this section, we explain the protocol of Fig. 1 in detail. We also prove the correctness of this protocol.

3.1 Our Construction

Following the syntax in Definition 2, we describe the OT protocol. Our construction follows the framework of the Naor-Pinkas OT [37]. We note that in the random oracle model, we can generate the crs by first running a coin-tossing protocol to generate a seed, and then get $(\mathbf{A}, \mathbf{r}) = \mathcal{F}_{\mathsf{RO}}(\mathsf{seed})$. Here $\mathbf{A}$ is the Mod-LWR public matrix and $\mathbf{r}$ is the random correlation in the receiver's otr messages.

- $\mathsf{OT}_1(1^\lambda, \mathsf{crs}, x \in \{0,1\})$:
  - Parse crs to get $\mathbf{A}, \mathbf{r}$.
  - Sample receiver's secret $\mathbf{s}' \leftarrow \beta_\mu(\mathcal{R}_q^{k\times 1})$.
  - Compute $\mathbf{b}'_x = \lfloor \mathbf{A}^T \cdot \mathbf{s}' \rceil_{q\to p}$ and $\mathbf{b}'_{1-x} = \mathbf{r} - \mathbf{b}'_x$.
  - Output $\mathsf{otr} = \mathbf{b}'_0$, $\mathsf{st} = \mathbf{s}'$.
- $\mathsf{OT}_2(1^\lambda, \mathsf{crs}, (m_0, m_1) \in \{0,1\}^{2\lambda}, \mathsf{otr})$:
  - Parse crs to get $\mathbf{A}, \mathbf{r}$.
  - Sample sender's secret $\mathbf{s} \leftarrow \beta_\mu(\mathcal{R}_q^{k\times 1})$.
  - Compute the Mod-LWR sample $\mathbf{b} = \lfloor \mathbf{A} \cdot \mathbf{s} \rceil_{q\to p}$.
  - Compute $v_0 = (\mathbf{b}'_0)^T \cdot (\mathbf{s} \mod p) + h_1, v_1 = (\mathbf{b}'_1)^T \cdot (\mathbf{s} \mod p) + h_1$.
  - Derive the reconciliation information which are used to correctly get the highest bit of v_0 and v_1: $c_0 = \lfloor v_0 \rfloor_{\frac{1}{2}p\to t}, c_1 = \lfloor v_1 \rfloor_{\frac{1}{2}p\to t}$.
  - Derive keys from the highest bit of v_0 and v_1. $k_0 = \mathsf{H}(\lfloor v_0 \rfloor_{p\to 2}), k_1 = \mathsf{H}(\lfloor v_1 \rfloor_{p\to 2})$
  - Encrypt the message with the generated keys. $e_0 = k_0 \oplus m_0, e_1 = k_1 \oplus m_1$, and outputs $\mathsf{ots} = (\mathbf{b}, c_0, c_1, e_0, e_1)$.
- $\mathsf{OT}_3(1^\lambda, x \in \{0,1\}, \mathsf{ots} = (\mathbf{b}, c_0, c_1, e_0, e_1), \mathsf{st})$:
  - Parse ots and st to get $(\mathbf{b}, c_0, c_1, e_0, e_1)$ and $\mathbf{s}'$ respectively.
  - Compute $v' = \mathbf{b}^T \cdot (\mathbf{s}' \mod p) + h_1$.
  - Use the highest bit of v' to derive key chosen by x, with c_x to reconcile the rounding error, i.e. $k_x = \mathsf{H}(\lfloor v' - 2^{\epsilon_p - \epsilon_q - 1} c_x + h_2 \rfloor_{p\to 2})$
  - Decrypt $m_x = e_x \oplus k_x$.

3.2 Correctness

The correctness of our OT protocol follows from the correctness of the Saber key exchange scheme. According to Definition 2, the correctness of an OT protocol means that the receiver can correctly get the message indexed by x after the interaction with overwhelming probability, namely, $\Pr[\mathsf{OT}_3(1^\lambda, \mathsf{ots}) \neq m_x] \leq \mathsf{negl}(\lambda)$. For simplicity, we omit the $q \to p$ subscript and $\mod$ operator in the following analysis.

During the protocol execution, the sender gets $\mathbf{b}'_x = \lfloor \mathbf{A}^T\mathbf{s}' \rceil$ and $\mathbf{b}'_{\bar{x}} = \mathbf{r} - \lfloor \mathbf{A}^T\mathbf{s}' \rceil$ and computes $v_x = \lfloor \mathbf{s}'^T\mathbf{A} \rceil \cdot \mathbf{s} + h_1$ and $v_{\bar{x}} = \lfloor \mathbf{r} - \mathbf{s}'^T\mathbf{A} \rceil \cdot \mathbf{s} + h_1$. The receiver computes $v' = \lfloor \mathbf{s}^T\mathbf{A}^T \rfloor \cdot \mathbf{s}' + h_1$. We now show that with reconciliation information c_x, the keys derived respectively from v' and v_x agree with overwhelming probability.

We utilize a useful observation about reconciliation failure due to Bos et al. [7]. The reconciliation $c = \lfloor u' \rfloor$ between two integer values u, $u' \in \mathbb{Z}_p$ is correct if the distance between u and u' is smaller than $\left(\frac{p}{4} - \frac{p}{4t}\right)$ and fails if the distance between u and u' is bigger than $\left(\frac{p}{4} + \frac{p}{4t}\right)$. Between these values, the probability of success decreases linearly from 1 to 0. Therefore, we should prove that the distance between $v' = \lfloor s^T\mathbf{A}^T \rceil \cdot \mathbf{s}' + h_1$ and $v_x = \lfloor \mathbf{s}'^T\mathbf{A} \rceil \cdot \mathbf{s} + h_1$ is less than $\frac{p}{4} - \frac{p}{4t}$ with only negligible probability.

Following Theorem 1 of [19], which shows proof of successful reconciliation, we explicitly write out the errors introduced by scaling and rounding for the analysis. We use $\mathbf{e}$ and $\mathbf{e}'$ to denote the errors respectively for $\mathbf{A} \cdot \mathbf{s}$ and $\mathbf{A}^T \cdot s'$, i.e. $\lfloor \mathbf{A} \cdot \mathbf{s} \rceil_{q \to p} = \frac{p}{q}\mathbf{A} \cdot \mathbf{s} + \mathbf{e}$ and $\lfloor \mathbf{A}^T \cdot \mathbf{s}' \rceil_{q \to p} = \frac{p}{q}\mathbf{A}^T \cdot \mathbf{s}' + \mathbf{e}'$. We recall the result in Lemma 1.

Lemma 1 (Thoerem 1 in [19]). *Let $e_r \in \mathcal{R}_q$ be a polynomial with uniformly distributed coefficients with range $[-\frac{p}{4t}, \frac{p}{4t}]$. Define*

$$\delta = \Pr[\left\| \left((\mathbf{s}')^T\mathbf{e} - (\mathbf{e}')^T\mathbf{s} + e_r\right) \mod p \right\|_\infty > \frac{q}{4}],$$

then the two parties can agree on a n-bit key with probability $1 - \delta$.

Proof. Using $\mathbf{e}$ and $\mathbf{e}'$ we have defined above, the polynomial v' and v_x can be written as following

$$v' = \left(\frac{p}{q}A \cdot \mathbf{s} + \mathbf{e}\right)^T \mathbf{s}' + h_1 = \frac{p}{q}\mathbf{s}'^T\mathbf{A}\mathbf{s} + \mathbf{s}'^T\mathbf{e} + h_1$$
$$v_x = \left(\frac{p}{q}\mathbf{A}^T \cdot \mathbf{s}' + \mathbf{e}'\right)^T \mathbf{s} + h_1 = \frac{p}{q}\mathbf{s}'^T\mathbf{A}\mathbf{s} + \mathbf{e}'^T\mathbf{s} + h_1$$

We thus have $\Delta v_x = \|v_x - v'\| = \|\mathbf{s}'^T\mathbf{e} - \mathbf{e}'^T\mathbf{s} \mod p\|_\infty$. Using the observation of Bos et al. [7], the probability of reconciliation failure can be computed as $\Pr[\|\Delta v_x + e_r\|_\infty \leq \frac{p}{4}]$. The key agreement of v_x and v' is $(1 - \delta)$ correct.

Concretely, using the recommended parameter set of Saber [19], we have $\delta < 2^{-136}$. The following corollary follows directly from Lemma 1 and the correctness of the symmetric-key encryption scheme.

Corollary 1. *The OT protocol in Fig. 1 is correct with $1 - \delta$ probability.*

4 Security

Similar to practically efficient OT protocols in the literature [18,37,6], the provable UC security of our Saber-based OT protocol in the last section remains

elusive. In this section, we list several ways to argue the security of the basic protocol by adding additional checks and tweaking the parameters of the underlying Mod-LWR problem.

Canetti et al. [17] proposed a weakened definition of security for OT such that protocols secure under this definition suffices for the KOS OT extension protocol [28] and the resulting composed protocol is UC-secure. Therefore, proving that our protocol satisfies the basic notion certifies its usage in OT extension, which is a common technique to reduce the cost of oblivious transfer.

Under the weakened security notion, security against a corrupted sender follows from the Mod-LWR assumption. Nevertheless, due to the existence of additional reconciliation information, which is absent in Diffie-Hellman-type protocols as in [17], we experience difficulty when reducing security against a corrupted receiver to the Mod-LWR problem, as it's unclear how to acquire such information in the original Mod-LWR game.

To circumvent this issue, we propose two solutions. The first one considers the reconciliation information as leakage, and by recent results [32], the Mod-LWR problem with key leakage can be reduced to the Mod-LWE problem without leakage with shorter keys. The second solution is to abolish the reconciliation information altogether and let two parties perform local rounding. This approach appears under different contexts in the literature (e.g. in the original reduction from LWR to LWE [5] and the HSS construction [13]). The caveat of this approach is that now we require a super-polynomial modulus-to-noise ratio to ensure negligible error.

We note that the security proof of KOS [28] is recently found to be flawed and the SoftSpoken OT extension protocol is considered to be state-of-the-art [47]. Nevertheless, as we show in Section 5 the original argument in [17] still applies to the SoftSpoken OT protocol. Therefore, we still choose to prove that our protocol satisfies the weakened OT security notion in [17].

4.1 Relaxing the OT Functionality for OT Extensions

In [17], it is shown that the OT extension protocols only requires a *relaxed* notion of base OT security. In particular, it suffices for the base OT to output random messages, and the sender can launch a selective failure attack on the receiver's choice bit.

We recall the weakened functionality allowing selective failure attack for a corrupted sender in Fig. 3 and the corresponding security notion in Definition 6. Theorem 2 shows that this weakened security notion suffices for the composition with the SoftSpoken OT extension protocol.

Definition 6. *Let $\mathcal{F}_{\mathsf{SF\text{-}rOT}}$ be the oblivious transfer functionality as shown in Figure 3. We say that a protocol Π_{OT} securely computes $\mathcal{F}_{\mathsf{SF\text{-}rOT}}$ with sender-sided simulation with input extractability of receiver if the following holds:*

1. *For every PPT adversary S^* controlling the sender in the real model, there exists a non-uniform PPT adversary Sim for the ideal model, such that for*

any environment $\mathcal{Z}$:

$$\mathsf{IDEAL}_{\mathcal{F}_{\mathsf{SF\text{-}rOT}},\mathsf{Sim},\mathcal{Z}} \approx \mathsf{REAL}_{\Pi_{\mathsf{OT}},\mathsf{S}^*,\mathcal{Z}}.$$

2. *For every PPT adversary R^* controlling the receiver, the following holds:*
   - ***Extractability**: If the sender did not abort, then there exists a PPT extractor Ext such that the following holds.*

$$\Pr\left[((m_0,m_1),(b,m_b)) \leftarrow \langle \mathsf{S},\mathsf{R}^*\rangle, x' \leftarrow \mathsf{Ext}^{\mathsf{R}^*} : x \neq x' \wedge (m_0,m_1) \neq \bot\right] = \mathsf{negl}(\lambda)$$

   - ***Input Privacy**: R^* cannot compute both sender messages except with negligible probability.*

$$\Pr\left[((m_0,m_1),(m_0',m_1')) \leftarrow \langle \mathsf{S},\mathsf{R}^*\rangle : m_0 = m_0' \wedge m_1 = m_1'\right] = \mathsf{negl}(\lambda)$$

4.2 Consistency Checks

Following the technique of [17], we add an additional checking round to meet the security requirement of Definition 6. We present the protocol in Fig. 4.

We argue the security against a corrupted sender in Lemma 2. As for the case of a corrupted receiver, the situation is more tricky since reducing the sender's message privacy to the Mod-LWR problem requires generating the reconciliation information. Toward this end, we propose two solutions.

- We can view the reconciliation information as leakage and reduce the sender's input privacy to the stronger Mod-LWR with leaky secret assumption, as shown in Lemma 4. [32] shows that this assumption can be reduced to the standard Mod-LWE assumption with a shorter secret.
- Another approach is to rely on local processing to let the two parties agree on the same key. This can be done by locally rounding the v value, and the same approach appeared under different contexts in the literature. The downside is that we have to increase the modulus-to-noise ratio to achieve negligible error probability. We summarize the result in Lemma 5. We note that this approach was also used in [34].

We present and prove the lemmas in the following. We state the security of the protocol $\Pi_{\mathsf{SF-rOT}}$ in Theorem 1 without proving it, because this result follows naturally from the respective lemmas. Since we are proving that ℓ parallel executions of $\Pi_{\mathsf{SF-rOT}}$ with batched checking securely implement $\mathcal{F}_{\mathsf{SF\text{-}rOT}}$, we use the subscript i to denote the messages related to the i-th execution.

Theorem 1. *Assuming the Mod-LWR assumption and 1) Mod-LWE assumption or 2) the super-polynomial noise-to-noise ratio, i.e., $\frac{\mu}{p} = \mathsf{negl}(\lambda)$, then $\Pi_{\mathsf{SF-rOT}}$ UC-securely implements ℓ instances of $\mathcal{F}_{\mathsf{SF\text{-}rOT}}$ functionality in the observable random oracle model with sender-sided simulation.*

Functionality $\mathcal{F}_{\mathsf{SF\text{-}rOT}}$ of $\binom{2}{1}$-OT

$\mathcal{F}_{\mathsf{SF\text{-}rOT}}$ interacts with a sender S and receiver R:

- On input $(\textsc{Choose}, \mathsf{rec}, \mathsf{sid})$ from R; if no message of the form $(\mathsf{rec}, \mathsf{sid}, x)$ has been recorded in the memory, sample $x \leftarrow \{0,1\}$, store $(\mathsf{rec}, \mathsf{sid}, x)$ and send $(\mathsf{rec}, \mathsf{sid})$ to S and (x, sid) to R. If a message of the form $(\mathsf{sen}, \mathsf{sid}, (m_0, m_1))$ is stored, send $(\mathsf{sent}, \mathsf{sid}, (x, m_x))$ to R and $(\mathsf{sent}, \mathsf{sid}, (x, m_x))$ to S and ignore future messages with the same sid.
- On input $(\textsc{Choose}^*, \mathsf{rec}, \mathsf{sid}, x)$ from R^* where $x \in \{0,1\}$; if no message of the form $(\mathsf{rec}, \mathsf{sid}, x)$ has been recorded in the memory, store $(\mathsf{rec}, \mathsf{sid}, x)$ and send $(\mathsf{rec}, \mathsf{sid})$ to S and (x, sid) to R. If a message of the form $(\mathsf{sen}, \mathsf{sid}, (m_0, m_1))$ is stored, send $(\mathsf{sent}, \mathsf{sid}, (x, m_x))$ to R and $(\mathsf{sent}, \mathsf{sid}, (m_0, m_1))$ to S and ignore future messages with the same sid.
- On input $(\textsc{Guess}^*, \mathsf{sen}, \mathsf{sid}, x')$ from S^*, if $(\mathsf{rec}, \mathsf{sid}, x)$ exists in memory, $x' \in \{0, 1, \bot, \top\}$ and there does not exist $(\mathsf{sen}, \mathsf{sid}, (\textsc{Guess}, \cdot))$ in memory then store $(\mathsf{sen}, \mathsf{sid}, (\textsc{Guess}, x'))$ in memory and perform the following:
  - If $x' = \bot$, do nothing.
  - If $x' = \top$, send $(\textsc{Cheat-Detect}, \mathsf{S})$ to R and $(\textsc{Cheat-Detect})$ to S.
  - If $x' = x$, send $(\textsc{Cheat-Undetect})$ to S.
  - If $x' \neq x$, send $(\textsc{Cheat-Detect}, \mathsf{S})$ to R and $(\textsc{Cheat-Detect})$ to S.
- On input $(\textsc{Transfer}, \mathsf{sen}, \mathsf{sid})$ from S, if no message of the form $(\mathsf{sen}, \mathsf{sid}, (m_0, m_1))$ is stored; sample $m_0, m_1 \leftarrow \{0,1\}^\lambda$, store $(\mathsf{sen}, \mathsf{sid}, (m_0, m_1))$ in memory and send $(\textsc{Received}, \mathsf{sid})$ to R and S. If a message of the form $(\mathsf{rec}, \mathsf{sid}, b)$ is stored, send $(\mathsf{sent}, \mathsf{sid}, (x, m_x))$ to R and $(\mathsf{sent}, \mathsf{sid}, (m_0, m_1))$ to S and ignore future messages with the same sid.
- On input $(\textsc{Transfer}^*, \mathsf{sen}, \mathsf{sid}, (m_0, m_1))$ from S^*, if no message of the form $(\mathsf{sen}, \mathsf{sid}, (m_0, m_1))$ is stored then store $(\mathsf{sen}, \mathsf{sid}, (m_0, m_1))$ in memory and send $(\textsc{Received}, \mathsf{sid})$ to R and S. If a message of the form $(\mathsf{rec}, \mathsf{sid}, x)$ is stored, send $(\mathsf{sent}, \mathsf{sid}, (x, m_x))$ to R and $(\mathsf{sent}, \mathsf{sid}, (m_0, m_1))$ to S and ignore future messages with the same sid.
- On input $(\textsc{Abort}, \mathsf{rec}, x, \mathsf{sid})$ from R^*, if messages of the form $(\mathsf{sen}, \mathsf{sid}, (m_0, m_1))$ is stored; send $(\mathsf{sent}, \mathsf{sid}, (x, m_x))$ to R and $(\textsc{Abort}, \mathsf{sid}, (m_0, m_1))$ to S. Ignore future messages with the same sid.

Fig. 3. The ideal oblivious transfer functionality with selective failure attack on the receiver's choice bit.

Corrupted Sender. We argue that the protocol $\Pi_{\mathsf{SF-rOT}}$ can implement the functionality $\mathcal{F}_{\mathsf{SF\text{-}rOT}}$ according to Definition 6 against a corrupted sender. We first present the simulator in Figure 5 and then argue its effectiveness in Lemma 2.

Lemma 2. *Assuming the Mod-LWR assumption holds, the protocol $\Pi_{\mathsf{SF-rOT}}$ securely implements the $\mathcal{F}_{\mathsf{SF-rOT}}$ functionality against a corrupted sender according to Definition 6.*

Proof. We prove the effectiveness of the simulation via a series of hybrid experiments.

Protocol $\Pi_{\mathsf{SF-rOT}}$ with Saber

Parameters: Mod-LWE parameters q, p, t, n, k and security parameter λ.
Random Oracles: $\mathcal{F}_{\mathsf{RO1}} : \{0,1\}^\lambda \times \{0,1\}^\lambda \to \mathcal{R}_p^{k\times 1}$, $\mathcal{F}_{\mathsf{RO2}} : \{0,1\}^\lambda \times \mathcal{R}_2^{k\times 1} \to \{0,1\}^\lambda$, $\mathcal{F}_{\mathsf{RO3}} : \{0,1\}^\lambda \times \{0,1\}^\lambda \to \{0,1\}^\lambda$, $\mathcal{F}_{\mathsf{RO4}} : \{0,1\}^\lambda \times \{0,1\}^{l\times\lambda} \to \{0,1\}^\lambda$. In the following protocol, we omit sid in $\mathcal{F}_{\mathsf{RO}}(\mathsf{sid}, \cdot)$ for simplicity.
Common Random String: The matrix $\mathbf{A} \in \mathcal{R}_q^{k\times k}$ and the vector $\mathbf{r} \in \mathcal{R}_p^{k\times 1}$.
Inputs: Sender has no input while the receiver holds a choice bit $x \in \{0,1\}$.
Outputs: Sender gets two random messages $k_0, k_1 \in \{0,1\}^\lambda$ while receiver gets the chosen message k_x.

Sender		**Receiver**
$input : none$		$input : x$
$output : k_0, k_1$		$output : k_x$
		$\mathbf{s}'_i \leftarrow \beta_\mu(\mathcal{R}_q^{k\times 1})$
		$\mathbf{b}'_x = \lfloor \mathbf{A}^T \cdot \mathbf{s}' \rceil_{q\to p}$
$\mathbf{s} \leftarrow \beta_\mu(\mathcal{R}_q^k)$	$\xleftarrow{\mathbf{b}'_0}$	$\mathbf{b}'_{1-x} = \mathbf{r} - \lfloor \mathbf{A}^T \cdot \mathbf{s}' \rceil_{q\to p}$
$\mathbf{b}'_1 = \mathbf{r} - \mathbf{b}'_0$		
$\mathbf{b} = \lfloor \mathbf{A} \cdot \mathbf{s} \rceil_{q\to p}$		
$v_0 = \mathbf{b}'_0 \cdot (\mathbf{s} \mod p) + h_1$		
$v_1 = \mathbf{b}'_1 \cdot (\mathbf{s} \mod p) + h_1$		
$c_0 = \lfloor v_0 \rfloor_{\frac{1}{2}p\to t}$, $c_1 = \lfloor v_1 \rfloor_{\frac{1}{2}p\to t}$		
$k_0 = \mathcal{F}_{\mathsf{RO2}}(\lfloor v_0 \rfloor_{p\to 2})$		
$k_1 = \mathcal{F}_{\mathsf{RO2}}(\lfloor v_1 \rfloor_{p\to 2})$		$v' = \mathbf{b}^T \cdot (\mathbf{s}' \mod p) + h_1$
$\mathsf{chall} = \mathcal{F}_{\mathsf{RO3}}(k_0) \oplus \mathcal{F}_{\mathsf{RO3}}(k_1)$		$k_x = \mathcal{F}_{\mathsf{RO2}}(\lfloor v' - 2^{\epsilon_p - \epsilon_q - 1} c_x + h_2 \rfloor_{p\to 2})$
$\gamma = \mathcal{F}_{\mathsf{RO3}}(\mathcal{F}_{\mathsf{RO3}}(k_0))$		
	$\xrightarrow{\mathbf{b}, c_0, c_1, \mathsf{chall}, \gamma}$	
		$\mathsf{ans} = \mathcal{F}_{\mathsf{RO3}}(k_x) \oplus x \cdot \mathsf{chall}$
	$\xleftarrow{\mathsf{ans}}$	Check $\mathcal{F}_{\mathsf{RO3}}(\mathsf{ans}) = \gamma$
Check $\mathcal{F}_{\mathsf{RO3}}(\mathsf{ans}) = \mathcal{F}_{\mathsf{RO3}}(\mathcal{F}_{\mathsf{RO3}}(k_0))$		

Batch Verification: When running ℓ instances of the OT protocols together, the two parties can hash the $\mathsf{ans}_1, ..., \mathsf{ans}_\ell$ together using an additional random oracle $\mathcal{F}_{\mathsf{RO4}}$. In this way we can reduce communication bandwidth in the third message.

Fig. 4. The Saber-based oblivious transfer protocol with additional consistency checks.

Hybrid 1 This is the real world experiment. The receiver R uses real inputs.

Hybrid 2 In this hybrid, we replace the receiver's first message according to the simulation strategy, i.e. using random inputs independent from the real input. By the Mod-LWR assumption, the output of this hybrid is computationally indistinguishable from the previous hybrid.

Hybrid 3 In this hybrid, the simulator extracts the output values $(k_{i,0}, k_{i,1})$ for $i \in [\ell]$. This step is only conceptual and does not change output distribution.

Hybrid 4 In this hybrid, the simulator aborts if the extraction for β, y, or y' fails. By the property of the random oracle, such extraction fails only when the adversary finds collision in it, which happens with negligible probability.

Hybrid 5 In this hybrid, we extract the selective failure input x'_i for $i \in [\ell]$ according to the simulation strategy and sends it to $\mathcal{F}_{\mathsf{SF\text{-}rOT}}$. Since the simulator simply mimics the actual behavior of the honest receiver, this step does not change the output distribution.

Hybrid 6 In this hybrid, the simulator sends the extracted output values $(k_{i,0}, k_{i,1})_{i\in[\ell]}$ to $\mathcal{F}_{\mathsf{SF\text{-}rOT}}$ if it does not abort. Since the functionality guarantees the abort probability to be the same with the receiver with real input, this step does not change the output distribution. This is the ideal world experiment.

Corrupted Receiver We show how to extract the receiver's choice bit by observing the random oracle transcript in Fig. 6 and states its correctness in Lemma 3.

Lemma 3. *The extractor in Fig. 6 satisfies the requirement of Definition 6 if OT satisfies input privacy.*

Proof. Let $i \in [\ell]$ be the index. Since the honest sender does not aborts, the corrupted receiver R^* must have sent back a valid answer ans_i. By the property of the random oracle, the probability of R^* outputting a correct answer without querying the RO is negligible. Moreover, input privacy ensures that the probability of the adversary getting both output is negligible. Therefore, by following the extraction strategy in Fig. 6, we can extract the effective input of R^*.

Arguing for input privacy is a trickier task. This is because the information reconciliation data c_0, c_1 is related to the sender's local secret $\mathbf{s}$ and cannot be simulated by the reduction process in the Mod-LWR game. Our first idea is to view the reconciliation data as leakage on the secret $\mathbf{s}$. By Corollary 5.13 of [32], the Mod-LWR problem with a leaky secret can be reduced to the Mod-LWE problem with shorter secret dimensions. Namely, we have the following result.

Lemma 4. *Assuming Mod-LWE, the protocol $\Pi_{\mathsf{SF-rOT}}$ satisfies the security requirement for a corrupted sender in Definition 6.*

Proof (sketch). By Definition 6, input privacy requires that for any PPT adversary $\mathcal{A}$ the probability of outputting both messages is negligible. We argue this using a two-step process.

Sim **for** S^*

Choose: Sim runs the honest receiver's algorithm with random $x_i \in \{0,1\}$ for $i \in [\ell]$.
Transfer: S^* sends $(\{\mathbf{b}_i\}_{i\in[\ell]}, \mathsf{Chall}, \gamma)$.
Response:

- *Message Decryption*: Sim extracts and sender's messages as follows:
  1. Sim computes k_{i,x_i} following the honest receiver's program.
  2. Sim extracts candidate $k_{i,\bar{x_i}}$ values for each $i \in [\lambda]$ by observing the queries made to $\mathcal{F}_{\mathsf{RO3}}$ for computing $\mathsf{chall}_i = \mathcal{F}_{\mathsf{RO3}}(k_{i,0}) \oplus \mathcal{F}_{\mathsf{RO3}}(k_{i,1})$.
- *Challenge Verification and Response Computation*: Sim extracts values by observing RO queries as follows:
  1. Sim extracts β s.t. $\mathcal{F}_{\mathsf{RO3}}(\beta) = \gamma$. Set $\mathsf{Ans} = \beta$ by observing $\mathcal{F}_{\mathsf{RO3}}$. Sim observes $\mathcal{F}_{\mathsf{RO4}}$ to extract $y = \{y_i\}_{i\in[\ell]}$, s.t. $\mathcal{F}_{\mathsf{RO4}}(y) = \mathsf{Ans}$.
  2. For $i \in [\ell]$, Sim extracts y_i' s.t. $\mathcal{F}_{\mathsf{RO3}}(y_i') = y_i$ for each $i \in [\ell]$. If Sim either finds two or more matching queries, or he finds no matching query then he invokes $\mathcal{F}_{\mathsf{SF-rOT}}$ with input messages $k_{i,0} = k_{i,1} = \perp$ and aborts. For $i \in [\ell]$, Sim performs the following steps.
  3. If $y_i' = k_{i,0}$ then set $x_i' = 0$. Else, if $\mathcal{F}_{\mathsf{RO3}}(k_{i,0}) \oplus \mathsf{chall}_i = y_i'$ then set $x_i' = 1$. Invoke $\mathcal{F}_{\mathsf{SF-rOT}}$ with input $(\textsc{Guess}, \mathsf{Sim}, \mathsf{sid}, x_i')$.
  4. Else, Sim aborts in the simulated execution.
- For $i \in [\ell]$, Sim computes Ans' following the honest receiver algorithm using input $\{x_i\}_{i\in[\ell]}$ and sends it to S^* .
- Sim sends $(\textsc{Transfer}^*, \mathsf{Sim}, \mathsf{sid}, (k_{i,0}, k_{i,1}))$ for $i \in [\ell]$ to $\mathcal{F}_{\mathsf{SF\text{-}rOT}}$.

Fig. 5. Simulation against a statically corrupt sender S^*

Ext for R^*

Choose: Ext invokes R^* and receives the messages $\mathsf{b}'_{i,0}$ for $i \in [\ell]$.
Transfer: Ext follows the honest sender's algorithm and send back the messages to R^*.
Response and Verification: Ext receives ans_i from R^*.

- If there exists $i \in [\ell]$ s.t. ans_i is invalid according to the sender's algorithm then Ext outputs $\perp$.
- For $i \in [\ell]$:
  - If R^* queried $\mathcal{F}_{\mathsf{RO2}}$ with input $(\lfloor v_{i,0} \rfloor_{p\to 2})$ then Ext sets $x_i = 0$;
  - If R^* queried $\mathcal{F}_{\mathsf{RO2}}$ with input $(\lfloor v_{i,1} \rfloor_{p\to 2})$ then Ext sets $x_i = 1$;
  - Otherwise, Ext sets $x_i = \perp$.
- Ext outputs $\{x_i\}_{i\in[\ell]}$.

Fig. 6. Extractor for a corrupted receiver R^*.

1. First we show Mod-LWR with leaky secret implies input privacy. Firstly, we can use a similar trick as in [19] to lift $\mathbf{r} \in \mathbb{Z}_p^n$ into $\mathbb{Z}_q^n$. Let

$$(\mathbf{A}', \mathbf{y}) = \left(\begin{bmatrix} \mathbf{A} \\ \mathbf{r}^T \end{bmatrix}, \lfloor \begin{bmatrix} \mathbf{A} \\ \mathbf{r}^T \end{bmatrix} \cdot \mathbf{s}_i \rceil_{q \to p} \right) \qquad \mathsf{aux} = (\mathbf{b}'_{i,0})^T \cdot \mathbf{s}_i$$

   be the Mod-LWR samples and leakage on $\mathbf{s}_i$ respectively. The Mod-LWR problem with leakage states that conditioned on $(\mathbf{A}', \mathsf{aux})$, $\mathbf{y}'$ is computationally indistinguishable from uniformly random vector over $\mathbb{Z}_p^{n+1}$.

   Now we show how to reduce the input privacy of $\Pi_{\mathsf{SF-rOT}}$ to the Mod-LWR problem with leakage. Instead of generating $v_{i,1} = \lfloor (\mathbf{r} - \mathbf{b}'_{i,0})^T \cdot \mathbf{s}_i \rceil_{q \to p}$, we compute $v_{i,1} = \mathsf{aux} - y$ where y denotes the last coordinate of $\mathbf{y}$. Notice that this step will introduce an error $e \in \{-1, 0, 1\}$ on each polynomial coefficient. But the error will be eliminated by subsequent rounding operations when generating $k_{i,0}$ and $k_{i,1}$.

   When $\mathbf{y}$ follows the uniformly random distribution, the sum of $v_{i,0}$ and $v_{i,1}$ is uniformly random in the view of the adversary. By the property of the random oracle the adversary cannot output both $k_{i,0}$ and $k_{i,1}$.
2. Moreover, by Corollary 5.13 of [32], the Mod-LWR problem with leakage can be reduced to the Mod-LWE problem without leakage, albeit with shorter keys.

Another idea is to abolish the information reconciliation data altogether and let the two parties perform rounding locally. This approach appeared in the literature in LWR reduction [5] and lattice-based homomorphic secret sharing [13]. We state the result as follows.

Lemma 5. *Suppose* $\frac{\mu}{p} = \mathsf{negl}(\lambda)$ *then we can modify the protocol* $\Pi_{\mathsf{SF-rOT}}$ *to satisfy the input privacy of Definition 6 as follows. The sender does not send* $c_{i,0}, c_{i,1}$ *for* $i \in [\ell]$ *while the receiver derives* $k_{i,x_i} = \mathcal{F}_{\mathsf{RO2}}(\lfloor v'_i + h_2 \rfloor_{p \to 2})$.

Proof (sketch). By Lemma 1 of [13], when $\frac{\mu}{p} = \mathsf{negl}(\lambda)$ the correctness of the modified protocol still holds with overwhelming probability. Therefore, we can apply the first step in the proof of Lemma 4 to argue that under the Mod-LWR assumption, any PPT receiver cannot query the pre-images of both $k_{i,0}, k_{i,1}$ for any $i \in [\ell]$.

5 SoftSpoken OT Extension with Weakened Base OT

In [47], a flaw in the security proof of KOS OT extension [28] is discovered. Therefore, in this section, we adapt the security argument in [17] (which applies to the flawed KOS OT extension) to the state-of-the-art SoftSpoken OT extension protocol. In particular, we argue that with a base OT protocol $\Pi_{\mathsf{SF-rOT}}$ satisfying the weakened $\mathcal{F}_{\mathsf{SF-rOT}}$ functionality, the SoftSpoken OT extension protocol can still be argued UC-secure.

We recall the composed SoftSpoken OT protocol in Fig. 8 and Fig. 9 and prove their security in Theorem 2. Since the composed protocol and the proof techniques follows from a straightforward adaptation of the techniques in [17], we defer the protocol description and the security proof to Appendix A.

Theorem 2. *In $\Pi_{\mathsf{SoftSpoken}}$, assuming PRGs are secure pseudorandom generator, $\mathcal{R}$ is an ϵ-universal hash family, $\mathcal{F}_{\mathsf{RO}}$ is a hash function satisfying the functionality of observable random oracle and $\Pi^{k}_{\mathsf{SF-rOT}}$ implements k instances of $\mathcal{F}_{\mathsf{SF-rOT}}$, then $\Pi_{\mathsf{SoftSpoken}}$ UC-securely implements $N = \mathsf{poly}(\lambda)$ instances of extended OT functionality.*

6 Implementation

Parameters. We use the concrete parameters from Saber [19], where $\mathcal{R}_q = \mathbb{Z}_q[x]/(x^{256}+1), q = 2^{13}$, $\mathcal{R}_p = \mathbb{Z}_p[x]/(x^{256}+1), q = 2^{10}$, β_μ is binomial distribution with $\mu = 5, 4, 3$, k is the rank of module, $k = 2, 3, 4$. In our implementation, we choose the parameters of Saber where $k = 3$, which achieves 192-bit security.

Remark 1 *There was an error in Saber's security estimates in the round 2 documents and the paper. Still, it has been confirmed by several parties that "the correct security levels are lower than those stated in the round 2 documents, but do not affect the NIST levels as such" [39].*

We implement our protocol using emp-toolkit [49]. We adapted emp-ot's base OT test and Saber's reference implementation of IND-CPA KEM. We instantiate the random oracle using the SHA256 hash function. We perform experiments on an Intel processor at 2.90GHz (Intel(R) Core(TM) i7-10700F CPU @ 2.90GHz), with 8 cores, 16GB of RAM and Linux OS. Each party is given one single thread to execute on. The parties communicate over local loopback.

We compare our protocols with several other implementations, including the Chou-Orlandi OT [18], Naor-Pinkas OT [37], and the Kyber version of Masny-Rindal OT [33]. The first three adapt implementations in emp-ot, and the last one uses the libOTe [43]. Both the parameters in our protocol and Kyber are chosen to achieve 192-bit security, and the two DH-based OT protocols can achieve classic 128-bit security. Our code is publicly available at `https://github.com/RabbitCabbage/Saber-OT.git` The experimental result is shown in Table 1, where the communication size is computed according to one single base OT while the execution time is the total running time of a batch of 128 OT executions (capturing the common use case in OT extension).

Our results show that our OT protocol outperforms Kyber-based Masny-Rindal OT in terms of both computation and communication costs. Furthermore, in terms of computation, our OT protocol is faster than the current CDH-based OT, making it competitive to replace CDH-based OT in the high-bandwidth network setting (LAN). The fast running time is due to our protocol construction and Saber's design choice. In particular, the receiver in our OT construction has fewer hash function calls, and deterministic noise introduced by high-speed shift

Table 1. Experiment results of the comparison between our protocol and protocols in the literature with classical and post-quantum security.

Protocol	Assumption	Execution Time (Sender/Receiver)	Communication Size	Post Quantum	Security Level
Chou-Orlandi [18]	CDH	8ms/9ms	64B	No	128 bit
Naor-Pinkas [37]	CDH	9ms/9ms	80B	No	128 bit
Masny-Rindal [33]	Mod-LWE	16ms/23ms	4.5KB	Yes	192 bit
This work	Mod-LWR	5ms/6ms	2.2KB	Yes	192 bit

operations makes Saber's C implementation faster than Kyber. Like Kyber, our protocol needs more communication, a common disadvantage of lattice-based cryptosystems, but in a LAN setting, this additional communication has little impact. Additionally, we still have spaces for optimization. On the one hand, we implement our protocol without AVX instruction set, but Kyber's implementation in libOTe has this optimization; on the other hand, similar to the Naor-Pinkas OT based on CDH assumption, the Mod-LWR samples of each party can be computed in a preprocessing phase, saving online time.

References

1. Aiello, W., Ishai, Y., Reingold, O.: Priced oblivious transfer: How to sell digital goods. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 119–135. Springer, Heidelberg (May 2001). https://doi.org/10.1007/3-540-44987-6_8
2. Alwen, J., Krenn, S., Pietrzak, K., Wichs, D.: Learning with rounding, revisited - new reduction, properties and applications. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part I. LNCS, vol. 8042, pp. 57–74. Springer, Heidelberg (Aug 2013). https://doi.org/10.1007/978-3-642-40041-4_4
3. Asharov, G., Lindell, Y., Schneider, T., Zohner, M.: More efficient oblivious transfer and extensions for faster secure computation. In: Sadeghi, A.R., Gligor, V.D., Yung, M. (eds.) ACM CCS 2013. pp. 535–548. ACM Press (Nov 2013). https://doi.org/10.1145/2508859.2516738
4. Asharov, G., Lindell, Y., Schneider, T., Zohner, M.: More efficient oblivious transfer extensions with security for malicious adversaries. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015, Part I. LNCS, vol. 9056, pp. 673–701. Springer, Heidelberg (Apr 2015). https://doi.org/10.1007/978-3-662-46800-5_26
5. Banerjee, A., Peikert, C., Rosen, A.: Pseudorandom functions and lattices. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 719–737. Springer, Heidelberg (Apr 2012). https://doi.org/10.1007/978-3-642-29011-4_42
6. Bellare, M., Micali, S.: Non-interactive oblivious transfer and applications. In: Brassard, G. (ed.) CRYPTO'89. LNCS, vol. 435, pp. 547–557. Springer, Heidelberg (Aug 1990). https://doi.org/10.1007/0-387-34805-0_48
7. Bos, J.W., Costello, C., Ducas, L., Mironov, I., Naehrig, M., Nikolaenko, V., Raghunathan, A., Stebila, D.: Frodo: Take off the ring! Practical, quantum-secure key exchange from LWE. In: Weippl, E.R., Katzenbeisser, S., Kruegel, C., Myers,

A.C., Halevi, S. (eds.) ACM CCS 2016. pp. 1006–1018. ACM Press (Oct 2016). https://doi.org/10.1145/2976749.2978425
8. Boyle, E., Couteau, G., Gilboa, N., Ishai, Y., Kohl, L., Resch, N., Scholl, P.: Correlated pseudorandomness from expand-accumulate codes. In: CRYPTO 2022, Part II. pp. 603–633. LNCS, Springer, Heidelberg (Aug 2022). https://doi.org/10.1007/978-3-031-15979-4_21
9. Boyle, E., Couteau, G., Gilboa, N., Ishai, Y., Kohl, L., Resch, N., Scholl, P.: Oblivious transfer with constant computational overhead. In: EUROCRYPT 2023, Part I. pp. 271–302. LNCS, Springer, Heidelberg (Jun 2023). https://doi.org/10.1007/978-3-031-30545-0_10
10. Boyle, E., Couteau, G., Gilboa, N., Ishai, Y., Kohl, L., Rindal, P., Scholl, P.: Efficient two-round OT extension and silent non-interactive secure computation. In: Cavallaro, L., Kinder, J., Wang, X., Katz, J. (eds.) ACM CCS 2019. pp. 291–308. ACM Press (Nov 2019). https://doi.org/10.1145/3319535.3354255
11. Boyle, E., Couteau, G., Gilboa, N., Ishai, Y., Kohl, L., Scholl, P.: Efficient pseudorandom correlation generators: Silent OT extension and more. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part III. LNCS, vol. 11694, pp. 489–518. Springer, Heidelberg (Aug 2019). https://doi.org/10.1007/978-3-030-26954-8_16
12. Boyle, E., Couteau, G., Gilboa, N., Ishai, Y., Kohl, L., Scholl, P.: Correlated pseudorandom functions from variable-density LPN. In: 61st FOCS. pp. 1069–1080. IEEE Computer Society Press (Nov 2020). https://doi.org/10.1109/FOCS46700.2020.00103
13. Boyle, E., Kohl, L., Scholl, P.: Homomorphic secret sharing from lattices without FHE. In: Ishai, Y., Rijmen, V. (eds.) EUROCRYPT 2019, Part II. LNCS, vol. 11477, pp. 3–33. Springer, Heidelberg (May 2019). https://doi.org/10.1007/978-3-030-17656-3_1
14. Brakerski, Z., Döttling, N.: Two-message statistically sender-private OT from LWE. In: Beimel, A., Dziembowski, S. (eds.) TCC 2018, Part II. LNCS, vol. 11240, pp. 370–390. Springer, Heidelberg (Nov 2018). https://doi.org/10.1007/978-3-030-03810-6_14
15. Byali, M., Patra, A., Ravi, D., Sarkar, P.: Fast and universally-composable oblivious transfer and commitment scheme with adaptive security. Cryptology ePrint Archive, Report 2017/1165 (2017), https://eprint.iacr.org/2017/1165
16. Canetti, R.: Universally composable security: A new paradigm for cryptographic protocols. In: 42nd FOCS. pp. 136–145. IEEE Computer Society Press (Oct 2001). https://doi.org/10.1109/SFCS.2001.959888
17. Canetti, R., Sarkar, P., Wang, X.: Blazing fast OT for three-round UC OT extension. In: Kiayias, A., Kohlweiss, M., Wallden, P., Zikas, V. (eds.) PKC 2020, Part II. LNCS, vol. 12111, pp. 299–327. Springer, Heidelberg (May 2020). https://doi.org/10.1007/978-3-030-45388-6_11
18. Chou, T., Orlandi, C.: The simplest protocol for oblivious transfer. In: Lauter, K.E., Rodríguez-Henríquez, F. (eds.) LATINCRYPT 2015. LNCS, vol. 9230, pp. 40–58. Springer, Heidelberg (Aug 2015). https://doi.org/10.1007/978-3-319-22174-8_3
19. D'Anvers, J.P., Karmakar, A., Roy, S.S., Vercauteren, F.: Saber: Module-LWR based key exchange, CPA-secure encryption and CCA-secure KEM. In: Joux, A., Nitaj, A., Rachidi, T. (eds.) AFRICACRYPT 18. LNCS, vol. 10831, pp. 282–305. Springer, Heidelberg (May 2018). https://doi.org/10.1007/978-3-319-89339-6_16

20. Döttling, N., Garg, S., Hajiabadi, M., Masny, D., Wichs, D.: Two-round oblivious transfer from CDH or LPN. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020, Part II. LNCS, vol. 12106, pp. 768–797. Springer, Heidelberg (May 2020). https://doi.org/10.1007/978-3-030-45724-2_26
21. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Ladner, R.E., Dwork, C. (eds.) 40th ACM STOC. pp. 197–206. ACM Press (May 2008). https://doi.org/10.1145/1374376.1374407
22. Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game or A completeness theorem for protocols with honest majority. In: Aho, A. (ed.) 19th ACM STOC. pp. 218–229. ACM Press (May 1987). https://doi.org/10.1145/28395.28420
23. Green, M., Hohenberger, S.: Universally composable adaptive oblivious transfer. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 179–197. Springer, Heidelberg (Dec 2008). https://doi.org/10.1007/978-3-540-89255-7_12
24. Impagliazzo, R., Rudich, S.: Limits on the provable consequences of one-way permutations. In: 21st ACM STOC. pp. 44–61. ACM Press (May 1989). https://doi.org/10.1145/73007.73012
25. Ishai, Y., Kilian, J., Nissim, K., Petrank, E.: Extending oblivious transfers efficiently. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 145–161. Springer, Heidelberg (Aug 2003). https://doi.org/10.1007/978-3-540-45146-4_9
26. Ishai, Y., Prabhakaran, M., Sahai, A.: Founding cryptography on oblivious transfer - efficiently. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 572–591. Springer, Heidelberg (Aug 2008). https://doi.org/10.1007/978-3-540-85174-5_32
27. Kalai, Y.T.: Smooth projective hashing and two-message oblivious transfer. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 78–95. Springer, Heidelberg (May 2005). https://doi.org/10.1007/11426639_5
28. Keller, M., Orsini, E., Scholl, P.: Actively secure OT extension with optimal overhead. In: Gennaro, R., Robshaw, M.J.B. (eds.) CRYPTO 2015, Part I. LNCS, vol. 9215, pp. 724–741. Springer, Heidelberg (Aug 2015). https://doi.org/10.1007/978-3-662-47989-6_35
29. Kilian, J.: Founding cryptography on oblivious transfer. In: 20th ACM STOC. pp. 20–31. ACM Press (May 1988). https://doi.org/10.1145/62212.62215
30. Kolesnikov, V., Kumaresan, R.: Improved OT extension for transferring short secrets. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part II. LNCS, vol. 8043, pp. 54–70. Springer, Heidelberg (Aug 2013). https://doi.org/10.1007/978-3-642-40084-1_4
31. Lindell, A.Y.: Efficient fully-simulatable oblivious transfer. In: Malkin, T. (ed.) CT-RSA 2008. LNCS, vol. 4964, pp. 52–70. Springer, Heidelberg (Apr 2008). https://doi.org/10.1007/978-3-540-79263-5_4
32. Liu, F.H., Wang, Z.: Rounding in the rings. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020, Part II. LNCS, vol. 12171, pp. 296–326. Springer, Heidelberg (Aug 2020). https://doi.org/10.1007/978-3-030-56880-1_11
33. Masny, D., Rindal, P.: Endemic oblivious transfer. In: Cavallaro, L., Kinder, J., Wang, X., Katz, J. (eds.) ACM CCS 2019. pp. 309–326. ACM Press (Nov 2019). https://doi.org/10.1145/3319535.3354210
34. Masny, D., Watson, G.J.: A PKI-based framework for establishing efficient MPC channels. In: Vigna, G., Shi, E. (eds.) ACM CCS 2021. pp. 1961–1980. ACM Press (Nov 2021). https://doi.org/10.1145/3460120.3484806

35. McQuoid, I., Rosulek, M., Roy, L.: Minimal symmetric PAKE and 1-out-of-N OT from programmable-once public functions. In: Ligatti, J., Ou, X., Katz, J., Vigna, G. (eds.) ACM CCS 2020. pp. 425–442. ACM Press (Nov 2020). https://doi.org/10.1145/3372297.3417870
36. McQuoid, I., Rosulek, M., Roy, L.: Batching base oblivious transfers. In: Tibouchi, M., Wang, H. (eds.) ASIACRYPT 2021, Part III. LNCS, vol. 13092, pp. 281–310. Springer, Heidelberg (Dec 2021). https://doi.org/10.1007/978-3-030-92078-4_10
37. Naor, M., Pinkas, B.: Efficient oblivious transfer protocols. In: Kosaraju, S.R. (ed.) 12th SODA. pp. 448–457. ACM-SIAM (Jan 2001)
38. Naor, M., Pinkas, B.: Computationally secure oblivious transfer. Journal of Cryptology **18**(1), 1–35 (Jan 2005). https://doi.org/10.1007/s00145-004-0102-6
39. NIST: Official comments (round 3) - SABER. https://csrc.nist.gov/csrc/media/Projects/post-quantum-cryptography/documents/round-3/official-comments/SABER-round3-official-comment.pdf, (Accessed on 02/02/2024)
40. Orrù, M., Orsini, E., Scholl, P.: Actively secure 1-out-of-N OT extension with application to private set intersection. In: Handschuh, H. (ed.) CT-RSA 2017. LNCS, vol. 10159, pp. 381–396. Springer, Heidelberg (Feb 2017). https://doi.org/10.1007/978-3-319-52153-4_22
41. Patra, A., Sarkar, P., Suresh, A.: Fast actively secure OT extension for short secrets. In: NDSS 2017. The Internet Society (Feb / Mar 2017)
42. Peikert, C., Vaikuntanathan, V., Waters, B.: A framework for efficient and composable oblivious transfer. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 554–571. Springer, Heidelberg (Aug 2008). https://doi.org/10.1007/978-3-540-85174-5_31
43. Peter Rindal, L.R.: libOTe: an efficient, portable, and easy to use Oblivious Transfer Library. https://github.com/osu-crypto/libOTe
44. Quach, W.: UC-secure OT from LWE, revisited. In: Galdi, C., Kolesnikov, V. (eds.) SCN 20. LNCS, vol. 12238, pp. 192–211. Springer, Heidelberg (Sep 2020). https://doi.org/10.1007/978-3-030-57990-6_10
45. Rabin, M.O.: How to exchange secrets with oblivious transfer. Cryptology ePrint Archive, Report 2005/187 (2005), https://eprint.iacr.org/2005/187
46. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: Gabow, H.N., Fagin, R. (eds.) 37th ACM STOC. pp. 84–93. ACM Press (May 2005). https://doi.org/10.1145/1060590.1060603
47. Roy, L.: SoftSpokenOT: Quieter OT extension from small-field silent VOLE in the minicrypt model. In: CRYPTO 2022, Part I. pp. 657–687. LNCS, Springer, Heidelberg (Aug 2022). https://doi.org/10.1007/978-3-031-15802-5_23
48. Schwabe, P., Avanzi, R., Bos, J., Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schanck, J.M., Seiler, G., Stehlé, D., Ding, J.: CRYSTALS-KYBER. Tech. rep., National Institute of Standards and Technology (2022), available at https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022
49. Wang, X., Malozemoff, A.J., Katz, J.: EMP-toolkit: Efficient MultiParty computation toolkit. https://github.com/emp-toolkit (2016)
50. Yang, K., Weng, C., Lan, X., Zhang, J., Wang, X.: Ferret: Fast extension for correlated OT with small communication. In: Ligatti, J., Ou, X., Katz, J., Vigna, G. (eds.) ACM CCS 2020. pp. 1607–1626. ACM Press (Nov 2020). https://doi.org/10.1145/3372297.3417276

51. Yao, A.C.C.: How to generate and exchange secrets (extended abstract). In: 27th FOCS. pp. 162–167. IEEE Computer Society Press (Oct 1986). https://doi.org/10.1109/SFCS.1986.25

A SoftSpoken OT Extension with Relaxed OT Functionality

A.1 Preliminaries and Notations

SoftSpokenOT [47] is a generalization of the classic oblivious transfer extensions of IKNP [25], which can be viewed based on $\mathbb{F}_2$-VOLE by using a PRG to extend $\binom{2}{1}$-OT to message size N. SoftSpoken instead bases the OT extension on a $\mathbb{F}_{2^k}$-VOLE. Instead of $\binom{2}{1}$-OT, base OTs for this construction are $\binom{2^k}{2^k-1}$-OTs. Using SoftSpoken, we now only need λ/k of these $\mathbb{F}_{2^k}$-VOLEs to perform OT extension from base OTs with λ-bit messages. The technical overviews are as follows.

We follow the notations of [47] to use λ as the security parameter of SoftSpoken OT extension. We denote the sender of the base OT with P_S and the receiver with P_R. For $\binom{2^k}{2^k-1}$-OT we define a random function $F : \mathbb{F}_{2^k} \to \mathbb{F}_2^\lambda$ which is known to P_S, while P_R has a random point Δ and the restriction F^* of F with $\mathbb{F}_{2^k}$. Therefore, the sender P_S can compute the function on all 2^k elements in field $\mathbb{F}_{2^k}$, but the receiver can only compute all except $F(\Delta)$. This fits in the functionality of $\binom{2^k}{2^k-1}$-OT. The base OT sender P_S computes two vectors $\boldsymbol{u}, \boldsymbol{v}$ as follows:

$$\boldsymbol{u} = \bigoplus_{y \in \mathbb{F}_{2^k}} F(y)$$

$$\boldsymbol{v} = \bigoplus_{y \in \mathbb{F}_{2^k}} yF(y)$$

And the receiver computes a vector $\boldsymbol{w}$ from restricted F^*:

$$\boldsymbol{w} = \bigoplus_{y \in \mathbb{F}_{2^k}} (y \oplus \Delta)F^*(y)$$

Therefore, the vectors mentioned above satisfies a $\mathbb{F}_{2^k}$-VOLE, i.e. $\boldsymbol{w} \oplus \boldsymbol{v} = \bigoplus_y \Delta F(y) = \Delta \cdot \boldsymbol{u}$. In the following protocol description, we denote the i-th column of a matrix A with A^i, and the j-th row of a column with A^j. Diagonal matrices are noted $\mathrm{diag}(\boldsymbol{x} = \{x_0, \ldots, x_{n-1}\}) = \begin{bmatrix} x_0 & & \\ & \ddots & \\ & & x_{n-1} \end{bmatrix}$.

A.2 SoftSpoken OT Extension Protocol from $\Pi_{\mathsf{SF-rOT}}$

In this section we instantiate the SoftSpoken OT extension using λ invocations of $\binom{2^k}{2^k-1}$-OT functionality, each of which comes from the relaxed $\binom{2}{1}$-OT functionality $\mathcal{F}_{\mathsf{SF-rOT}}$ in Fig. 3. To do this, we need to change $\binom{2}{1}$-OT functionalities

$\mathcal{F}_{\mathsf{SF-rOT}}$'s to a $\binom{2^k}{2^k-1}$ one using a punctured PRF, which is given as a intermediate step in [47]. The details are unfolded in the **Seed OT Phase I** of $\Pi_{\mathsf{SoftSpoken}}$ in Fig. 8.

Our goal is to implement N extended random OT. We assume that there exists an arbitrary $\binom{2}{1}$-OT protocol $\Pi_{\mathsf{SF-rOT}}$ to implement λ instances of $\binom{2^k}{2^k-1}$-OT functionalities (e.g. we can use a punctured PRF to transfer Saber-based $\binom{2}{1}$-OT in Fig. 4 to a valid instantiation). This construction needs λ/k instances of $\binom{2^k}{2^k-1}$-OTs, thus λ instances of $\binom{2}{1}$-OTs in total.

Let $\mathsf{PRG} : \{0,1\}^\lambda \to \{0,1\}^{2\lambda}$ be a length-doubling pseudorandom generator. We use PRG_0 and PRG_1 to denote the functions that output the first and second halves of PRG. Let $k \in \mathbb{N}$ be the GGM-tree depth. We define the punctured GGM tree algorithms as follows.

- $\mathsf{pGGM.Gen}$: Given input seed $s \in \{0,1\}^\lambda$ the generation algorithm runs as follows(we use z as intermediate results of punctured PRF):
  1. Denote $r^{0,0} = s$, for $i \in [k], j \in [2^i]$, generate $r^{i+1,2j} = \mathsf{PRG}_0(r^{i,j})$ and $r^{i+1,2j+1} = \mathsf{PRG}_1(r^{i,j})$.
  2. Define $F : \mathbb{F}_{2^k} \to \{0,1\}^\lambda$ such that $F(i) = r^{k,i}$.
  3. For $i \in [k]$, define $t_0^i = \oplus_{j\in[2^{i-1}]} r^{i,2j}$ and $t_1^i = \oplus_{j\in 2^{i-1}} r^{i,2j+1}$.
  4. Return $F, \boldsymbol{t} = \{t_0^i, t_1^i\}_{i\in[k]}$.
- $\mathsf{pGGM.Eval}$: Given the punctured key $\boldsymbol{t} = \{t^i_{x_i}\}$ for $i \in [k]$ and the path $\boldsymbol{x} = (\bar{x}_1, ..., \bar{x}_k)$ the evaluation algorithm runs as follows.
  1. Define $r^1_{x_1} = t^1_{x_1}$. For $i \in [2,k]$, $j \in [2^{i-1}]$ and $j \neq x_1 \| \ldots \| x_{i-1}$, evaluate $r^i_{2j} = \mathsf{PRG}_0(r_j^{i-1})$ and $r^i_{2j+1} = \mathsf{PRG}_1(r_j^{i-1}) \oplus r_j^{i-1}$ and defines $r^i_{x_1\|\ldots\|x_{i-1}\|\bar{x}_i} = t^i_{\bar{x}_i} \oplus \oplus_{j\in[2^{i-1}], j\neq x_1\|\ldots\|x_{i-1}} r^i_{2j+\bar{x}_i}$.
  2. Outputs F^* such that $F^*(j) = r_j^k$ for $j \neq \Delta$.

Fig. 7. The Punctured GGM Tree Constructions.

We denote a batch of k $\Pi_{\mathsf{SF-rOT}}$'s with $\Pi^k_{\mathsf{SF-rOT}}$, and there are λ/k different batches. We denote the all-but-one OT protocols with using the technique in [47]. The sender of OT extension is denoted by S, invoking each of the base $\binom{2^k}{2^k-1}$-OTs as receiver; while the receiver of the OT extension R invokes base OTs as sender. From these all-but-one OT, we can get λ/k different $\mathbb{F}_{2^k}$-VOLE correlations. Then we use the technique of repetition code to make $\boldsymbol{u}$ monochrome and combine the small field VOLE correlations into a VOLE correlation over the large field $\mathbb{F}_{2^\lambda}$. The details of this part is shown in the **OT Extension Phase I** of $\Pi_{\mathsf{SoftSpoken}}$ in Fig. 9.

Protocol $\Pi_{\mathsf{SoftSpoken}}$ from $\Pi_{\mathsf{SF-rOT}}$, (Part I)

Parameters: $N, k, \lambda \in \mathbb{N}$.
Sub-procedures: The pseudorandom generator PRG, base-OT functionality $\mathcal{F}_{\mathsf{SF-rOT}}$, and random oracle functionality $\mathcal{F}_{\mathsf{RO}}$.
Inputs: The parties does not have private input.
Outputs: S outputs N pairs of random messages $\{a_{i,0}, a_{i,1}\}_{i\in[N]}$ while R outputs N choice bits $\boldsymbol{u} \in \mathbb{F}_2^N$ and the chosen messages $\{a_{i,u_i}\}_{i\in[N]}$.

Seed $\binom{2^k}{2^k-1}$ base OTs: The parties run λ/k instance of all-but-one OT from $\mathcal{F}_{\mathsf{SF-rOT}}$. In the following we describe the i-th instance for $i \in [\lambda/k]$.

1. For $j \in [k]$, R invokes the j-th batch $\Pi^{k,i}_{\mathsf{SF-rOT}}$ with message $(\textsc{Transfer}, \mathsf{sen}, \mathsf{sid})$ to obtain random messages $m_{i,j,0}, m_{i,j,1} \in \{0,1\}^\lambda$.
2. R samples $s_i \leftarrow \{0,1\}^\lambda$ and computes $(F_i, \{(t_0^{i,j}, t_1^{i,j})\}) \leftarrow \mathsf{pGGM.Gen}(s_i)$.
3. S invokes the j-th batch $\Pi^{k,i}_{\mathsf{SF-rOT}}$ with message $(\textsc{Choose}, \mathsf{rec}, \mathsf{sid})$ to obtain its choice bit $x_{i,j}$ and $\{m_{i,j,x_{i,j}}\}$. S computes its choice point $\Delta_i = \sum_{j\in[k]} \bar{x}_{i,j} \cdot X^j \in \mathbb{F}_{2^k}$.
4. If S receives any $\textsc{Cheat-Detected}$ messages from $\Pi^{k,i}_{\mathsf{SF-rOT}}$, then he aborts.
5. R sends $\{\mathcal{F}_{\mathsf{RO}}(m_{i,j,0}) \oplus t_0^{i,j}, \mathcal{F}_{\mathsf{RO}}(m_{i,j,1}) \oplus t_1^{i,j}\}$ to S, who then recovers $\{t^j_{x_{i,j}}\}$.
6. R computes $F_i^* = \mathsf{pGGM.Eval}(\{t^j_{x_{i,j}}\}, \Delta_i)$.

Consistency Check Phase I:

1. For each $i \in [\lambda/k]$, R computes challenge $\alpha_{i,y} := \mathcal{F}_{\mathsf{RO}}(F_i(y))$ for each $y \in [2^k]$, $\alpha_i = \mathcal{F}_{\mathsf{RO}}(\alpha_{i,0} \| \dots \| \alpha_{i,2^k-1})$, $\beta_i := \bigoplus_{y\in[2^k]} \alpha_{i,y}$,and sends $\{\alpha_i, \beta_i\}_{i\in[\lambda/k]}$ to S.
2. For each $i \in [\lambda/k]$, S computes $\alpha^*_{i,y} := \mathcal{F}_{\mathsf{RO}}(F_i^*(y))$ for each $y \in [2^k] \setminus \{\Delta_i\}$, then computes the one at Δ_i using the P_S^i's challenge i.e. $\alpha^*_{i,\Delta_i} = \beta_i \oplus \bigoplus_{y\in[2^k]\setminus\{\Delta_i\}} \alpha^*_{i,y}$, at last computes $\alpha_i^* = \mathcal{F}_{\mathsf{RO}}(\alpha^*_{i,0} \| \dots \| \alpha^*_{i,2^k-1})$. P_R^i checks whether $\alpha_i = \alpha_i^*$ holds, and he aborts if the check fails.

Fig. 8. SoftSpoken OT Extension with $\Pi_{\mathsf{SF-rOT}}$

A.3 Security Proof

We prove UC-security of our OT extension protocol $\Pi_{\mathsf{SoftSpoken}}$ by relying on the security properties of $\Pi_{\mathsf{SF-rOT}}$ and the security of cryptographic primitives e.g. PRG and RO in Theorem 2.

Theorem 3 (Theorem 2, restated). *In $\Pi_{\mathsf{SoftSpoken}}$, assuming PRGs are secure pseudorandom generator, $\mathcal{R}$ is an ϵ-universal hash family, $\mathcal{F}_{\mathsf{RO}}$ is a hash function satisfying the functionality of observable random oracle and $\Pi^k_{\mathsf{SF-rOT}}$ implements k instances of $\mathcal{F}_{\mathsf{SF-rOT}}$, then $\Pi_{\mathsf{SoftSpoken}}$ UC-securely implements $N = \mathsf{poly}(\lambda)$ instances of extended OT functionality.*

Proof. We will first argue security against a corrupt sender S^* by constructing a simulator Sim in Fig. 10. The simulator Sim for a statically corrupt sender S^*

Protocol $\Pi_{\mathsf{SoftSpoken}}$ from $\Pi_{\mathsf{SF-rOT}}$, (Part II)

OT Extension Phase I:

1. R forms two matrices $\mathbf{U} \in \mathbb{F}_2^{N\times(\lambda/k)}$ and $\mathbf{V} \in F_{\mathbb{F}_{2^k}}^{N\times(\lambda/k)}$, where for $i \in [\lambda/k]$ the i-th columns are $U^i = \bigoplus_{y\in\mathbb{F}_{2^k}} \mathsf{PRG}(F_i(y))$, $V^i = \bigoplus_{y\in\mathbb{F}_{2^k}} y \cdot \mathsf{PRG}(F_i(y))$
2. S forms a matrix $\mathbf{W} \in \mathbb{F}_{2^k}^{N\times(\lambda/k)}$, where for the i-th column $\mathbf{W}^i = \bigoplus_{y\in\mathbb{F}_{2^k}} (y \oplus \Delta_i) \cdot \mathsf{PRG}(F_i^*(y))$. It also computes a diagonal matrix $\mathrm{diag}(\boldsymbol{\Delta}) = \mathrm{diag}(\{\Delta_1, \ldots, \Delta_{\lambda/k}\}) \in \mathbb{F}_{2^k}^{(\lambda/k)\times(\lambda/k)}$.
3. To make $\mathbf{U}$ monochrome, R uses the first column of $\mathbf{U}$ as chosen bits i.e. $\boldsymbol{u} = \mathbf{U}^1$ and computes a difference matrix $\mathbf{D} = (\boldsymbol{u} \oplus U^2, \ldots, \boldsymbol{u} \oplus U^{\lambda/k})$ and send it to S, who computes $\mathbf{W} := \mathbf{W} \oplus (0\|\mathbf{D}) \cdot \mathrm{diag}(\boldsymbol{\Delta})$.

Consistency Check Phase II:

1. S samples a universal hash function R and sends to R. R computes $\{\Gamma_U := R\tilde{\boldsymbol{u}}, \Gamma_V := RV\}$ and sends it to S.
2. S checks that $\Gamma_V = RW' - \Gamma_U(1, \ldots, 1) \cdot \mathrm{diag}(\boldsymbol{\Delta})$ and aborts if the check fails.

OT Extension Phase II:

1. For every $i \in [N]$, S outputs $a_{i,0} = \mathcal{F}_{\mathsf{RO}}(W_i)$ and $a_{i,1} = \mathcal{F}_{\mathsf{RO}}(W_i \oplus \boldsymbol{\Delta})$.
2. For every $i \in [N]$, R outputs u_i and $a_{i,u_i} = \mathcal{F}_{\mathsf{RO}}(V_i)$.

Fig. 9. SoftSpoken OT Extension with $\Pi_{\mathsf{SF-rOT}}$, continued

constructs the U and V as the honest receiver does. Since the base OT $\Pi_{\mathsf{SF-rOT}}$ securely implements $\mathcal{F}_{\mathsf{SF-rOT}}$, S^* cannot get both of the messages, thus the all-but-one OTs' message indexed by Δ_i remains hidden. Sim invokes Ext to extract base OTs' choice bits $\{\boldsymbol{x}\}_{i\in[\lambda/k]}$ and computes S^*'s messages.

We argue indistinguishability between real world and ideal world simulated by Sim in Fig. 10 by providing the following hybrid experiments.

Hybrid 1 This is real-world experiment.

Hybrid 2 In this hybrid, the simulator sends a uniformly random matrix $\mathbf{D}$ in **OT Extension Phase I**. By the input privacy of the $\Pi_{\mathsf{SF-rOT}}$, this step does not change the output of the experiment.

Hybrid 3 In this hybrid, the simulator Sim extracts the input bits $\{\boldsymbol{x}_i\}_{i\in[\lambda/k]}$ of $\Pi_{\mathsf{SF-rOT}}^{k,i}$ for every $i \in [\lambda/k]$ using Ext in Fig. 6. Indistinguishability follows due to the correctness of Ext. Ext ensures that the receivers of base OTs does not distinguish this simulation and thus will not abort, so that S^* of OT extension will not abort, either.

Hybrid 4 In this hybrid, Sim computes the matrix $\mathbf{W}$ and $\{a_{i,0}, a_{i,1}\}_{i\in[N]}$ using the input bits extracted. This hybrid is identical to the last one because of the correctness of the OT extension protocol.

Sim for S^*

Seed $\binom{\mathbf{2^k}}{\mathbf{2^k-1}}$ base OTs:

1. Sim invokes batched $\Pi^k_{\mathsf{SF-rOT}}$ for λ/k times as sender with messages (TRANSFER, sen, sid) trying to obtain the random functions $F_i : \mathbb{F}_{2^k} \to \mathbb{F}_2^\lambda$ for every $i \in [\lambda/k]$, and reveals parts of the GGM tree as the honest receiver does.
2. If Sim gets ABORT message from any of the $\Pi_{\mathsf{SF-rOT}}$, then he aborts.
3. Sim invokes the Ext algorithm 6 for the S^*'s base OT receivers and gets the S^*'s input bits $\{\boldsymbol{x}_i\}_{i\in[\lambda/k]}$ for the base OT receivers. Thus Sim reconstructs the chosen elements $\boldsymbol{\Delta}$ for the $\binom{2^k}{2^k-1}$-OTs.
4. Sim computes $W^i = \bigoplus_{y\in\mathbb{F}_{2^k}\setminus\{\Delta_i\}} y \cdot \mathsf{PRG}_2(F_i(y))$, then he updates to $W'^i = W^i \oplus ((\tilde{\boldsymbol{u}},\ldots,\tilde{\boldsymbol{u}}) \oplus U) \cdot \mathrm{diag}(\boldsymbol{\Delta})$,

Consistency Check Phase I: Sim changes the challenge $\{\alpha_i, \beta_i\}_{i\in[\lambda/k]}$ to random bits $\{\alpha'_i, \beta'_i\}_{i\in[\lambda/k]} \xleftarrow{\$} \{0,1\}^{k\times 2\lambda}$ and programs the observable RO modeled by $\mathcal{F}_{\mathsf{RO}}$ s.t. $\mathcal{F}_{\mathsf{RO}}(\alpha_{i,0}\| \ldots \|\alpha_{i,2^k-1}) = \alpha'_i$ when S^* access $\mathcal{F}_{\mathsf{RO}}$, and sends them to S^*.
OT Extension Phase I: Sim sends a uniformly random $\mathbf{D}$ matrix to S^*.
Consistency Check Phase II: Sim receives the hash function $R \in \mathcal{R}$ from S^* changes the challenge Γ_U and Γ_V from the honest receiver to $\Gamma'_U \xleftarrow{\$} \{0,1\}^{\ell(\epsilon)}, \Gamma'_V = RW' - \Gamma'_U(1,\ldots,1)\cdot\mathrm{diag}(\boldsymbol{\Delta})$ and sends them to S^*.
OT Extension Phase II:

1. Sim has obtained W' using the extractor, so that it can extract S^*'s random messages $\{a_{n,0}, a_{n,1}\}_{n\in[N]}$ from every row of W'.
2. Sim invokes the n-th ideal functionality $\mathcal{F}_{\mathsf{OT}}$ for N times with inputs $\{a_{n,0}, a_{n,1}\}_{n\in[N]}$.

Fig. 10. Simulation against a statically corrupt sender S^*

Hybrid 5 In this hybrid, Sim uses uniformly random vectors $\{\alpha'_i, \beta'_i\}_{i\in[\lambda/k]}$ to simulate the consistency check of the all-but-one base OTs. Indistinguishability comes from the security properties of observable random oracle H, and S^* verifies correctly and does not abort.

Hybrid 6 In this hybrid, the simulator uses uniformly random matrices Γ_U, Γ_V to simulate the consistency check. By the hiding property of the universal hash function, this does not change the output of the experiment. This is the ideal world interaction.

The security against a corrupted receiver is easier to argue, since in OT extension the receiver acts as the sender in the base OT, and the weakened security of $\mathcal{F}_{\mathsf{SF-rOT}}$ ensures simulation-based security against a corrupted sender. Therefore, the simulation strategy is a combination of the respective simulation strategies of the base OT and the OT extension protocols.