

Article

Universally Composable Oblivious Transfer with Low Communication

Jiashuo Song, Dongfei Wang *, Zhenzhen Zhang, Zhenzhen Li, Haiyang Ding and Zichen Li

Digital Rights Management Research Center, Beijing Institute of Graphic Communication, Beijing 102600, China

* Correspondence: wdfchina@bigc.edu.cn; Tel.: +86-17-701-203-450

Abstract: In this paper, a universally composable 1-out-of-N oblivious transfer protocol with low communication is built. This protocol obtained full simulation security based on the modulo learning with rounding (Mod-LWR) assumption. It can achieve universally composable security in the random oracle machine (ROM) model by combining random OT based on the key exchange protocol with the authentication encryption algorithm. It can be proven to resist static adversary attacks by simulating all corruption cases. Based on computer simulation and detailed mathematical derivation, this protocol was practicable and had better efficiency and lower communication.

Keywords: oblivious transfer (OT); universal composability (UC); learning with rounding assumption; random oracle model (ROM); static adversary attacks

1. Introduction

With the continuous updating and iteration of secure multi-party computation (MPC), many secure multi-party applications have been recently developed and used. For example, Cheetah [1], a new research achievement of Ali, is a secure two-party computing framework that can improve the performance of MPC. It has been used in the risk control field. In addition, many companies and institutes also participate in designing and developing privacy computing platforms [2,3], such as the morse secure computing platform developed by Fu-min bank.

Secure multi-party computation is not an independent technology. However, it includes many fundamental primitives such as secret sharing [4], homomorphic encryption [5], zero-knowledge proof [6], and bit commitment [7], as well as many secure multi-party applications based on the primitives above, such as private set insertion [8], private information retrieval, and secret geometry computing [9]. Besides, all of the efficient MPC applications are based on OT or OT extension. Thus, efficient OT protocols with low communication costs are the focus and difficulty in MPC research.

Oblivious transfer [10], an essential and fundamental primitive of MPC, was first proposed by Rabin in 1981. There are two participants in the OT protocol: the sender (S) and receiver (R). S sends R a message, and R has a $1/2$ probability of receiving this message. Besides, S does not know whether R has received the message. In 1995, Even proposed a random 1-out-of-2 oblivious transfer [11] based on public-key cryptography, where the receiver can obtain only one of the two messages from the sender randomly. In 1986, Brassard expanded 1-out-of-2 OT to 1-out-of-N OT [12] to improve the efficiency of the OT protocol. Tzeng proposed a k-out-of-n OT protocol [13] based on the DDH (decisional Diffie–Hellman) difficult problem. It allows the receiver to obtain k of n messages from the sender to make the OT protocol more efficient and practical.

The OT protocol is the bottleneck of the efficiency and communication of the MPC because of its importance. In 2001, Naor and Pinkas proposed an efficient two-round 1-out-of-N OT protocol [14] based on the Diffie–Hellman difficult problem. However, it can obtain half-simulation security and construct a simulator for one party in the ideal environment. In 2008, Damgård proposed an attack method against MPC protocols [15] in the half-simulation paradigm, proving that OT protocols with half-simulation security



Citation: Song, J.; Wang, D.; Zhang, Z.; Li, Z.; Ding, H.; Li, Z. Universally Composable Oblivious Transfer with Low Communication. *Appl. Sci.* **2023**, *13*, 2090. <https://doi.org/10.3390/app13042090>

Academic Editor: Nuno Silva

Received: 17 November 2022

Revised: 3 February 2023

Accepted: 3 February 2023

Published: 6 February 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

cannot be used sequentially in combination with other protocols. Hence, the OT protocol is required to prove its security in a fully simulation-based paradigm, where the simulator can be constructed in any corruption case. In 2008, Yehuda proposed an OT protocol [16] with full-simulation security based on the cut-and-choose technology.

In 2001, Canetti proposed the concept of the universally composable (UC) security model [17] to explain and prove the security of the protocol combined with others. From then to now, many universally composable protocols have been designed. In 2019, Peikert et al. proposed an OT protocol based on dual-mode encryption [18], which obtains UC security under the LWE (learning with errors) assumption. In 2020, Lai proposed a UC-secure isogeny-based protocol [19] under ROM.

Most OT protocols are based on number theory, such as the discrete logarithm problem and the substantial number decomposition problem. Considering that these problems may be solvable for quantum computers in an efficient manner, the post-quantum MPC protocol is a hot spot and the focus of contemporary cryptography research. For example, the United States included post-quantum cryptography in the latest revision of the Critical and Emerging Technologies List on 8 February 2022. Post-quantum cryptography is divided into lattice-based cryptography and code-based cryptography, among others. The lattice-based cryptographic protocols have received more attention owing to their provable security and lower asymptotic complexity. In 2019, Pedor proposed a framework for the OT protocol [20] under the UC security model based on key exchange protocol and symmetric encryption. Based on the Pedor's framework, Chou constructed a UC-secure OT protocol [21]. In addition, Liu proposed an efficient UC-secure OT protocol [22] based on the NewHope key exchange protocol [23] on the lattice. Quach proposed a two-round UC-secure OT protocol [24] under the CRS model. It achieved statistical sender security and statistical receiver security under the LWE assumption. The errors in the LWE difficult problem are not deterministic, which must be added. However, the errors in the LWR difficult problem are generated by scaling and rounding. Besides, you can solve the LWE problem with an LWR RO machine under special parameter settings.

The OT protocol is one of the most used fundamental cryptographic primitives in secure multi-party computation. The main problems of today's oblivious transfer protocols are as follows:

1. Most existing OT protocols are based on the number theory and cannot resist quantum computer attacks.
2. Most existing OT protocols can only achieve full or half simulation security in the stand-alone model.
3. The efficient MPC protocols with low communication costs [25] have become the research focus.

1.1. Our Contribution

Hence, a low communication OT protocol on lattice under Mod-LWR assumption was proposed in this paper. It obtained full-simulation security in the ROM. In addition, we extended it to a 1-out-of-N OT protocol with a minor tweak. Finally, we simulated this protocol and compared it with other protocols in the theoretical aspect. The simulation shows that the average communication and running time are only 2.45 kb and 0.5 ms per time, respectively. The comparison indicates that the protocol has the advantages of high efficiency and low communication.

1.2. Organization

This paper is organized as follows. Section 1 briefly introduces the technical background and Section 2 provides a detailed introduction to the notations and techniques used in the article. In Section 3, the components of the OT protocol are introduced. The 1-out-of-2 and 1-out-of-N OT protocols are presented in Sections 4 and 5, respectively. Section 6 shows the results and the comparison with other protocols. Finally, a conclusion is given in Section 7.

2. Preliminaries

In this section, we introduce the notations and the techniques used in the paper.

2.1. Notations

Denote Z_q as the ring of integers modulo an integer q . For any integer z , we define $z \bmod q$ as the reduction of z in $[0, q)$ and define $p \mid q$ as the modulo operator on R_q by modulo of all the coefficients. R_q is a quotient ring $Z_q[X]/(X^n + 1)$ with n , where n is a fixed power of 2. For a ring R , we define $R^{l \times k}$ as the ring of $l \times k$ -matrices over R .

We denote $\mathcal{U}$ as the uniform distribution and denote β_μ as the centered binomial distribution with parameter μ , where the corresponding standard deviation $\sigma = \sqrt{\mu/2}$. If $\mathcal{X}$ is the probability distribution over a set S , we denote $x \leftarrow \mathcal{X}$ as sampling from the set S according to $\mathcal{X}$. If $\mathcal{X}$ is defined on Z_q , $X \leftarrow \mathcal{X}(R_q^{l \times k})$ denotes that all coefficients are sampling from the set $R_q^{l \times k}$ according to $\mathcal{X}$.

2.2. MOD-LWR Problems

The LWR (learning with rounding) assumption was proposed by Banerjee to construct the pseudorandom function. Compared with the learning with error assumption, the errors in the LWR assumption are generated by scaling and rounding all of the coefficients from modulo q to p (with $p < q$). For a fixed $s \leftarrow \beta_\mu(R_q^{l \times 1})$ and a uniform random $a \leftarrow \mathcal{U}(Z_q^{l \times 1})$, the LWR sample is as follows:

$$\left(a, b = \left\lfloor \frac{p}{q} (a^T s) \right\rfloor\right) \in Z_q^{l \times 1} \times Z_p \quad (1)$$

Definition 1 (the decisional LWR problem). *It is hard to distinguish LWR distribution samples from uniform distribution samples.*

The Mod-LWR assumption is the variant of LWR, which replaces the ring Z_p with a quotient ring R_p . Then, for a fixed $s \leftarrow \beta_\mu(R_q^{l \times 1})$ and a uniform random $a \leftarrow \mathcal{U}(R_q^{l \times 1})$, the Mod-LWR sample is as follows:

$$\left(a, b = \left\lfloor \frac{p}{q} (a^T s) \right\rfloor\right) \in R_q^{l \times 1} \times R_p \quad (2)$$

For fixed integers m, k, μ, p, q , the advantage of an adversary $\mathcal{A}$ in distinguishing from samples from a Mod-LWR distribution and that from the uniform distribution are defined as follows:

$$Adv_{m,l,\mu,q,p}^{Mod-LWR}(\mathcal{A}) = \left| \Pr \left(b' = 1 : \begin{array}{l} A \leftarrow \mathcal{U}(R_q^{m \times l}); s \leftarrow \beta_\mu(R_q^{l \times 1}); \\ b' = A(A, \lfloor (p/q) A_s \rfloor); \end{array} \right) - \Pr \left(b' = 1 : \begin{array}{l} A \leftarrow \mathcal{U}(R_q^{m \times l}); u \leftarrow \mathcal{U}(R_q^{l \times 1}); \\ b' = A(A, u); \end{array} \right) \right| \quad (3)$$

2.3. Universally Composable Security Model

To describe and analyze the security of cryptography protocols in parallel concurrency, Canetti proposed the universally composable security framework, which specifies the existence of $\mathcal{A}$ (adversary), $\mathcal{S}$ (simulator), and $\mathcal{Z}$ (environment) in addition to the parties of the protocol. The protocol can prove universally composable security by showing the indistinguishability between the ideal environment and real execution.

In the real execution, π denotes the running protocol, and $\mathcal{A}$ denotes the adversary who corrupts the participant to attack the protocol. In the ideal environment, $\mathcal{F}$ is the corresponding function of π , and $\mathcal{S}$ is the corresponding simulator of $\mathcal{A}$. Besides, $\mathcal{Z}$ can be considered as the external environment that can communicate directly with the protocol participants, the adversary, and the simulator except for the ideal function.

Definition 2 (UC-Secure). *If, for any adversary in the real execution, there exists a corresponding simulator in the ideal environment such that the environment cannot distinguish between interacting with a protocol in the real execution or interacting with an ideal function in the ideal environment, and the formula is as follows:*

$$IDEAL_{\mathcal{F}, \mathcal{S}, \mathcal{Z}} \stackrel{c}{=} EXEC_{\pi, \mathcal{A}, \mathcal{Z}} \quad (4)$$

Thus, it can be proven that the protocol π can safely simulate the function $\mathcal{F}$.

3. Constitutions of Protocol

In this section, we introduce the constructions of the OT protocol in this paper.

3.1. Key Exchange Protocol

The key exchange protocol used in this paper has two integral roles:

1. The parties can finally agree on a unanimous key.
2. The final key is indistinguishable from random strings.

The Saber protocol [26] based on the Mod-LWR difficult problem is shown in Table 1. It is one of the candidate algorithms for the post-NIST quantum cryptography competition.

Table 1. Saber key exchange protocol.

Alice	Bob
$seed_a \leftarrow \mathcal{U}(\{0, 1\}^{256})$ $a \leftarrow gen(seed_a) \in R_q^{l \times l}$ $s \leftarrow \beta_\mu(R_q^{l \times 1})$ $A = bits(as + h, \in_q, \in_p) \leftarrow R_q^{l \times 1}$	$seed_{q,A} \xrightarrow{\quad}$ $a \leftarrow gen(seed_a) \in R_q^{l \times l}$ $s' \leftarrow \beta_\mu(R_q^{l \times 1})$ $b = bits(A^T s' + h, \in_q, \in_p) \in R_q^{l \times 1}$ $v' = A^T bits(s', \in_p, \in_p) + h_1 \in R_p$ $c = bits(v', \in_p - 1, \in_t) \in R_t$
$v = b^T bits(s, \in_p, \in_p) + h_1 \in R_p$ $k = bits(v - 2^{\epsilon_p - \epsilon_t - 1} c + h_2, \in_p, 1)$	$k' = bits(v', \in_p, 1)$

Besides the basic symbols described above, many constants are also used in this protocol that can reduce the probability of failure: $h \in R_q^{l \times 1}$ is a constant polynomial vector whose coefficients are equal to $2^{\epsilon_q - \epsilon_p - 1}$; $h_1, h_2 \in R_q$ are constant polynomials; and the efficiencies of h_1 and h_2 are $2^{\epsilon_q - \epsilon_p - 1}$ and $2^{\epsilon_p - 2} - 2^{\epsilon_p - \epsilon_t - 2}$, respectively.

Saber uses a function $bits(x, i, j)$ shown in Figure 1, which is used to obtain the continual j bits of the x from the index i to create an integer in $\mathbb{Z}_{2^j}$. We can apply $bits(x, i, j)$ to a matrix or polynomial by all coefficients. Unless all of the coefficients are divisible by a higher power of 2, the final result can be considered sampled from the uniform distribution. However, in our protocol, the coefficients are generated from a binomial distribution, so the event that causes the result to deviate is that the coefficients are both 0. By analysis, the probability is negligible. Hence, Saber can meet the requirements of the OT protocol.

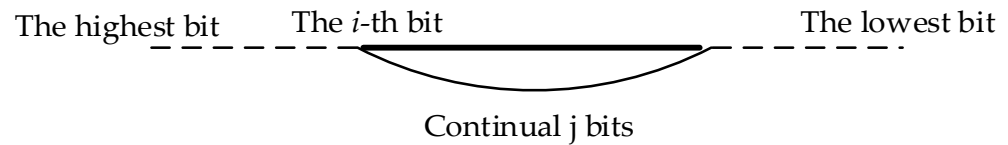


Figure 1. The schematic of $\text{bits}(x, i, j)$ function.

3.2. Hash Function

The OT protocol also needs a hash function to extend a key of length l bits from a string of n bits. The hash function is simulated by a random oracle machine. However, multiple UC-secure OT protocols should be able to run concurrently, so we use the local random oracle machine. The local random oracle machine can use intermediate values such as seed_a, A to distinguish different queries. Hence, the hash function is defined as follows: $\text{Hash}\left(\left(R_q^{l \times 1} \times R_q^{l \times 1}\right) \times R_t\right) \times R_q^n \rightarrow R_2^l$. This way, the protocol parties can query different results for different instances.

3.3. Symmetric Encryption Function

A standard UC-secure OT function consists of a UC-secure ROT function and an encryption function. To ensure the confidentiality and integrity of the information during the encryption and decryption of the OT protocol, we use an authentication symmetric encryption function. The authentication encryption algorithm [27] can achieve both encryption and authentication.

Like the hash function, we use the intermediate value $\left(\left(R_q^{l \times 1} \times R_q^{l \times 1}\right) \times R_t\right) \times R_q^n$ as an additional input to the authentication symmetric encryption function to distinguish among instances. As a component of UC-secure OT protocol, the authenticated symmetric encryption function has the character of non-committing and robustness. Let $\mathcal{K}$, $\mathcal{M}$, and $\mathcal{C}$ represent the authenticated encryption's key, plaintext, and ciphertext space, respectively.

Definition 3 (Non-committing). We say an authentication encryption scheme (Enc, Dec) is non-committing if there exists PPT algorithm $\mathcal{S}_1, \mathcal{S}_2$ such that (e, k) and (e', k') are distinguishable where $k \leftarrow \mathcal{K}$, $r \leftarrow \mathcal{U}(\cdot)$, $e \leftarrow E(k, r, M)$, $e' \leftarrow \mathcal{S}_2(1^k)$, and $k' \leftarrow \mathcal{S}_2(e', r, M)$.

Definition 4 (Robustness). Given additional input r , S is a set of random keys from $\mathcal{K}$. Let $V_{S, r, e}$ denote a set of valid keys that satisfies $\text{Dec}(k, r, e) \neq \perp$, for the given ciphertexts r, e , i.e., We say (Enc, Dec) is robust if, for every ciphertext, the probability of $|V_{S, e}| > 1$ is negligible.

As the ciphertexts of common symmetric encryption algorithms do not contain any commitment to the plaintext messages or encryption process, it is impossible to determine whether the key inputted for decryption is correct. However, if the authentication code obtained by decrypting the ciphertext with the inputted key does not match the authentication sequence obtained by encrypting, the algorithm directly outputs "Failure." and does not output the wrong decryption result. Hence, it has higher security than common symmetric encryption.

The protocol is robust because the probability of getting the same authentication code using different keys and additional inputs is negligible.

4. UC-Secure 1-out-of-2 OT Protocol

A UC-secure OT protocol based on the Mod-LWR problem can be obtained using the above technologies.

In this section, the ROT protocol is described in detail at first. Then, the OT protocol is introduced and, finally, the security of the OT protocol against static attacks is demonstrated under the random oracle model.

4.1. Random Oblivious Transfer

The ROT protocol shown in Table 2 can be divided into three phases: parameter generation, making a choice, and key derivation. After the ROT protocol, the sender can obtain two indistinguishable and uniformly distributed keys, while the receiver only knows the chosen one.

Table 2. Random oblivious transfer protocol.

Alice		Bob
Parameter Generation:		
$seed_a \leftarrow (U(\{0,1\}^{256}))$		$a \leftarrow gen(seed_a)$
$a \leftarrow gen(seed_a)$		$s' \leftarrow \beta_\mu(R_q^{l \times 1})$
$s \leftarrow \beta_\mu(R_q^{l \times 1})$	$seed_{a,s}, T$	$b' = bits(a^T s' + h, \in_q, \in_p)$
$T \leftarrow U(R_q^{l \times 1})$		$v' = A^T bits(s', \in_p, \in_p) + h_1$
$A = bits(as + h, \in_q, \in_p) \leftarrow R_q^{l \times 1}$		$c = bits(v', \in_p - 1, \in_t)$
Make a choice:		
	B, c	$if(\sigma = 0) B = b'$
		$if(\sigma = 1) B = b' + T$
Key derivation:		
$v_1 = B^T bits(s, \in_p, \in_p) + h_1$		
$v_2 = (B - T)^T bits(s, \in_p, \in_p) + h_1$		
$sk_1 = bits(v_1 - 2^{\in_p - \in_t - 1} c + h_2, \in_p, 1)$		$sk = Hash(bits(v', \in_p, 1))$
$sk_2 = bits(v_2 - 2^{\in_p - \in_t - 1} c + h_2, \in_p, 1)$		

4.1.1. Correctness of the Protocol

The correctness of the protocol can be derived owing to the correctness of the Saber protocol. In other words, if two honest or semi-honest participants interact under the protocol, they can end up with a shared key sk_i and R knows nothing about $sk_{1-\sigma}$.

4.1.2. The Privacy of the Receiver

In the protocol, the R only sends (B, c) to the S , but the S can get (v_1, v_2, sk_1, sk_2) through (B, c) . Therefore, it is required that no information about σ can be obtained from (B, c) and (v_1, v_2, sk_1, sk_2) to ensure that the R 's privacy is not compromised. The following theorem is used to prove the privacy security of the R .

Theorem 1. Suppose no probabilistic polynomial time adversary can guess σ with non-negligible probability from (B, c) and (v_1, v_2, sk_1, sk_2) . In that case, it means that the R 's privacy is guaranteed during the execution. Then, it is necessary to prove the following three points:

1. $B_{\sigma=0}$ and $B_{\sigma=1}$ are indistinguishable.
2. c and σ are independent of each other.
3. v_1 and v_2 are indistinguishable.

Proof of Theorem 1.

1. Assuming that the adversary corrupts S , the corrupted sender is denoted as S^* .

It follows that $b' = bits(a^T s' + h, \in_q, \in_p)$ and $bits(u, \in_q, \in_p)$ are indistinguishable because of the Mod-LWR assumption, where $u \leftarrow \mathcal{U}(R_q^{l \times 1})$. Besides, the difference between $B_{\sigma=0}$ and $B_{\sigma=1}$ is T , where $T \leftarrow U(R_q^{l \times 1})$. Hence, $B_{\sigma=0} = b'$ and $B_{\sigma=1} = b' + T$ are indistinguishable.

2. As c sent to the sender is computed by the receiver and the sender can obtain σ from c . Assuming the S^* can get s' from c , then the S^* can distinguish between $B_{\sigma=0}$ and $B_{\sigma=1}$, and finally get σ . Therefore, it is necessary to prove that c and σ are independent of each other by showing that the possibility of obtaining σ from c is negligible.

The first step is to prove indistinguishability. It can be shown by the Mod-LWR problem that c is indistinguishable from uniformly distributed sampling; thus, σ cannot be obtained directly from c .

The second step is to prove independence. A series of games shown in Table 3 show that c cannot help the S^* to distinguish $B_{\sigma=0}$ from $B_{\sigma=1}$. Let X_i denote the event that the S^* guesses on n in the i -th game. Define the advantage of the adversary S^* to distinguish c from uniformly distributed sampling as follows:

$$ADV_{p,q,t}^{\text{Mod-LWR}}(S^*) = |Pr(S^*(a, A, T, B, c) = 1) - Pr(S^*(a, A, T, B, c') = 1)| \quad (5)$$

Table 3. The games mentioned in Proof of Theorem 1.

Game0:	Game1:	Game2:	$D_1(a, A)$:	$D_2((a, b'), (A, v'))$:
$seed_a \leftarrow \mathcal{U}(\{0, 1\}^{256})$ $a \leftarrow \text{gen}(seed_a)$ $s, s' \leftarrow \beta_\mu(R_q^{l \times 1})$ $T \leftarrow \mathcal{U}(R_q^{l \times 1})$ $A = \text{bits}(as + h, \in_q, \in_p)$ $b' = \text{bits}(a^T s' + h, \in_q, \in_p)$ $v' =$ $A^T \text{bits}(s', \in_p, \in_p) + h_1$ $c = \text{bits}(v', \in_p - 1, \in_t)$ $B = b' + \sigma T$ $c' \leftarrow \mathcal{U}(R_t)$ $n \leftarrow \mathcal{U}(\{0, 1\})$ If $(n = 0)$, then return (a, A, T, B, c) else return (a, A, T, B, c')	$seed_a \leftarrow \mathcal{U}(\{0, 1\}^{256})$ $a \leftarrow \text{gen}(seed_a)$ $s, s' \leftarrow \beta_\mu(R_q^{l \times 1})$ $T \leftarrow \mathcal{U}(R_q^{l \times 1})$ $b' = \text{bits}(a^T s' + h, \in_q, \in_p)$ $v' =$ $A^T \text{bits}(s', \in_p, \in_p) + h_1$ $c = \text{bits}(v', \in_p - 1, \in_t)$ $B = b' + \sigma T$ $c' \leftarrow \mathcal{U}(R_t)$ $n \leftarrow \mathcal{U}(\{0, 1\})$ If $(n = 0)$, then return (a, A, T, B, c) else return (a, A, T, B, c')	$seed_a \leftarrow \mathcal{U}(\{0, 1\}^{256})$ $a \leftarrow \text{gen}(seed_a)$ $s, s' \leftarrow \beta_\mu(R_q^{l \times 1})$ $T \leftarrow \mathcal{U}(R_q^{l \times 1})$ $A \leftarrow \mathcal{U}(R_q^{l \times 1})$ $c = \text{bits}(v', \in_p - 1, \in_t)$ $B = b' + \sigma T$ $c' \leftarrow \mathcal{U}(R_t)$ $n \leftarrow \mathcal{U}(\{0, 1\})$ If $(n = 0)$, then return (a, A, T, B, c) else return (a, A, T, B, c')	$s, s' \leftarrow \beta_\mu(R_q^{l \times 1})$ $T \leftarrow \mathcal{U}(R_q^{l \times 1})$ $b' = \text{bits}(a^T s' + h, \in_q, \in_p)$ $v' =$ $A^T \text{bits}(s', \in_p, \in_p) + h_1$ $c = \text{bits}(v', \in_p - 1, \in_t)$ $B = b' + \sigma T$ $c' \leftarrow \mathcal{U}(R_t)$ $n \leftarrow \mathcal{U}(\{0, 1\})$ If $(n = 0)$, then return (a, A, T, B, c) else return (a, A, T, B, c')	$c = \text{bits}(v', \in_p - 1, \in_t)$ $B = b' + \sigma T$ $c' =$ $\text{bits}(\mathcal{U}(R_p), \in_p - 1, \in_t)$ $n \leftarrow \mathcal{U}(\{0, 1\})$ If $(n = 0)$, then return (a, A, T, B, c) else return (a, A, T, B, c')

GAME 0: In this game, (a, s, A) , (a, s', b') , and (A, s', v') are Mod-LWR pairs, so

$$ADV(S^*) = |Pr(X_0) - 1/2| \quad (6)$$

GAME 1: In this game, suppose that the D_1 is a probabilistic polynomial-time algorithm with input (a, A) . If the D_1 can distinguish between Game 0 and Game 1, the S^* can use the D_1 to distinguish between $A = as$ and $\mathcal{U}(R_q^{l \times 1})$. Hence,

$$|Pr(X_0) - Pr(X_1)| \leq ADV_{p,q,t}^{\text{Mod-LWR}}(S_{D_1}^*) \quad (7)$$

GAME 2: In this game, suppose that the D_2 is a probabilistic polynomial-time algorithm with input $((a, b'), (A, v'))$. If the D_2 can distinguish between Game 1 and Game 2, then the S^* can use the D_2 to distinguish between $b' = a^T s'$ and $\mathcal{U}(R_q^{l \times 1})$, $v' = A^T s'$, and $\mathcal{U}(R_p)$. Hence,

$$|Pr(X_1) - Pr(X_2)| \leq ADV_{p,q,t}^{\text{Mod-LWR}}(S_{D_2}^*) \quad (8)$$

In Game 2, the adversary guesses the value of n by distinguishing between c and c' . However, c is obtained by scaling and rounding, which is uniformly distributed over R_p . Hence, c and c' are indistinguishable.

If the S^* can distinguish between Game 1 and Game 0 or Game 2 and Game 1, the adversary can obtain σ and s' . However, the adversary cannot distinguish them because of the Mod-LWR difficult problem. Above all, c and σ are independent of each other.

3. 1 and 3 are based on the same assumption, so contradiction can be used. If a PPT adversary can distinguish between v_1 and v_2 on input B , he can distinguish between B_0 and B_1 to obtain σ . Assuming the adversary can solve problem 3, the adversary

must be able to solve problem 1, which contradicts problem 1. Hence, v_1 and v_2 are indistinguishable. $\square$

4.1.3. The Privacy of the Sender

Assuming the receiver only can get the chosen message after the protocol and knows nothing about other messages, the protocol can guarantee the sender's privacy.

In the protocol, the key for encryption and decryption is generated by the hash function. If the receiver knows the sender's private key s , it is possible to obtain $sk_{1-\sigma}$ by computing $(B - (1 - \sigma)T)^T \text{bits}(s, \epsilon_p, \epsilon_p) + h_1$. However, getting s from A is a difficult problem. Alternatively, if the sender can find a *key* that satisfies $\text{Hash}(\text{key}) = \text{Hash}(sk_{1-\sigma})$ and $\text{key} \neq sk_{1-\sigma}$, the sender can get $m_{1-\sigma}$. However, the random oracle machine that runs the hash function is controlled by the simulator and has collision resistance, so this case can also be ignored.

Next, a series of games shown in Table 4 are used to prove that the receiver cannot get $sk_{1-\sigma}$, thus the protocol can guarantee the sender's privacy. Theorem 2 can prove the security of the sender.

Table 4. The games mentioned in Proof of Theorem 2.

Game 0:	Game 1:	D_3 :
$seed_a \leftarrow \mathcal{U}(\{0, 1\}^{256})$ $a \leftarrow \text{gen}(seed_a)$ $s, s' \leftarrow \beta_\mu(R_q^{l \times 1})$ $T \leftarrow \mathcal{U}(R_q^{l \times 1})$ $A = \text{bits}(as + h, \epsilon_q, \epsilon_p)$ $b' = \text{bits}(a^T s' + h, \epsilon_q, \epsilon_p)$ $v' = A^T \text{bits}(s', \epsilon_p, \epsilon_p) + h_1$ $c = \text{bits}(v', \epsilon_p - 1, \epsilon_t)$ $B = b' + \sigma T$ $v_{1-\sigma} = (B - \sigma T)^T \text{bits}(s, \epsilon_p, \epsilon_p) + h_1$ $sk = \text{bits}(v_{1-\sigma} - 2^{\epsilon_p - \epsilon_t - 1} c + h_2, \epsilon_p, 1)$ $sk' \leftarrow \mathcal{U}(R)$ <i>If</i> ($n = 0$), <i>then</i> $\text{return}(a, A, T, B, c, sk)$ <i>else</i> $\text{return}(a, A, T, B, c, sk')$	$seed_a \leftarrow \mathcal{U}(\{0, 1\}^{256})$ $a \leftarrow \text{gen}(seed_a)$ $s, s' \leftarrow \beta_\mu(R_q^{l \times 1})$ $T \leftarrow \mathcal{U}(R_q^{l \times 1})$ $A = \text{bits}(as + h, \epsilon_q, \epsilon_p)$ $c = \text{bits}(v', \epsilon_p - 1, \epsilon_t)$ $B = b' + \sigma T$ $v_{1-\sigma} = (B - \sigma T)^T \text{bits}(s, \epsilon_p, \epsilon_p) + h_1$ $sk = \text{bits}(v_{1-\sigma} - 2^{\epsilon_p - \epsilon_t - 1} c + h_2, \epsilon_p, 1)$ $sk' \leftarrow \mathcal{U}(R)$ <i>If</i> ($n = 0$), <i>then</i> $\text{return}(a, A, T, B, c, sk)$ <i>else</i> $\text{return}(a, A, T, B, c, sk')$	$c = \text{bits}(v', \epsilon_p - 1, \epsilon_t)$ $B = b' + \sigma T$ $v_{1-\sigma} = (B - \sigma T)^T \text{bits}(s, \epsilon_p, \epsilon_p) + h_1$ $sk = \text{bits}(v_{1-\sigma} - 2^{\epsilon_p - \epsilon_t - 1} c + h_2, \epsilon_p, 1)$ $sk' \leftarrow \mathcal{U}(R)$ <i>If</i> ($n = 0$), <i>then</i> $\text{return}(a, A, T, B, c, sk)$ <i>else</i> $\text{return}(a, A, T, B, c, sk')$

Theorem 2. Owing to the Mod-LWR difficulty problem, no PPT adversary R^* can obtain both v_1 and v_2 .

Proof of Theorem 2. The R^* determines the value of n at the end of each game. If the advantage of the R^* in judging the value of n is not negligible, the R^* wins the game.

In Game 0, v' , c , and uniformly distributed samples are indistinguishable, so

$$\text{ADV}(S^*) = |\text{Pr}(X_0) - 1/2| \quad (9)$$

Suppose that the D_3 is a probabilistic polynomial-time algorithm with input $((a, b'), (A, v'))$. When b' and v' are computed, the D_3 outputs (a, A, T, B, c, sk) ; however, when b' and v' are randomly sampled from a uniform distribution, the D_3 outputs (a, A, T, B, c, sk') . Owing to Mod-LWR difficulties, Game 0 and Game 1 are indistinguishable.

The next step is to prove that sk can be regarded as a uniformly distributed sampling on the ring. In Game 1, b' , v , and T are samples from a uniform distribution, so B is uniformly distributed on $R_q^{l \times 1}$ and c is uniformly distributed on R_t .

It is necessary to prove that sk and sk' are indistinguishable. Suppose there exists $\bar{b}'$, which satisfies $\bar{b}' = \text{bits}(a^T s' + h, \in_q, \in_p)$ and $\bar{b}' \approx B - \sigma T$, then, for the R^* , $(B - \sigma T)^T \text{bits}(s, \in_p, \in_p) + h_1$ and $\bar{b}'^T \text{bits}(s, \in_p, \in_p) + h_1$ are distinguishable. Besides, we know that c and σ are independent of each other from Theorem 1. In addition, $\text{bits}(v_{1-\sigma} - 2^{\epsilon_p - \epsilon_t - 1}(\text{bits}((A^T \text{bits}(s', \in_p, \in_p) + h_1), \in_p - 1, \epsilon_t)) + h_2, \in_p, 1)$ and $\text{bits}(v_{1-\sigma} - 2^{\epsilon_p - \epsilon_t - 1}(\text{bits}((A^T \mathcal{U}(R_p) + h_1), \in_p - 1, \epsilon_t)) + h_2, \in_p, 1)$ are distinguishable. In summary, sk and sk' are indistinguishable for R^* , and $\Pr(X_1) = 1/2$. Therefore, the protocol guarantees the sender's privacy. $\square$

4.2. Oblivious Transfer Protocol

A standard UC-secure OT protocol can be obtained by combining the random OT mentioned in Section 4.1 and the authentication symmetric encryption algorithm mentioned in Section 3.3. After receiving the key, the S encrypts two messages M_1 and M_2 with two keys C_1 and C_2 , respectively. Then, the S sends them to the R , and the R decrypts the chosen ciphertext.

Theorem 3 demonstrates that the OT protocol can be simulated as an ideal function under the random oracle machine model and can resist static adversary attacks.

Theorem 3. *If the following assumptions hold, then the OT protocol can be simulated as $\mathcal{F}_{OT}$.*

1. *The adversary only performs static attacks.*
2. *The random oracle machine simulates the hash function.*

Proof of Theorem 3. The static adversary attacks can be divided into four types as follows:

1. Neither the S nor the R is corrupted.
2. Only the S is corrupted.
3. Only the R is corrupted.
4. Both the R and the S are corrupted.

Let $\mathcal{A}$ denote the adversary interacting with the honest participant in the real environment. For any $\mathcal{A}$, a simulator $\mathcal{S}$ can be constructed in the ideal environment, which runs the ideal function $\mathcal{F}_{OT}$ to simulate the interaction with the honest party. It is impossible for the $\mathcal{Z}$ to distinguish between the real execution and the ideal environment. All corruption cases are shown as follows.

1. When neither the S nor the R is corrupted, the adversary can attack the procession of data transmission. To demonstrate the universally composable security of the protocol, we build a simulator $\mathcal{S}$. The simulation for the ideal environment is built as follows:
 - The $\mathcal{S}$ simulates the S for parameter selection. The $\mathcal{S}$ chooses $seed_a \leftarrow \mathcal{U}(\{0, 1\}^{256})$, $s \leftarrow \beta_\mu(R_q^{l \times 1})$, and $T \leftarrow \mathcal{U}(R_q^{l \times 1})$; generates the key $a \leftarrow \text{gen}(seed_a) \in R_q^{l \times 1}$; and then computes $A = \text{bits}(as + h, \in_q, \in_p)$.
 - The $\mathcal{S}$ simulates the R to make a choice. The $\mathcal{S}$ chooses $s \leftarrow \beta_\mu(R_q^{l \times 1})$ and calculates $b' = \text{bits}(a^T s' + h, \in_q, \in_p)$, $v' = A^T \text{bits}(s', \in_p, \in_p) + h_1$, and $c = \text{bits}(v', \in_p - 1, \epsilon_t)$.
 - The $\mathcal{S}$ generates (C_1, C_2) by random sampling from the ciphertext space.
 - The $\mathcal{S}$ outputs $(seed_a, A, T)$, (B, c) , and (C_1, C_2) .

From Theorems 1 and 2, we know that the protocol can guarantee the privacy of the sender S and the receiver R , so the $\mathcal{Z}$ cannot distinguish whether $(seed_a, A, T)$ and (B, c) are in the real execution or ideal environment. The $\mathcal{Z}$ can also not distinguish between (C_1, C_2) in the real execution and the ideal environment owing to the authentication symmetric encryption algorithm. Hence, the ideal environment and the real execution are indistinguishable from the perspective of the $\mathcal{Z}$.

2. When the adversary $\mathcal{A}$ corrupts the S , denote S as S^* . We build a simulator $\mathcal{S}$ to prove that the ideal environment and the real execution are indistinguishable. First, the $\mathcal{S}$ interacts with the S^* to get the real input (M_1, M_2) , then plays the role of the S to interact with the ideal function $\mathcal{F}_{OT}$, which finally enables the R to get M_σ . As the $\mathcal{S}$ responds to S^* 's queries using a random oracle machine, it is possible to decrypt the ciphertext using query records and get the sender's real input. The $\mathcal{S}$ is constructed as follows:
 - The $\mathcal{S}$ uses the random oracle machine to answer queries randomly before interacting with the S^* .
 - After receiving $(seed_a, T, A)$ from the S^* , the $\mathcal{S}$ generates the parameters by seed; chooses $s' \leftarrow \beta_\mu(R_q^{l \times 1})$; calculates $b' = bits(a^T s' + h, \in_q, \in_p)$, $v' = A^T bits(s', \in_p, \in_p) + h_1$, and $c = bits(v', \in_p - 1, \in_t)$; and sends (b', c) to the S^* . From this moment on, the $\mathcal{S}$ records all queries of the form $Hash_{((A,T),b',c)}$ from the S^* .
 - After receiving (C_1, C_2) from the S^* , the $\mathcal{S}$ computes $Dec(C_i, Hash_{record})$ for each i , where $Hash_{record}$ denotes all the queries records of the S^* , and records the M that can be decrypted successfully from C .
 - The $\mathcal{S}$ sends (M_1, M_2) to $\mathcal{F}_{OT}$.

The next step is to demonstrate the indistinguishability under the $\mathcal{Z}$'s view. The $\mathcal{Z}$ only can use (B, c) and M_σ to distinguish the real execution and ideal environment.

From Theorem 1, the protocol can guarantee the receiver's privacy so that the (B, c) generated by the $\mathcal{S}$ and the (B', c') caused by the honest R in the protocol are indistinguishable.

In addition, if the adversary can find an x that satisfies $Hash(x) \neq sk_\sigma$ and $Dec(C_\sigma, Hash(x)) = M$, the x will help R output a different M . However, owing to the robustness of the authentication encryption algorithm, the probability is negligible.

3. When only the adversary $\mathcal{A}$ corrupts the R , denote R as R^* . We build a simulator $\mathcal{S}$ to prove the indistinguishability. The $\mathcal{S}$ needs to play the S 's role to interact with the R^* to obtain the σ , then uses σ to interact with $\mathcal{F}_{OT}$ to obtain M_σ to make the R decrypt correctly, and finally generates (C_1, C_2) based on M_σ . The $\mathcal{S}$ can obtain the value of σ by the queries records because the $\mathcal{S}$ runs the hash function and the protocol guarantees the sender's privacy. The $\mathcal{S}$ is constructed as follows:
 - The $\mathcal{S}$ chooses $seed_a \leftarrow \mathcal{U}(\{0, 1\}^{256})$, $s \leftarrow \beta_\mu(R_q^{l \times 1})$, and $T \leftarrow \mathcal{U}(R_q^{l \times 1})$; then generates key $a \leftarrow gen(seed_a)$; and finally computes $A = bits(as + h, \in_q, \in_p)$.
 - The $\mathcal{S}$ uses the random oracle machine to answer the query and records queries of the form $Hash_{((A,T),...)}$ before interacting with R^* .
 - The $\mathcal{S}$ looks for records of the form $Hash_{((A,T),B,c)}$ and judges if the output of the function is equal to sk after receiving (B, c) from the R^* . If equal, the $\mathcal{S}$ records the value of i as σ and interacts with the function $\mathcal{F}_{OT}$ to obtain M_σ ; otherwise, the $\mathcal{S}$ fails and sets the value of σ to null.
 - The $\mathcal{S}$ starts to generate (C_1, C_2) after interacting with the R^* . If the value of σ is null, the $\mathcal{S}$ randomly samples (sk_1, sk_2) in the ciphertext space and uses it as (C_1, C_2) ; otherwise, the $\mathcal{S}$ sets the value of sk_σ to the query result and computes $C_\sigma = Enc(sk_\sigma, M_\sigma)$. Finally, the $\mathcal{S}$ randomly generates $C_{1-\sigma}$ and sends (C_1, C_2) to the R^* .
 - The $\mathcal{S}$ continues to answer queries at random after outputting (C_1, C_2) . If the R^* continues with an inquiry of the form $KDF_{((A,T),B,c)}$, the answer is as follows. If the value of σ is not null, the $\mathcal{S}$ outputs that the protocol has ended. Otherwise, the $\mathcal{S}$ gets the value of σ like the above method. Finally, the $\mathcal{S}$ encrypts the M_σ using the generated key and sends (C_1, C_2) to the R^* .

The next step is to prove that the $\mathcal{Z}$ cannot distinguish between the ideal environments and the real execution. It is shown that the $\mathcal{Z}$ only can use $(seed, A, T)$ and (C_1, C_2) to distinguish between the real execution and the simulation.

First, $seed$, A , T are sampled from the distribution. According to Theorem 2, the protocol can guarantee the sender's privacy, so the (C_1, C_2) generated during the real execution of the protocol is indistinguishable from the random string. From the construction of the simulator, $C_{1-\sigma}$ in the simulation is an arbitrary string and C_σ is the real ciphertext, which is indistinguishable from a random string. Finally, if the protocol is stopped midway, the R^* cannot get (C_1, C_2) , and additional information can be obtained only by asking the random oracle machine. However, getting the value of c from v is impossible because of the Mod-LWR difficult problem. In summary, the $\mathcal{Z}$ cannot distinguish between the real execution and ideal environment.

4. When both the S and the R are corrupted, the simulator directly copies the adversary's input as its input and outputs the adversary's output to achieve indistinguishability in $\mathcal{Z}$ view. $\square$

5. UC-Secure 1-out-of-N OT Protocol

In this section, we expand the 1-out-of-2 OT protocol to a 1-out-of-N protocol with a minor tweak. The 1-out-of-N OT protocol is shown in the Table 5. In the 1-out-of-N OT ideal function, the S inputs $(M_1, \dots, M_n)$, where $n > 2$, then the R inputs choice bit $\sigma \in [1, n]$. Finally, the R outputs M_σ .

Table 5. UC-secure 1-out-of-N OT protocol.

Alice		Bob
Parameter Generation:		
$seed_a \leftarrow \mathcal{U}(\{0, 1\}^{256})$		$a \leftarrow gen(seed_a)$
$a \leftarrow gen(seed_a)$		$s' \leftarrow \beta_\mu(R_q^{l \times 1})$
$s \leftarrow \beta_\mu(R_q^{l \times 1})$	$seed_a, A, T$	$b' = bits(a^T s' + h, \in_q, \in_p)$
$T_0 = \{0\}^l$		$v' = A^T bits(s', \in_p, \in_p) + h_1$
$T_i \leftarrow \mathcal{U}(R_q^{l \times 1}), i \in [1, n-1]$		$c = bits(v', \in_p - 1, \in_t)$
$A = bits(as + h, \in_q, \in_p)$		
Make a choice:	$\xleftarrow{B, c}$	$B = b' + T_{\sigma-1}, \sigma \in [1, n]$
Key derivation:		
For every $i \in [1, n]$:		
$v_i = (B - T_i)^T bits(s, \in_p, \in_p) + h_1$		$sk = Hash(bits(v', \in_p, 1))$
$sk_i = bits(v_i - 2^{\in_p - \in_t - 1} c + h_2, \in_p, 1)$		
Encryption:	$\xrightarrow{C_1, C_2}$	Decryption:
$C_i = Enc(Hash(sk_i), M_i), i \in [1, n]$		$M_\sigma = Dec(C_\sigma, sk)$

Proof of Security

1. The first step is to prove that the protocol can guarantee the parties' privacy. From Theorem 1, it is clear that $B = b'$ and $B = b' + T_{\sigma-1}$ are indistinguishable, so the receiver's privacy is secure. From Theorem 2, the receiver only gets the information he chooses, so the sender's privacy is secure.
2. The next step is to prove that the protocol can be simulated as $\mathcal{F}_{OT}$. When the S is corrupted, the S records the random oracle machine queries in the specific format. Then, the S uses these records to obtain the real input from the sender and inputs it into the ideal function. When the R is corrupted, the S receives the real input σ from the receiver through the records of the random oracle machine. Then, the S inputs it to the ideal function. Finally, in the same way as Theorem 3, the real execution and the ideal environment from the perspective of $\mathcal{Z}$ are indistinguishable. Hence, the OT protocol can be simulated as $\mathcal{F}_{OT}$.

6. Discussion

We modified the codes on SABER's GitHub to simulate the protocol. The CPU is i7-9700f and the memory size is 16 GB. The simulation was repeated 100,000 times and the probability of success was 100%. The average communication cost is 24,576 bits and the average running time is 0.54509 ms. Therefore, the protocol can run efficiently with low communication costs.

Considering that none of the currently available lattice UC-safe OT protocols are based on the Mod-LWR difficulty problem, this protocol is compared to other OT protocols based on the other lattice difficulty problems such as LWE, RLWE, and so on.

The metrics used to judge the efficiency of UC-secure OT protocols are the number of rounds, communication cost, and computation cost. For the lattice OT protocols, the communication cost can be calculated from the size of intermediate amount and the computation cost can be calculated by operations on the lattice, such as multiplication and addition. Comparing the OT protocol proposed with other state-of-the-art protocols in the theoretical aspect, the results are shown in Table 6.

Table 6. Comparison with other OT protocols.

Scheme	Assumption	Model	Security	Strategy	Rounding	Communication Cost	Computational Cost
Peikert [16]	LWE	CRS	UC	Static	At least 2	$O(n^2 \lg n)(Z_q)$	$O(n^3 \lg n)(\times)$ $O(n^3 \lg n)(+)$
Liu [23]	RLWE	ROM	UC	Static	3	$O(n^2)(Z_q)$ $O(n^2)(Z_2)$	$O(n^3)(\times)$ $O(n^3 \lg n)(+)$
Quach [24]	LWE	CRS	UC	Static	2	$O(n^2 \lg n)(Z_q)$	$O(n^3 \lg n)(\times)$ $O(n^3 \lg n)(+)$
Our	Mod-LWR	ROM	UC	Static	3	$O(n)(R_p)$ $O(n^2)(R_t)$	$O(n^3)(\times)$ $O(n^3 \lg n)(+)$

The OT protocol proposed by Peikert is under the common reference string (CRS) model, which needs to interact with the ideal function to obtain the CRS before starting to run. The OT protocol proposed by Quach is also under the CRS model, which uses dual-mode encryption to achieve statistical receiver security. The OT protocol proposed by Liu needs $\text{rec}(\cdot)$ and $\text{dbl}(\cdot)$ operations, which leads to additional computational costs. Compared with the OT protocol proposed in [16,24], the protocol has advantages for communication and computational costs. Compared with the OT proposed in [22], the protocol reduces the communication cost by reducing the scale and size of the polynomial. In summary, the protocol in this paper has the advantage of high efficiency and low communication.

7. Conclusions

This paper constructs a UC-secure OT protocol on lattice based on the Mod-LWR difficult problem. The protocol obtains universally composable security under the random oracle machine model and can resist static adversary attacks. Besides, the comparison with other UC-secure OT protocols shows that the protocol has constant interaction rounds and low communication costs.

Depending on the attack strategy of the adversary, it can be classified as a static adversary, dynamic adversary, or proactive adversary. Therefore, the following research focuses on an OT protocol that can resist other types of attacks.

Author Contributions: Conceptualization, J.S.; formal analysis and investigation, J.S., D.W., Z.Z., H.D. and Z.L. (Zichen Li); writing—original draft preparation, J.S.; writing—review and editing, D.W.; visualization, J.S.; supervision, D.W.; Z.Z., Z.L. (Zhenzhen Li), H.D. and Z.L. (Zichen Li). All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported in part by the National Natural Science Foundation of China (61370188), in part by the Beijing Municipal Education Commission Scientific Research Project (KM202010015009), in part by the Beijing Municipal Education Commission Scientific Research Project Funding (KM202110015004), in part by the Beijing Institute of Graphic Communication Doctoral Funding Project (27170120003/020), in part by the Beijing Institute of Graphic Communication Research Innovation Team Project (Eb202101), in part by the Intramural Discipline Construction Project of Beijing Institute of Graphic Communication (21090121021), in part by the Key Educational Reform Project of Beijing Institute of Graphic Communication under (22150121033/009), in part by the BIGC (Ec202201), in part by the Beijing Institute of Graphic Communication Doctoral Funding Project (27170122006), and in part by the general research project of Beijing Association of Higher Education (MS2022093).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data available on request from the authors.

Conflicts of Interest: The authors declare no conflict of interest.

References

- Huang, Z.; Lu, W.; Hong, C.; Ding, J. Cheetah: Lean and fast secure two-party deep neural network inference. *Cryptol. ePrint Arch.* **2022**, *207*, 1–20.
- Yang, J.; Wang, T.; Li, N.; Cheng, X.; Su, S. Answering Multi-Dimensional Range Queries under Local Differential Privacy. *arXiv* **2020**, arXiv:2009.06538. [CrossRef]
- Hong, C.; Katz, J.; Kolesnikov, V.; Lu, W.-J.; Wang, X. Covert Security with Public Verifiability: Faster, Leaner, and Simpler. In Proceedings of the 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, 19–23 May 2019; pp. 97–121. [CrossRef]
- Pan, J.-S.; Liu, T.; Yan, B.; Yang, H.-M.; Chu, S.-C. Using color QR codes for QR code secret sharing. *Multimedia Tools Appl.* **2022**, *81*, 15545–15563. [CrossRef]
- Wang, X.; Luo, T.; Li, J. An Efficient Fully Homomorphic Encryption Scheme for Private Information Retrieval in the Cloud. *Int. J. Pattern Recognit. Artif. Intell.* **2019**, *34*, 2055008. [CrossRef]
- Rackoff, C.; Simon, D.R. Non-Interactive Zero-Knowledge Proof of Knowledge and Chosen Ciphertext Attack. In *Advances in Cryptology—CRYPTO ’91*; Springer: Berlin/Heidelberg, Germany, 1992; pp. 433–444. [CrossRef]
- Chailloux, A.; Kerenidis, I. Optimal Bounds for Quantum Bit Commitment. In Proceedings of the 2011 IEEE 52nd Annual Symposium on Foundations of Computer Science, Washington, DC, USA, 22–25 October 2011; pp. 354–362. [CrossRef]
- Pinkas, B.; Schneider, T.; Zohner, M.; Segev, G. Phasing: Private Set Intersection using Permutation-based Hashing. In Proceedings of the 24th USENIX Security Symposium, Austin, TX, USA, 10–12 August 2012; p. 17.
- Weiguo, Z.; Man, S.U.N.; Zhenhua, C.; Wei, C. Secure Multi-party Computation of Spatial Relationship and Its Application. *JEIT* **2016**, *38*, 2294–2300. [CrossRef]
- Rabin, M.O. How to Exchange Secrets with Oblivious Transfer. *IACR Cryptol. ePrint Arch.* **2005**, 187. Available online: <https://www.semanticscholar.org/paper/How-To-Exchange-Secrets-with-Oblivious-Transfer-Rabin/772cdcc8a67cc878b39409230cbf2488a1117e62> (accessed on 15 November 2022).
- Even, S.; Goldreich, O.; Lempel, A. A Randomized Protocol for Signing Contracts. *Commun. ACM* **1983**, *28*, 637–647. [CrossRef]
- Brassard, G.; Crepeau, C.; Robert, J.-M. All-or-nothing disclosure of secrets. In *Advances in Cryptology—CRYPTO ’86*; Springer: Berlin/Heidelberg, Germany, 1987; pp. 234–238.
- Tzeng, W.-G. Efficient 1-Out-n Oblivious Transfer Schemes. In *Public Key Cryptography*; Springer: Berlin/Heidelberg, Germany, 2002; pp. 159–171. [CrossRef]
- Naor, M.; Pinkas, B. Efficient Oblivious Transfer Protocols. 2001. Available online: <https://xueshu.baidu.com/usercenter/paper/show?paperid=be727901097ac71cc01239a43ca4e160> (accessed on 5 August 2022).
- Damgård, I.; Nielsen, J.B.; Orlandi, C. Essentially Optimal Universally Composable Oblivious Transfer. In *Information Security and Cryptology—ICISC 2008*; Springer: Berlin/Heidelberg, Germany, 2009; pp. 318–335.
- Lindell, A.Y. Efficient fully-simulatable oblivious transfer. In *Topics in Cryptology—CT-RSA 2008*; Springer: Berlin/Heidelberg, Germany, 2008; pp. 52–70.
- Canetti, R. Universally Composable Security: A New Paradigm for Cryptographic Protocols. *Cryptology ePrint Archive*. 2000. Available online: <https://eprint.iacr.org/2000/067> (accessed on 5 August 2022).
- Peikert, C.; Vaikuntanathan, V.; Waters, B. A Framework for Efficient and Composable Oblivious Transfer. In *Advances in Cryptology—CRYPTO 2008*; Springer: Berlin/Heidelberg, Germany, 2008; pp. 554–571. [CrossRef]
- Lai, Y.-F.; Galbraith, S.D.; de Saint Guilhem, C.D. Compact, efficient and UC-Secure isogeny-based oblivious transfer. In *Advances in Cryptology—EUROCRYPT 2021*; Springer: Cham, Switzerland, 2021; pp. 213–241.

20. Branco, P.; Ding, J.; Goulão, M.; Mateus, P. A framework for universally composable oblivious transfer from one-round key-exchange. In *Cryptography and Coding*; Springer: Cham, Switzerland, 2019; pp. 78–101.
21. Chou, T.; Orlandi, C. The Simplest Protocol for Oblivious Transfer. In *Progress in Cryptology—LATINCRYPT 2015*; Lauter, K., Rodríguez-Henríquez, F., Eds.; Springer: Cham, Switzerland, 2015; Volume 9230, pp. 40–58. [\[CrossRef\]](#)
22. Liu, M.; Hu, Y. Universally composable oblivious transfer from ideal lattice. *Front. Comput. Sci.* **2018**, *13*, 879–906. [\[CrossRef\]](#)
23. Xing, Y.; Li, S. An Efficient Implementation of the NewHope Key Exchange on FPGAs. *IEEE Trans. Circuits Syst. I: Regul. Pap.* **2020**, *67*, 866–878. [\[CrossRef\]](#)
24. Quach, W. UC-Secure OT from LWE, Revisited. In Proceedings of the SCN 2020: Security and Cryptography for Networks, Amalfi, Italy, 14–16 September 2020; pp. 192–211. [\[CrossRef\]](#)
25. Couteau, G.; Rindal, P.; Raghuraman, S. Silver: Silent VOLE and Oblivious Transfer from Hardness of Decoding Structured LDPC Codes. In *Advances in Cryptology—CRYPTO 2021*; Springer: Berlin/Heidelberg, Germany, 2021; pp. 502–534. [\[CrossRef\]](#)
26. D’Anvers, J.-P.; Karmakar, A.; Roy, S.S.; Vercauteren, F. Saber: Modulo-LWR based key exchange, CPA-Secure encryption and CCA-Secure KEM. In *Progress in Cryptology—AFRICACRYPT 2018*; Springer: Cham, Switzerland, 2018; pp. 282–305.
27. Wu, H.; Preneel, B. AEGIS: A fast authenticated encryption algorithm. In *Selected Areas in Cryptography—SAC 2013*; Springer: Berlin/Heidelberg, Germany, 2014; pp. 185–201.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.