Luo *et al. Journal of Cloud Computing* (2024) 13:22
https://doi.org/10.1186/s13677-023-00586-5

Journal of Cloud Computing:
Advances, Systems and Applications

RESEARCH

Open Access

Cloud-SMPC: two-round multilinear maps secure multiparty computation based on LWE assumption

Yun Luo[1], Yuling Chen[1*], Tao Li[1], Chaoyue Tan[2] and Hui Dou[1]

Abstract

Cloud computing has data leakage from all parties, security protection of private data, and existing solutions do not provide a trade-off between security and overhead. With distributed data communication due to data barriers, information interaction security and data computation security have become challenges for secure computing. Combining cloud computing with secure multiparty computation can provide a higher level of data protection while maintaining the benefits of cloud computing. In this case, data can be stored in the cloud and computed through SMPC protocols, thus protecting the privacy and security of the data. However, multiple rounds of information interaction are often required, increasing the communication overhead, and the security strength is limited by the hardness assumption. In this paper, we work to achieve an optimal setting of the number of rounds in secure multi-party computation on the cloud to achieve a sublinear communication overhead and improve the security concept. A 2-round SMPC protocol is constructed in the framework of Universally Composable (UC). A 2-round SMPC protocol is constructed that uses multilinear maps based on the Learning from Errors (LWE) assumption. The participant encodes the input and sends it via broadcast to reduce the interaction, homomorphic computational encoding information for secure access to computational data and secure the SMPC protocol through UC security. This paper extends the participants to multiple parties, reduces the communication rounds to 2, the protocol achieves sublinear communication overhead in *poly* polynomial time, smaller setup size to *poly*(k), and static security is achieved.

Keywords Secure multiparty computation, Cloud computing, UC framework, LWE assumption, Round complexity

Introduction

Cloud computing has grown considerably in recent years, and the development of computing models that store data and applications on remote servers has matured and become popular. While edge cloud computing [1–4] enables efficient data processing and transmission, and many applications are also available in the Internet of Things [5, 6]. Combining deep learning and model training with cloud computing for web personalized recommendation system and anomaly detection [7–10].But it also poses some security risks [11]. As data and computation are dispersed to edge nodes, attackers may exploit weaker security mechanisms to compromise these nodes, leading to problems such as data leakage or service disruption. Therefore, strict security measures, including data encryption, authentication, access control, and vulnerability management, must be adopted in edge cloud computing applications to ensure the security and stability of the system. In the background of big data, data security, communication security [12, 13] and secure data sharing, privacy computing have become particularly important.Many researchers have conducted

*Correspondence:
Yuling Chen
ylchen3@gzu.edu.cn
[1] State Key Laboratory of Public Big Data, College of Computer Science and Technology, Guizhou University, Guiyang, China
[2] State Key Laboratory of Public Big Data, Guizhou University, Guiyang, China

in-depth research and studies in many areas such as data security, privacy protection, and adversary attack and defense [14–18]. From the perspective of cryptography, secure multiparty computation [19] technology provides a reasonable solution. Yao proposed the "millionaire problem" in 1982, leading to the first secure two-party computation protocol [20], which uses the technique of circuits to represent computational functions as boolean circuits and provides computational security for secure two-party computation protocols under a semi-honest model. This was followed by Goldreich et al. who gave the first secure multiparty computation protocol [21] and guaranteed the security of the protocol under a semi-honest model. After decades of development, existing research has focused on the performance of SMPC, mainly on the number of communication rounds, communication complexity, computational complexity, and minimization of complexity assumptions to enhance the concept of security. Abraham et al. [22] construct a protocol based on verifiable secret sharing (VSS) that matches a semi-honest setting with a round complexity that is proportional to the circuit depth. A SMPC protocol against malicious adversaries and without trustworthy assumption settings was proposed [23], setting up a 5-round SMPC protocol based on Decisional Diffie-Hellman (DDH) assumption and a 4-round SMPC protocol constructed by one-way permutations based on sub-exponential security DDH assumption. For the problem of optimizing the number of rounds in a protocol, Ananth et al. [24] study the round complexity of n-party protocols in an honest majority setting to tolerate the corruption of $t < \frac{n}{2}$ participants and achieve abort security under the plain model where the security of the protocol depends only on the one-way function. For SMPC, cryptographic techniques such as Laconic Function Evaluation (LFE), oblivious transfer (OT) are used to construct secure multiparty computation protocols [25–27] and to reduce the number of rounds of interaction between participants. Existing studies have shown that high communication overhead, high complexity of rounds and low security strength in secure multiparty computation. Therefore, this paper is based on cloud computing to achieve a more secure and efficient secure multiparty computation scheme, the general process is as follows Fig. 1.

This paper is devoted to solving the round number complexity optimization problem in SMPC on the cloud by introducing harder security assumptions to improve the security concept, reduce the number of interactions, and achieve low communication overhead for privacy-secure cloud computing. The contributions of this paper are as follows.

1. To optimize the round complexity of the protocol in cloud-based SMPC, we construct a 2-round secure multiparty computation using a multilinear map based on the LWE assumption.
2. In this paper the implementation of the protocol is done in the UC framework, the ideal functionality is delivered to the computing on the cloud and each participant can access the ideal functionality, the protocol is finally implemented on the cloud with UC security for SMPC security and increased security strength.

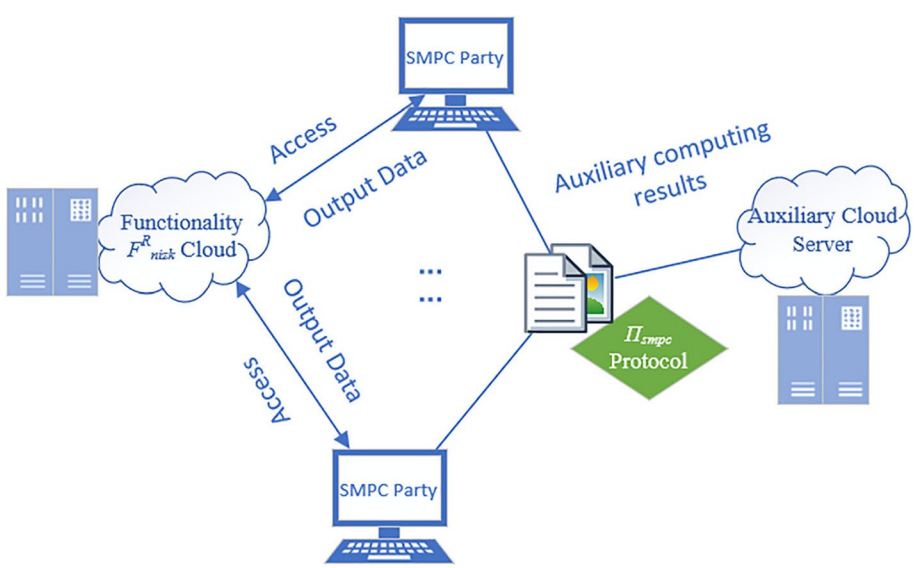

Fig. 1 Secure multiparty computation scheme on the cloud

3. The parameters of the SMPC protocol settings are only related to the LWE instantiation and the depth of the computational circuit, which achieving sublinear overhead for communication.

Related work section describes the Related Work in the area of secure multiparty computation, and Preliminaries section provides an overview of multilinear maps, learning with errors (LWE), universally composable (UC), garbled circuit, and zero-knowledge proof. Protocol construction section describes the specific construction of the scheme in this paper, and Security demonstration section is a security demonstration of the scheme construction.Finally, we summarize our work in Conclusions section.

Related work

Secure multiparty computation with constant rounds was first studied in [28] to reduce the number of interaction rounds, Gordon et al. [29] designed SMPC protocols with constant rounds in honest majority to ensure that the parties have fairness as well as the output is delivered correctly (Table 1). A series of subsequent works target the security of SMPC, based on various difficult assumptions to design protocols for information security against malicious adversaries [30–32]. Garbled circuits combined with non-committing encryption (NCE) under the plain model to construct secure multiparty computation protocols with adaptive constant rounds are also described for some extensions and applications [33]. By combining cryptographic primitives, based on learning with errors (LWE) assumptions use fully homomorphic encryption [34] to construct two rounds of secure multiparty computation, allowing one round of distributed decryption of ciphertexts with multiple secret keys [35, 36], which gave a great impetus to later research. The bilinear mapping operation provides a unique operation for secure multiparty computation that would encode the input and then perform the operation, and the evaluation process takes as input the set of confusing protocol components with labels corresponding to the input encoding of each party, and outputs the entire text of the distributed protocol [37], which in turn incorporates the garbled circuit to design the garbled protocol. With the study and development of lattice trapdoors [38–40], constructing encoding schemes based on trapdoors and improving the security level of the schemes under the LWE assumption, hierarchical multilinear encoding has been widely used in cryptography, from non-interactive key exchange protocols to broadcast and attribute-based encryption. ciampi et al [41] for the construction of secure two-party computation using oblivious transfer protocols, the construct such protocols from permutations of trapdoors based on four rounds of non-extensible zero-knowledge arguments for delayed inputs. The development of UC [42, 43] likewise has many applications in the field of secure multiparty computation. In the framework of UC, the security of the protocol relies on the security of UC to achieve indistinguishability between ideal and realistic environments. In the concept of static security, protocols for sublinear communication are constructed using threshold FHE as well as zero-knowledge proofs (NIZK) [44], which typically require four rounds of interaction under the threshold PKI model and five rounds under the CRS model. The optimization of the number of rounds is carried out in the honest majority setting, and the protocol design is carried out in the model where the circuit size is the polynomial communication size [24, 29], thus achieving static security. Existing studies have shown that high communication overhead, high complexity of rounds and low security strength in secure multiparty computation. The solution in this paper is dedicated to the optimization of the number of rounds and the communication overhead, introducing harder security assumptions and improving the notion of security.

Table 1 The computational function $F:(0,1^{l_{in}})^n \rightarrow 0,1^{l_{out}}$ is compared to the SMPC protocol represented by a circuit C of depth d in the honest majority setting

Protocol	Security	Rounds	Communication Complexity
[44]	Static	4	$poly(l_{in}, l_{out}, k, d, n)$
[24]	Static	3	$\|C\| \cdot poly(k, n)$
[29]	Static	3	$poly(l_{in}, l_{out}, k, d, n)$
[22]	Static	$o(d)$	$\|C\| \cdot poly(k, n)$
[46]	adaptive	4	$poly(l_{in}, l_{out}, k, d, n)$
[42]	adaptive	$O(1)$	$\|C\| \cdot poly(k, n)$
This work	Static	2	$poly(l_{in}, l_{out}, k, d, n)$

Preliminaries

In this section we will review multilinear maps, learning with errors (LWE), universally composable(UC), garbled circuit, and zero-knowledge proof. We denote $k \in N$ as the security parameter and for all $n \in N$,[n] is denoted as $\{1, 2, ..., n\}$. PPT denotes probabilistic polynomial time and $poly$ denotes positive polynomial. For a function μ with $\mu(k) < \frac{1}{poly(k)}$, the function μ is said to be negligible. let $x = (x_1, x_2, ..., x_n)$ be a vector, the norm of a vector x is defined as $\|x\|_\infty = max_i(x[i])$. If the two distributions D_1,D_2 are statistically close, we write them as $D_1 \overset{s}{\equiv} D_2$. If the two distributions D_1,D_2 are computational indistinguishability, we write them as $D_1 \overset{c}{\equiv} D_2$.

Multilinear maps

Multilinear maps [39, 40] is a mathematical tool that is abstractly defined and allows us to operate in a series of group elements and to extract a part of the information out of the output in combination with the results. Given t cyclic groups $G_1, G_2, ..., G_t$ and a target cyclic group G, then for a multilinear maps algorithm e of order t we have.

$$e : G \leftarrow G_1 \times G_2 \times ... \times G_t. \tag{1}$$

$$e({g_1}^{x_1}, {g_2}^{x_2}, ..., {g_t}^{x_t}) = g^{x_1 \times x_2 \times ... \times x_t}. \tag{2}$$

where $g_1, g_2, ..., g_t$ distributions represent t cyclic groups $G_1, G_2, ..., G_t$ generators, g is denoted as the generators of the target group G, $x_1, x_2, ..., x_t \in \{0, 1\}^*$, we can consider ${g_i}^{x_i} (i \in [t])$ as the encoding of x_i, and the t-order multilinear maps algorithm can encode the t unknown characters $x_1, x_2, ..., x_t$ string encoding ${g_1}^{x_1}, {g_2}^{x_2}, ..., {g_t}^{x_t}$ on which $x_1, x_2, ..., x_t$ on the group G with the joint product encoding $g^{\prod x_i, i \in [t]}$. Similarly we can get the corresponding additive operations, if $\boxplus$ denotes the operations defined in the group, we have $g^{x_1} \boxplus g^{x_2}) = g^{x_1 + x_2}$. However, for multilinear maps of order t, we can only perform at most multiplication of t layers and addition of any layer. For multilinear maps, the result after computation is presented in the form of ciphertext, and we can extract a part of the information of the ciphertext using the zero test algorithm. ZeroTest algorithm:Given an element h, the ZeroTest algorithm verifies that h is an element of the target group G.

$$ZeroTest(h) : h \in G \; or \; h \notin G \; by \; h \stackrel{?}{=} g^0 . \tag{3}$$

If $h = g^0$ then there is h is an element in the target group G. If $h \neq g^0$ then h is not an element in the target group G.

Learning with errors

The trapdoor-based LWE design has also been developed through the study of lattice trapdoors [38], where K denotes the security parameter and the parameters $n = n(k), q = q(k)$ of the LWE [45] instance are chosen to be integers, $\chi = \chi(k)$ is a distribution over Z, and $LWE_{n,q,m}$ assumes that for all polynomials $m = m(k)$ there is the following distribution that is indistinguishable.

$$(A, sA + e) \stackrel{c}{\equiv} (A, z). \tag{4}$$

where $A \leftarrow Z_q^{n \times m}$, $s \leftarrow Z_q^n$ is the input vector, $e \leftarrow \chi^m$ denotes the noise vector and $z \leftarrow Z_q^m$. In the LWE scheme with trapdoor [17, 19], for any $m' \in N$, A is represented as a uniform random distribution matrix with trapdoor $R \in Z_q^{m' \times n \log q}$ and constructing the LWE hard problem based on this matrix, another matrix D_1 can be generated by the matrix with trapdoor A such that $AD_1 = sA_1 + e_1$,Similarly$A_1 D_2 = sA_2 + e_2$, where the matrix A_1 is also a uniform random distribution matrix with a trap R_1, that is, we can generate the D_i matrix of the current level based on the trapdoor R_{i-1} of the previous level, and the whole process forms a nested chain structure.

Theorem 1 (secure MPC with sublinear communication [26, 46], informal). *Assuming LWE and secure erasures (alternatively, sub-exponential iO), every function can be securely computed by a 2-round protocol tolerating a malicious adversary that can adaptively corrupt all of the parties, such that the communication complexity, the online-computation complexity, and the size of the common reference string are sublinear in the function size.*

Universally composable

In [46–48] the universally composable framework is defined as the following two models and indistinguishable security properties are formed in the two models, resulting in UC security as well as compositional security.

Real Model: The whole execution process consists of a UC environment Z, an adversary A, and n participants, which starts with Z invoking all participants, generating all inputs and being able to read all outputs, and ends with Z outputting the result of the whole execution. The output of the environment Z under the realistic model is denoted by $Real_{\pi,A,Z}(x, k, r)$, where π denotes the protocol run by n participants according to the above specification, k is the security parameter, and r denotes the random information.

Ideal Model: F denotes an ideal function under an ideal model, S (simulator) denotes an ideal adversary, n Turing machines denote the participants and an environment Z. Under the ideal model, F defines the behavior of the desired computation and receives inputs from the participants to perform the computation, and then sends the output back to the participants. s cannot see the communication between the participants and F, but s can communicate with F. Denote the environment Z output under the ideal model by $Ideal_{F,S,Z}(x, k, r)$, where x denotes the input, k is the security parameter, and r denotes the random information.

Definition 1 (UC Security). Given a protocol π, an ideal function F, if for any PPT adversary A and the existence of an adversary S under an ideal model, the following distribution is computationally

indistinguishable for any environment Z. The protocol π is UC-realized in the presence of adversaries with an ideal function F.

$$Real_{\pi,A,Z}(x,k,r) \overset{c}{\equiv} Ideal_{F,S,Z}(x,k,r). \quad (5)$$

Hybrid Model:The F-hybrid model combines the rational model with the realistic model, extending the realistic model with an ideal function F. Each participant can interact with F. The output of Z under the hybrid model is denoted by $Hybrid^{F}_{\pi,A,Z}(x,k,r)$.

Definition 2 (security under hybrid model). Given an F and G are ideal function, π is a protocol run by n participants and π satisfies the UC-implementation ideal function G in the F-hybrid model, if for an adversary A in the hybrid model, there exists an adversary S under the ideal model such that the environment Z computation is indistinguishable from the following two distributions.

$$Real_{G,S,Z}(x,k,r) \overset{c}{\equiv} Hybrid^{F}_{\pi,A,Z}(x,k,r). \quad (6)$$

Theorem 2 (UC Compositional Security). *a UC-implementation F-protocol π, for any F-hybrid protocol ρ, has a combined protocol ρ^{π} simulating the execution of the protocol ρ, for adversary A, ideal adversary S, and no environment Z capable of distinguishing with a non-negligible probability whether it is interacting with an adversary A and the protocol ρ^{π} interacts with, or interacts with S and the protocol ρ. In other words, ρ is an F-hybrid protocol, π is a UC-implementation of F, and then there is ρ^{π} UC-realized of ρ.*

Garbled scheme

π is an n-participant protocol, x_i denotes the input of participant P_i, π_i denotes the next message function of participant P_i, when π uses $x_1,...,x_n$ as input to run as $\pi(x_1,...,x_n)$, also as the output of the protocol.

Definition 3 (Garbled scheme GC): a Garbled scheme [30, 33, 37] consists of the following polynomial-time algorithmic tuple GC=Setup,Garble,Eval, and some security features:

$Setup(1^k)$:This is a polynomial time algorithm that takes as input a security parameter and outputs a common reference string CRS.

$Garble(CRS,i,\pi_i,x_i)$:This polynomial-time algorithm takes as input the common reference string CRS, index i, π_i, and the parties' input values x_i, and outputs. (1) The next message function π_i is Garbled composed $\widetilde{\pi_i}$. (2) The input value x_i is encoded $\widetilde{x_i}$ with length l_e. (3) The corresponding coded labels $\left\{lab^{i}_{j,0}, lab^{i}_{j,1}\right\}_{j\in[n\cdot l_e]}$ after the input coding of a set of parties.

$Eval(\widetilde{\pi_i},\widetilde{x_i},lab^{i}_{\widetilde{x_1}||...||\widetilde{x_n}})$:The input $\widetilde{\pi_i}$, the encoded input set $\widetilde{x_i}$ and the corresponding $\widetilde{x_1}||...||\widetilde{x_n}$ encodes the input label $lab^{i}_{\widetilde{x_1}||...||\widetilde{x_n}}$, the output result value y or terminator $\perp$.

Correctness: for n-party agreement π and the set of inputs $\{x_i\}_{i\in[n]}$for each party we have:

$Pr[CRS \leftarrow Setup(1^k); (\widetilde{\pi_i},\widetilde{x_i},lab^{i}_{j,0},lab^{i}_{j,1}) \leftarrow Garble(CRS,i,\pi_i,x_i)\forall i\in[n] : (x_1,...,x_n) = Eval(\widetilde{\pi_i},\{\widetilde{x_i}\},lab^{i}_{\widetilde{x_1}||...||\widetilde{x_n}})] = 1$

Security:For all protocols π, all subsets of honest participants $H \in [n]$, and the inputs $H \in [n]$ chosen by each participant there exists a PPT algorithm such that:

$$\left\{CRS, \left\{\widetilde{\pi_i},\widetilde{x_i},lab^{i}_{\widetilde{x_1}||...||\widetilde{x_n}}\right\}_{i\in[n]}\right\} \overset{c}{\equiv} \left\{Sim(1^k,\pi,H,\{x_i\}_{i\notin H},\pi(x_1,...,x_n))\right\}$$

where $CRS \leftarrow Setup(1^k)$, for all $i \in [n]$ with $(\widetilde{\pi_i},\widetilde{x_i},lab^{i}_{j,0},lab^{i}_{j,1}) \leftarrow Garble(CRS,i,\pi_i,x_i)$.

Non-interactive zero-knowledge proofs

The NIZK [29, 44, 46] function is based on the zero-knowledge function in [47], which adjusts and obtains special properties of non-interactive zero-knowledge proof. The argument of NIZK is just a bit string, which anyone can use to verify the validity of the statement. The ideal function F^{R}_{nizk} is represented as follows.

Functionality F^{R}_{nizk}

Functionality F^{R}_{nizk} with n participants $P_1, P_2, ..., P_n$ and an adversary S interacting and parameterized by the NP relation R.

Step 1: Proof: input $(prove, sid, x, w)$ from P_i, ignore if $(x, w) \notin R$. Send $(prove, sid, x)$ to adversary S and wait for $(proof, sid, \pi)$, when the response is received store (sid, x, π) and send $(proof, sid, \pi)$ to participant P_i.

Step 2:Verification: input $(verify, sid, x, \pi)$ from P_j, check if (sid, x, π) has been stored, if not, send $(verify, P_j, sid, x, \pi)$ to adversary S and wait for response $(witness, w)$, when this response is received check if $(x, w) \in R$. If yes, store (sid, x, π) in this case. If (sid, x, π) has been stored then return the message $(verification, sid, x, \pi, 1)$ to P_j, otherwise return the message $(verification, sid, x, \pi, 0)$ to P_j.

The NIZK ideal function is parameterized by an NP relation R with n participants $P_1, P_2, ..., P_n., P_n$, participant P_i can send a prove request, denoted as (x, w), and the function verifies whether $(x, w) \in R$ and asks the adversary S to generate a proof π for statement x. The function stores (x, π) and returns the proof to P_i. For the other participants $\{p_j\}_{j\in[n], j\notin i}$ can send a verify request, denoted as (x, π),if (x, π) has been stored, the function outputs 1, otherwise the adversary is asked to present a proof w. If $(x, w) \in R$ the function returns 1, otherwise it returns 0.The proof for the following Theorem 3 is detailed in the literature [46].

Theorem 3 (informal). *Assuming LWE, if there exists adaptively secure NIZK arguments for NP, there exists adaptively secure NIZK arguments for NP with proof size sublinear in the circuit size of the NP relation.*

Protocol construction

In this section we construct protocols using a series of related techniques, firstly the construction of trapdoor matrices, secondly the application to secure multiparty computation using a trapdoor-based LWE encoding scheme to propose ideal functions that satisfy the properties of secure multiparty computation, and then a realistic protocol π_{smpc}.

N-participant trapdoor matrix construction

For performing trapdoor matrix construction in secure multiparty computation, we apply a variant scheme based on the trapdoor construction in [38] on secure multiparty computation. Given $m_1 = \lceil nlog(q) + \sqrt{n}\rceil, m_2 = \lceil nlog(q)\rceil, m = m_1 + m_2 = \lceil 2nlog(q) + \sqrt{n}\rceil$, the matrix $\boldsymbol{A}$ is denoted as $\boldsymbol{A} = [\boldsymbol{A_2}|\boldsymbol{A_1}], \boldsymbol{A_1} \in Z^{n\times m_2}$, $\boldsymbol{A_2} \in Z^{n\times m_1}$, a matrix $\boldsymbol{R} \in Z_q^{m_1\times m_2}$ is required to satisfy the following requirements when the threshold of $\boldsymbol{A}$. (1) $\boldsymbol{R}$ is "small". (2) Given the matrix $G \in Z_q^{n\times m_2}$, we have $\boldsymbol{A_1} = G - \boldsymbol{A_{2R}}$ and $A = [A_2|G]\begin{pmatrix} I & R \\ 0 & I \end{pmatrix}$. The process of generating $(\boldsymbol{A}, \boldsymbol{R})$: the selection matrix $\boldsymbol{R} \in_{Gaussian} Z_q^{m_1\times m_2}$, $\boldsymbol{R}$ is chosen randomly from the discrete Gaussian distribution, denoted as a trapdoor, and has $\|x_i\boldsymbol{R}\|_\infty \le \|x_i\|_\infty \lceil 2nlog(q)\rceil$. Choose a uniform distribution matrix $\boldsymbol{A_2} \in_{Uniform} Z^{n\times m_1}$ and set $\boldsymbol{A} = [\boldsymbol{A_2}|G]\begin{pmatrix} \boldsymbol{I} & \boldsymbol{R} \\ \boldsymbol{0} & \boldsymbol{I} \end{pmatrix} = [\boldsymbol{A_2}|G - \boldsymbol{A_{2R}}], \boldsymbol{A} \in Z_q^{n\times m}$.

The generation of n trapdoors is performed in the *setup* session of the protocol, and a matrix $\boldsymbol{A_i}$ with trapdoors $\boldsymbol{R_i}$ is generated corresponding to each participant P_i according to the introduction of a Common Reference String (CRS). the following is the generation algorithm for the n trapdoor matrix.

Algorithm 1 n trapdoor matrix generation algorithm

Require:
Input elements
$m_1 = \lceil nlog(q) + \sqrt{n}\rceil, m_2 = \lceil nlog(q)\rceil, m = m_1 + m_2$;
Sampling Matrix $G_i \in Z_q^{n\times m_2}$;//Input and sampling parameters
for all x_i such that $\|x_i\boldsymbol{R}\|_\infty \le \|x_i\|_\infty \lceil 2nlog(q)\rceil$ **do**
  Sampling $\boldsymbol{R_i} \in_{Gaussian} Z_q^{m_1\times m_2}$ in Gaussian distribution $D_{z^m,\sigma}$;
end for
Ensure: //Loop generation
for all i such that $1 \le i \le n$ **do**
  Sampling a uniform distribution matrix $A' \in_{Uniform} Z^{n\times m_1}$;
  Compute $\boldsymbol{A_i} = [\boldsymbol{A'}|G_i - \boldsymbol{A'}\boldsymbol{R_i}]$;//Compute the trapdoor matrix
  if $\boldsymbol{A_i}$ is not generating the $\boldsymbol{D_i}$ matrix of the current level based on the trapdoor $\boldsymbol{R_{i-1}}$ of the previous level **then**
    Resample the matrix and calculate the matrix $\boldsymbol{A_i}$;
  else
    Return the matrix $\boldsymbol{A_i}$ with trapdoors, and the corresponding trapdoor matrix $\boldsymbol{R_i}$;
  end if
end for

In the process of trapdoor matrix, we use the CRS, which stores the parameters of the participants to generate the trapdoor matrix, when the participants receive the CRS can be integrated to generate their own corresponding matrix, the whole process is only related to the security parameter k, the whole generation process is polynomial time size $poly(k)$.

SMPC in trapdoor LWE-based multilinear maps

We propose an encoding computation scheme for secure secure multiparty computation based on the graded encoding scheme mentioned in [39, 40], using a variant of its scheme applied to secure multiparty computation. A graded encoding scheme consists of the following polynomial program, $ges = (PrmGen, InstGen, Sample, Garble.enc, Eval, ZeroTest, Extract)$:

$InstGen(gp)$:Given the global parameter gp, the following processes are instantiated and generated:

(1) Use trapdoor-sampling to generate a matrix set $\boldsymbol{U_A}$ with a trapdoor set $\boldsymbol{R}$. Each participant corresponds to a trapdoor matrix under a common random reference string and has the following properties.

$$\forall \boldsymbol{R_i} \in \boldsymbol{R}, \forall \boldsymbol{A_i} \in \boldsymbol{U_A}, (\boldsymbol{A_i}, \boldsymbol{R_i}) \leftarrow trapGen(1^k, 1^n, 1^m, q). \quad (7)$$

(2) Generate the public parameters $pp{:=}(x, \{\boldsymbol{A_i} : \boldsymbol{A_i} \in \boldsymbol{U_A}\})$, where x denotes the public parameter used for the proof, and the private parameter $sp{:=}\boldsymbol{R_i} : \boldsymbol{R_i} \in \boldsymbol{R}$.

Sample(pp):Generate an input plaintext to implement sampling an LWE input $\boldsymbol{S} \leftarrow Z_q^n$.

Garble.Enc$(pp, sp, \boldsymbol{S})$: The input matrix $\boldsymbol{A_i} \in \boldsymbol{U_A}$, and the set of trapdoors R, the input $\boldsymbol{s_i} \leftarrow \boldsymbol{S}$, samples an LWE error matrix $\boldsymbol{e_i} \leftarrow \chi^m$ or $\|\boldsymbol{e_i}\| < \frac{q}{o(\sqrt{nlog(q)})}$, computes $\boldsymbol{A_{i-1}}\widetilde{\boldsymbol{D_i}} = \boldsymbol{s_i}\ \boldsymbol{A_i} + \boldsymbol{e_i}$ using the trapdoor $\boldsymbol{R_i} \in \boldsymbol{R}$, encodes the input $\boldsymbol{s_i}$ into $\widetilde{\boldsymbol{D_i}}$ and output $\widetilde{\boldsymbol{D_i}}$ and the corresponding encoded labels $\left\{lab_{j,0}^i, lab_{j,1}^i\right\}_{j\in[n\cdot l_e]}$.

Eval$(\widetilde{\boldsymbol{D_i}}, lab^i_{A||\widetilde{D_1}||...||\widetilde{D_n}})$: The calculation operations include addition and multiplication operations as follows.

N participants $P_1, P_2, ..., P_n$,with $\boldsymbol{s_1}, \boldsymbol{s_2}, \ldots, \boldsymbol{s_n}$ corresponding to the inputs of each participant, where $\boldsymbol{s_i} \leftarrow Z_q^n, i = [n]$.There are $n+1$ sets of matrices with trapdoors $\boldsymbol{U_A} = \{\boldsymbol{A}, \boldsymbol{A_1}, \ldots, \boldsymbol{A_n}\}$ and each participant encodes $\boldsymbol{s_i}$ using the corresponding matrix $\boldsymbol{A_i} \in \boldsymbol{U_A}$, P_1 encodes $\boldsymbol{A}\widetilde{\boldsymbol{D_1}} = \boldsymbol{s_1}\ \boldsymbol{A_1} + \boldsymbol{e_1}$ for its own $\boldsymbol{s_1}$, and P_2 encodes $\boldsymbol{A_1}\widetilde{\boldsymbol{D_2}} = \boldsymbol{s_2}\ \boldsymbol{A_2} + \boldsymbol{e_2}$ for its own $\boldsymbol{s_2}$ until P_n encodes $\boldsymbol{A_{n-1}}\widetilde{\boldsymbol{D_n}} = \boldsymbol{s_n}\ \boldsymbol{A_n} + \boldsymbol{e_n}$, the whole process forms a nested chain structure that generates the current matrix $\widetilde{\boldsymbol{D_i}}$ based on the matrix $\boldsymbol{A_{i-1}}$ with trapdoors at the previous level, so that the input $\boldsymbol{s_i}$ is encoded into $\widetilde{\boldsymbol{D_i}}$ and $\boldsymbol{s_i}$ is hidden.

In a multilinear maps system, given n pairwise operations from level 1 to n, $\boldsymbol{A}$ as well as $\widetilde{\boldsymbol{D_i}}, i \in [n]$, the coding results of all participants are multiplied together:

$$\begin{aligned} &\boldsymbol{A}\widetilde{\boldsymbol{D_1}}\widetilde{\boldsymbol{D_2}}\ldots\widetilde{\boldsymbol{D_n}} = (\boldsymbol{s_1}\boldsymbol{A_1} + \boldsymbol{e_1})\widetilde{\boldsymbol{D_2}}\ldots\widetilde{\boldsymbol{D_n}} \\ &= (\boldsymbol{s_1}\boldsymbol{A_1}\widetilde{\boldsymbol{D_2}} + \boldsymbol{e_1}\ \widetilde{\boldsymbol{D_2}})\widetilde{\boldsymbol{D_3}}\ldots\widetilde{\boldsymbol{D_n}} \\ &= (\boldsymbol{s_1}\boldsymbol{s_2}\boldsymbol{A_2} + \boldsymbol{s_1}\ \boldsymbol{e_2} + \boldsymbol{e_1}\ \widetilde{\boldsymbol{D_2}})\widetilde{\boldsymbol{D_3}}\ldots\widetilde{\boldsymbol{D_n}} \\ &= \ldots\ldots = \boldsymbol{s_1}\boldsymbol{s_2}\ldots\boldsymbol{s_n}\boldsymbol{A_n} + \boldsymbol{e_{noise}}\ . \end{aligned} \tag{8}$$

Where $\boldsymbol{e_{noise}}$ denotes the noise obtained by the final multiplication, which is obtained by the product of the above equation encoding the $\boldsymbol{s_1}\boldsymbol{s_2}....\boldsymbol{s_n}$ instances, performing n levels of nesting. In the information with the same order encoding can be combined with each other for addition and subtraction operations, which can be expressed as $g_i^{s_1}, g_i^{s_2}$ in the initial multilinear maps, for addition and subtraction operations to calculate $g_i^{s_1 \pm s_2}$. In the multilinear maps system with trapdoor LWE instances, for $\widetilde{\boldsymbol{D_i'}}$ with $\widetilde{\boldsymbol{D_i}}$ that has encoded $\boldsymbol{s_i'}$ with $\boldsymbol{s_i}$ of the same order i, making addition and subtraction operations yields.

$$\begin{aligned} &\boldsymbol{A_{i-1}}\widetilde{\boldsymbol{D_i'}} + \boldsymbol{A_{i-1}}\widetilde{\boldsymbol{D_i}} = \boldsymbol{A_{i-1}}\left(\widetilde{\boldsymbol{D_i}}' + \widetilde{\boldsymbol{D_i}}\right) \\ &= (\boldsymbol{s_{i'}} + \boldsymbol{s_i})\ \boldsymbol{A_{i-1}} + (\boldsymbol{e_i}' + \boldsymbol{e_i}). \end{aligned} \tag{9}$$

If there are multiple $\widetilde{\boldsymbol{D_i}}$ of the same order, we can get at this point we can get $\boldsymbol{A_{i-1}}\widetilde{\boldsymbol{D_{i_1}}} + \boldsymbol{A_{i-1}}\widetilde{\boldsymbol{D_{i_2}}} + ... + \boldsymbol{A_{i-1}}\widetilde{\boldsymbol{D_{i_n}}} = \sum_{i=i_1,i_2,...,i_n} \boldsymbol{s_i} + \boldsymbol{e'_{noise}}$, the same can be obtained from the operation of subtraction, at this point the multilinear maps system to achieve the basic operation.

ZeroTest$(pp, \boldsymbol{A}\widetilde{\boldsymbol{D_i}})$: Given the matrix $\widetilde{\boldsymbol{D_i}}$ after the LWE encoding input, the result obtained by the Operation operation combined with the multilinear maps operation,If and only if $\left\|\boldsymbol{A} \cdot \widetilde{\boldsymbol{D_i}}\right\|_{\boldsymbol{A}\in\boldsymbol{U_A}\setminus\boldsymbol{A_i}, i=[n]} \leq \frac{q}{o(\sqrt{nlog(q)})}$,the *ZeroTest* program outputs 1.

Extract$(pp, \boldsymbol{A}\widetilde{\boldsymbol{D_i}})$: The extractor takes as input the public parameter pp, $\boldsymbol{A}\widetilde{\boldsymbol{D_i}}$, and outputs a string that represents a λ bit.

For the graded encoding scheme, Fig. 2 represents the process of computation of encoded inputs for each participant, if the noise does not exceed a certain threshold value, it is a bounded value, and the computed information can be extracted from the *ZeroTest* program and *Extract* program, *ZeroTest* program and *Extract* program for correctness are detailed in [40].The operation of multiparty computation in a multilinear maps system is given below.

Algorithm 2 Evaluation algorithm

Require:
Input $\boldsymbol{e_i} \leftarrow \chi^m$, $n+1$ trapdoor matrix set
$\boldsymbol{U_A} = \{\boldsymbol{A}, \boldsymbol{A_1}, \ldots, \boldsymbol{A_n}\}$; Encoding matrix set $\left\{\widetilde{\boldsymbol{D_i}}, \widetilde{\boldsymbol{D_i'}}\right\}_{i\in[n]}$,and the corresponding encoded labels $\left\{lab_{j,0}^i, lab_{j,1}^i\right\}_{j\in[n\cdot l_e]}$;
Ensure:
1: **for all** i to n **do**
2: **if** $\widetilde{\boldsymbol{D_i}}$ and $\widetilde{\boldsymbol{D_i'}}$ are coded at the same level **then**
3: Compute $\boldsymbol{A_{i-1}}(\widetilde{\boldsymbol{D_i}} + \widetilde{\boldsymbol{D_i'}})$; //Compute addition operations
4: **else**
5: break;
6: **end if**
7: return $\boldsymbol{A_{i-1}}\widetilde{\boldsymbol{D_{i_1}}} + \boldsymbol{A_{i-1}}\widetilde{\boldsymbol{D_{i_2}}} + ... + \boldsymbol{A_{i-1}}\widetilde{\boldsymbol{D_{i_n}}}$ where $\left\{\widetilde{\boldsymbol{D_{i_1}}}, \widetilde{\boldsymbol{D_{i_2}}}, \ldots, \widetilde{\boldsymbol{D_{i_n}}}\right\}$ denotes the same-level coding matrix; $\prod \boldsymbol{A}\widetilde{\boldsymbol{D_i}} \leftarrow Eval(\widetilde{\boldsymbol{D_i}}, \left\{lab_{j,0}^i, lab_{j,1}^i\right\}_{j\in[n\cdot l_e]})_{i\in n}$;//Compute multiplication operations
8: **end for**

Ideal function F_{smpc}

There are n mutually distrustful participants $P_1, P_2, ..., P_n$ want to jointly compute in polynomial time the computable function $f(x_1, x_2, ..., x_n) = (y_1, y_2, ..., y_n)$, where $x_1, x_2, ..., x_n$ are the input variables, $y_1, y_2, ..., y_n$ are the output values. The protocol π of a multiparty computation of a computational function should satisfy the following requirements:

(1) Privacy: The input information of each participant is invisible with respect to other participants, each participant does not obtain more information from other participants than what is inferred from its own results.
(2) Correctness: the protocol π can correctly calculate the function f and return the corresponding correct result.

(3) Security: each party gets the corresponding correct output, and no other additional information can be obtained.

In this paper, we design a secure multiparty computation protocol based on the above requirements and design a secure multiparty computation ideal function $F_{smpc} : \left(\{0,1\}^{l_{in}}\right)^n \rightarrow \{0,1\}^{l_{out}}$ as shown below:

Ideal function F_{smpc}

Given a function $f : X \rightarrow Y, X \in Z_q^*, Y \in R \cup \perp$, n participants $P_1, P_2, ..., P_n$, adversary S,F_{smpc} is treated as follows.

Step 1. When input $(P_i, input, sid, x)$ is received from participant $P_i, i = [n], x \in X$, the sid is recorded and $(P_i, input, sid)$ is sent to adversary S.

Step 2. When $(P_i, compute, sid)$ is received from participant $P_i, i = [n]$, where $P_i.x \in Z_q^*$, compute $\prod_{i=[n]} P_i.x, y \in Y$.

Step 3. When $(P_i, output, sid)$ is received from participant $P_i, i = [n]$.

(1) Send $\perp$ if participant P_i is identified as Corrupted, otherwise send y.

(2) Send y for all participants.

Step 4. When $\left(P_i, corrupt - input, sid, x'\right)$ is received from the adversary S:

(1) Record that P_i is a Corrupted participant and record the adversary input x'.

(2) If the current participant P_i sends input x and does not write output y to P_i's tape in the output phase, then record x' if the adversary provides x', otherwise record x.

Step 5. When $(P_i, corrupt - output, sid)$ is received from adversary S:

(1) Check whether the participant P_i has been marked as Corrupted, ignore if it has been marked, otherwise mark P_i as Corrupted.

(2) Forward y to adversary S.

(3) If the current participant P_i sends input x and does not write output y to P_i's tape in the output phase, then record x' if the adversary provides x', and vice versa, then record x and make the calculation and send it again.

Cloud-based secure multiparty computation protocol π_{smpc}

This section constructs a 2-round protocol π_{smpc} under the LWE assumption, based on LWE encoding the input, using multilinear maps operations to compute it locally before transmitting it over the broadcast channel, with the following protocol. A total of three phases which including prephase and 2 rounds of interaction processes, as included in Fig. 3, enable each participant to safely compute each encoded input, through 2 rounds of communication and local computation, so as to compute the corresponding computational function result values.

π_{smpc}

Privacy input: for all $i \in [n]$, each participant P_i,the corresponding input $\boldsymbol{s_i} \leftarrow Z_q^n$.

Public input: a graded encoding scheme ges, a common random reference string and a garble circuit GC of depth d.

The protocol is as follows:

NIZK on input: Participant Access Functionality F_{nizk}^R Cloud. The cloud has the input $\boldsymbol{s_i} \leftarrow Z_q^n$ performs the following steps noted as proof:

(1) Sample the LWE instance noise $\boldsymbol{e_i} \leftarrow \chi^m$, where $\chi \leftarrow D_{z^m,\sigma}$.

(2) Compute the input coding $Sample(pp) : \boldsymbol{S} \leftarrow Z_q^n, \boldsymbol{s_i} \in \boldsymbol{S}$. The computational matrix $\widetilde{\boldsymbol{D_i}}$ satisfies $\boldsymbol{A_{i-1}}\widetilde{\boldsymbol{D_i}} = \boldsymbol{s_i}\boldsymbol{A_i} + \boldsymbol{e_i}$.

(3) For $G_i \in Z_q^{n \times \lceil nlog(q) \rceil}$, $\boldsymbol{R_i} \in_{Gaussian} Z_q^{m_1 \times m_2}$, $\boldsymbol{A_i} = \left[\boldsymbol{A'} \middle| G_i - \boldsymbol{A'}\boldsymbol{R_i}\right]$.

(4) Send the output to the participant.

Round 1: For all participants P_i, perform the following operations:

(1) Instantiate the global parameter $InstGen(gp)$ and use the threshold generation algorithm $trapGen$ to generate the trapdoor $\boldsymbol{R_i}$ and the corresponding trapdoor matrix $\boldsymbol{A_i} : (\boldsymbol{A_i}, \boldsymbol{R_i}) \leftarrow trapGen(1^k, 1^n, 1^m, q)$.

(2) Encode the input $\boldsymbol{s_i}$ by the encoding scheme $ges, \left(\widetilde{\boldsymbol{D_i}}, \left\{lab_{j,0}^i, lab_{j,1}^i\right\}_{j \in [n \cdot l_e]}\right) \leftarrow Garble.Enc(pp, sp, s_i)$.

(3) Generate session id sid and transmit $\left(P_i, input, sid, \left(\widetilde{\boldsymbol{D_i}}, \left\{lab_{j,0}^i, lab_{j,1}^i\right\}_{j \in [n \cdot l_e]}, proof\right)\right)$ on the broadcast channel.

Round 2: For all participants P_i, perform the following operations:

(1) Record $(P_j, input, sid,)$ when participant P_i receives $\left(P_j, input, sid, \cdot\right)_{j \in [n] \backslash i}$ and verify the $proof$, ignoring the subsequent $(P_j, input, \cdot)$.

(2) The input $(\widetilde{\boldsymbol{D_i}}, \left\{lab_{j,0}^i, lab_{j,1}^i\right\}_{j \in [n \cdot l_e]})$ that passes the validation is added to the operation, and if the other n-1 (considering no participant suspended, if there is a terminated participant, the terminated participant's input needs to be removed) coded inputs have been received, then $Eval(\widetilde{\boldsymbol{D_i}}, lab^i_{\boldsymbol{A}||\widetilde{\boldsymbol{D}}_1||...||\widetilde{\boldsymbol{D}}_n})$, the output result $y := \prod_{i=[n]} \boldsymbol{s_i}\boldsymbol{A_n} + \boldsymbol{e_{noise}}$.

(3) output$(P_i, output, y)$, and transmit on the broadcast channel, if there is a participant terminated during the protocol output $(P_i, output, sid_{abort}, y)$, indicating that it is out of the protocol and the other participants will not be calculated for their input join.

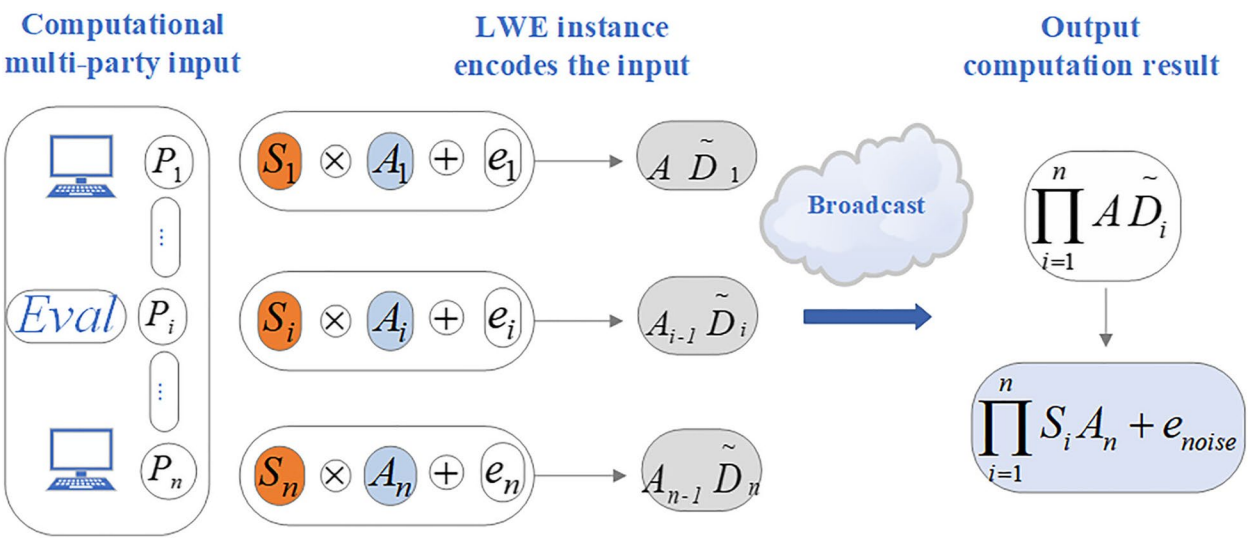

Fig. 2 The process of computation of encoded inputs for each participant

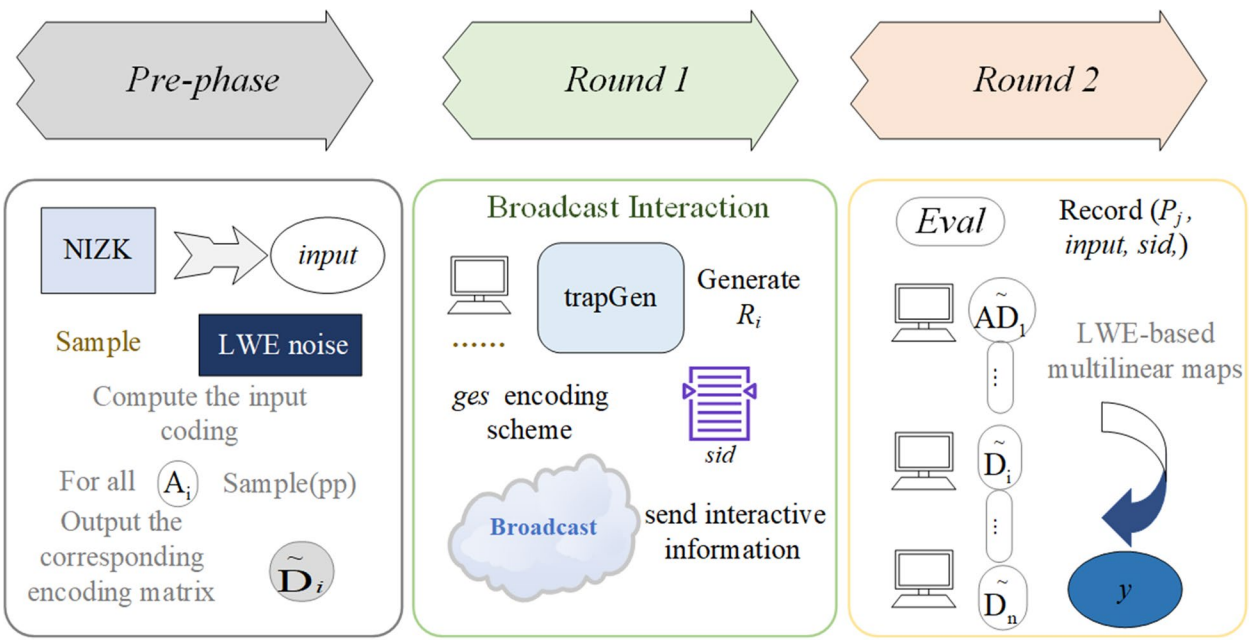

Fig. 3 π_{smpc} Flow Chart

After each participant receives the output result y on the broadcast channel, a *ZeroTesttest* will be performed on the result. If $\prod_{i=[n]} s_i$ is 0, only the noise distribution remains, and the threshold of the noise is used to determine whether the result encodes a value of 0. By simply designing the circuit using a combination of multilinear maps, the specific information contained in the ciphertext can be gradually inferred by a *ZeroTest* algorithm and extracted using a The extractor *Extract* is a very randomized extractor. However, *ZeroTest* cannot reveal too much information about the ciphertext, so we can use *ZeroTest* to extract part of the information for our computational purpose with certain security.

Semi-Malicious Security. A semi-malicious protocol can be defined over a broadcast channel where the input must be encrypted and then transmitted. This scheme is based on the LWE assumption that the n inputs are all elements in Z_q^n in an honest majority setting of the participants, and the inputs are encoded and then broadcast for transmission by an LWE instance, with each participant

using confusion circuit locally on the encoded inputs and the output is broadcasted.

Theorem 4 (Theorem 1, restated). *Assume the existence of a special ges scheme and NIZK scheme with LWE assumption, and that* $F : \left(\{0,1\}^{l_{in}}\right)^n \to \{0,1\}^{l_{out}}$ *is an effectively computable function of depth d. The function* F_{smpc} *can be implemented by a communication under an honest majority of the hybrid model two-round protocol UC realized with* $poly(l_{in}, l_{out}, d, k, n)$ *complexity and tolerates the presence of semi-malicious adversaries.*

Security demonstration

Secure Multi-Party Computation has two kinds of security, static security and adaptive security, static security means that during the operation of the MPC protocol, the security of the protocol can be guaranteed as long as the number of participants does not exceed the maximum number of participants predefined by the protocol. In other words, in the static security model, once the number of participants is determined, then the security of the protocol can be guaranteed. Adaptive security means that during the operation of the MPC protocol, even if there are malicious participants trying to interfere with the operation of the protocol, the security of the protocol is still guaranteed. the operation of the protocol, the security of the protocol can still be guaranteed. In realistic protocols, since a matrix U_A with trapdoors is used to generate a series of $\boldsymbol{D}$-matrices, will the original privacy inputs be exposed in the presence of trapdoors and also the encoded D-matrices of the privacy inputs are disclosed? A specific elaboration is given in [40].According to the encoding rules, the two matrices $\boldsymbol{A_{i-1}}$ with trapdoor are nested with $\boldsymbol{A_i}$, denoted as $\boldsymbol{A_{i-1}\widetilde{D_i} = s_i A_i + e_i}$, and when encoding to the last one $\boldsymbol{A_{n-1}\widetilde{D_n} = s_n\, A_n + e_n}$, the trapdoor of matrix $\boldsymbol{A_n}$ is not involved in the calculation, if $\boldsymbol{s_n}$ distribution is randomized enough, then the whole encoding process is an LWE instance. According to the LWE assumption, the last encoding process is represented by a uniform random distribution matrix $\triangle$, $\boldsymbol{A_{n-1}\widetilde{D_n}} = \triangle$, which becomes known as the product of $\boldsymbol{A_{n-1}}$ and $\boldsymbol{\widetilde{D_n}}$ as a uniform random distribution matrix $\triangle$. Given a trapped $\boldsymbol{A_{n-1}}$ with a trapdoor and a uniformly randomly distributed matrix $\triangle$, if $\boldsymbol{\widetilde{D_n}}$ can be generated without this trapdoor, then $\boldsymbol{A_{n-1}}$ with $\boldsymbol{\widetilde{D_n}}$ does not give away information about the trapdoor. Suppose there are two environments, real and simulated, and in the real environment using the trapdoor of $\boldsymbol{A_{n-1}}$ trapdoor to generate $\boldsymbol{\widetilde{D_n}}$ in the real environment and not using $\boldsymbol{A_{n-1}}$ trapdoor to generate in the simulated environment, the results of the two are computationally indistinguishable.The following lemma was obtained according to the literature [40]. If the LWE assumption holds, the input encoded based on the trapdoor LWE assumption is secure.

Theorem 5 *The ideal function* F_{smpc} *is a polynomial-time computable deterministic function with N inputs and one output, and the protocol* $ges = (PrmGen, InstGen, Sample, Garble.enc, Eval, ZeroTest, Extract)$ *is Secure multiparty computation in trapdoor LWE-based multilinear maps operations, then the protocol* π_{smpc} *UC realized the ideal function* F_{smpc} *in the honest majority participant setting.*

Proof

To demonstrate security under an honest majority of participants based on a valid PPT simulator Sim, Adv represents a static semi-malicious adversary and the simulator is simulated as follows.□

The Simulator: In the first round, it can encrypt the false inputs $\widehat{\boldsymbol{s_i}}$ and get the inputs of the other participants on the "witness tape", which can encode the inputs. And send these inputs to the ideal function and receive the corresponding output y. After getting this result, the simulator computes $\widetilde{y} \leftarrow Sim.eval(\widetilde{\pi_i}, \widehat{s_i}, \widetilde{\boldsymbol{D_i}}, lab^i_{A||\widetilde{D_1}||\ldots||\widetilde{D_n}})$ and broadcast it.

Hybrid Games: Define a series of hybrid games to demonstrate the indistinguishability of real and ideal scenarios:

$$Real_{\pi_{smpc},Adv,Z} \overset{c}{\equiv} Ideal_{F_{smpc},Sim,Z}. \tag{10}$$

The output of the entire environment Z is used as the output of each game.

The game $Real_{\pi_{smpc},Adv,Z}$: In the real world, the protocol π_{smpc} is executed in the environment Z in the presence of a semi-malicious adversary Adv.

The game $HYB^1_{\pi_{smpc},Adv,Z}$: In this game, we modify the experiment of $Real_{\pi_{smpc},Adv,Z}$ as follows, introducing the F^R_{nizk}-hybrid model, where each participant P_i encodes its own input followed by $(prove, sid, x, \boldsymbol{s_i})$ to F^R_{nizk}, outputs a Proof π, and sends $(proof, sid, \widetilde{\boldsymbol{D_i}}, \pi)$ for broadcast, and when participant $\{P_j\}_{j\in[n]\setminus i}$ receives the message, P_j sends a verification request to $F^R_{nizk}(verify, sid, x, \pi)$, and F^R_{nizk} returns 1 or 0 after verification.

Claim 1 $Real_{\pi_{smpc},Adv,Z} \overset{c}{\equiv} HYB^1_{\pi_{smpc},Adv,Z}$ *Proving the indistinguishability of realistic protocols under hybrid models.*

$$Pr[(x,\widehat{\boldsymbol{D_i}}) \in \boldsymbol{R}|\widehat{\boldsymbol{D_i}}:= Sim\begin{pmatrix} \pi:=\pi \leftarrow F^R_{nizk}(\boldsymbol{s_i}) \\ \widehat{\boldsymbol{s_i}} \leftarrow Sample(pp,1^k) \\ \pi' \leftarrow Sim(x,\widehat{\boldsymbol{s_i}},\pi) \\ \widehat{\boldsymbol{D_i}} \leftarrow Sim.ecn(pp,sp,\widehat{s_i}) \end{pmatrix}] \leq negligible \quad (11)$$

Proof

Let Adv be the adversary in the real environment and Sim denote the adversary in the ideal environment such that for any environment Z only the real or ideal environment can be distinguished with negligible probability, and for the adversary Sim in the ideal environment, any input from the environment Z is sent to Adv and any output of Adv is regarded as the output of Sim.For the adversary Sim in interaction with the ideal function F^R_{nizk}, provide input $\boldsymbol{s_i}$, and when $(proof, sid, \widetilde{\boldsymbol{D_i}}, \pi)$ is received from F^R_{nizk}, emulate an identical message for Adv. When the real-world adversary Adv taps participant P_i, then the adversary Sim in the ideal environment also taps participant P_i and forwards all internal states to Adv.If at this time the adversary Adv replaces the message $\boldsymbol{s_i}$ with the false message $\widehat{\boldsymbol{s_i}}$ on behalf of the participant P_i and forges the proof π' against π and broadcasts the message $(proof, sid, \widehat{\boldsymbol{D_i}}, \pi')$, when the other participants receive this message and verify the proof when , query whether F^R_{nizk} has stored π', and since π' is not generated by F^R_{nizk}, determine whether $(x,\widehat{\boldsymbol{D_i}}) \in R$. According to the security of LWE assumptions and the security of zero-knowledge proofs, only the input encoded by LWE instances can pass the verification , in other words, the probability that a non-LWE encoded input passes verification is negligible.□

So $HYB^1_{\pi_{smpc},Adv,Z}$ is indistinguishable from $Real_{\pi_{smpc},Adv,Z}$ computation, and the scheme under the hybrid model is semantically secure.

The game $HYB^2_{\pi_{smpc},Adv,Z}$: Unlike $HYB^1_{\pi_{smpc},Adv,Z}$, a realistic proof protocol π_{nizk} will be used instead of the ideal function F^R_{nizk}, modifying the proof process to a local circuit for computation.

Claim 2 $HYB^1_{\pi_{smpc},Adv,Z} \overset{c}{\equiv} HYB^2_{\pi_{smpc},Adv,Z}$

Proof

realistic zero-knowledge proof protocol notated as π_{nizk}, composed by the garbled circuit GC, first generates the proof parameters $(S_p, S_v) \leftarrow GC.Setup(1^K)$ through the circuit, which in turn computes the proof $\pi \leftarrow GC.Prove(S_p, x, \widetilde{\boldsymbol{D_i}})$, sends S_v, π is broadcasted and sent at the first round, and the other participants compute $GC(x, \widetilde{\boldsymbol{D_i}})$ through the NAND gate for Verify $0/1 \leftarrow GC.Verify(S_v, x, \pi)$. If the LWE assumption holds, since the probability that an adversary performs a pseudo-proof under a protocol with honest majority participants and is adopted by honest participants is negligible, for environment Z, it does not distinguish whether it is in the environment where the protocol π_{nizk} interacts with Adv or in the environment where F^R_{nizk} interacts with Sim. In other words, if the LWE assumption holds, the protocol π_{nizk} can UC to achieve the ideal function F^R_{nizk}.□

The game $Ideal_{F_{smpc},Sim,Z}$: computes the ideal function F_{smpc} and outputs the result correctly under the ideal model.

Claim 3 $HYB^2_{\pi_{smpc},Adv,Z} \overset{c}{\equiv} Ideal_{F_{smpc},Sim,Z}$

Proof

experiments by the semantic security of the underlying ges scheme, encryption of the input by LWE assumptions, and then computation using multilinear maps operations, encryption is computationally indistinguishable, π_{smpc} is able to compute the encoded input correctly and get a correct in the presence of semi-malicious adversaries, honest majority of participants output, and since the protocol π_{nizk} can UC the ideal function F^R_{nizk}, from Theorem 2 it follows that the protocol π_{smpc} can UC the ideal function F_{smpc}, then $HYB^2_{\pi_{smpc},Adv,Z}$ and $Ideal_{F_{smpc},Sim,Z}$ computation is indistinguishable.□

Combining the above statements, we get $Real_{\pi_{smpc},Adv,Z} \overset{c}{\equiv} Ideal_{F_{smpc},Sim,Z}$, which leads to the proof of Theorem 5.

To conclude, Tables 1 and 2 summarize the previous work and the results of this paper in an honest majority setting, the main parameters considered are security, number of rounds, communication complexity, setup settings, etc. Under the LWE assumption, this scheme requires only 2 rounds of communication interactions for secure distributed multi-party secure computation and achieves static security in an honest majority of settings. Compared with previous work, this paper optimizes the number of rounds of secure multiparty computation and reduces the Setup Size, and the communication overhead is sublinear. Although static security is achieved, which already meets the security requirements in most scenarios, this is a minor limitation of the work in this paper, and research improvements for further adaptive security are necessary in future work.

Table 2 This is an additional description of Table 1

Setup size	Setup type	Computation Complexity	Assumptions
$poly(k, d)$	Threshold PKI	$poly(l_{in}, l_{out}, k, d, n)$	LWE, NIZK
–	–	$poly(\|C\|, k)$	PKE and zaps
$poly(k, d)$	CRS	$poly(l_{in}, l_{out}, k, d, n)$	LWE, NIZK
–	–	$poly(\|C\|, k)$	VSS
$poly(l_{in}, k, d, n)$	CRS	$poly(l_{in}, l_{out}, k, d, n)$	LWE, NIZK
$poly(\|C\|, k)$	Ref	$poly(\|C\|, k)$	OWF, iO
$poly(k)$	CRS	$poly(l_{in}, l_{out}, k, d, n)$	LWE, NIZK

Conclusions

Cloud Secure MultiParty Computation (CSPC) is suitable for a number of application prospects such as cloud-based data streaming information sharing, data trading and e-auctions in distributed environments, for which CSPC provides a secure computation as well as privacy guarantees. In this paper, we combine the concept of cloud computing and secure multiparty computation and use the harder polynomial time puzzle assumption to provide the security concept of the protocol as well as the strength, based on the LWE assumption, the input of the participants is encoded using LWE instances with lattice trapdoor under a graded encoding scheme and transmitted over the broadcast channel, the execution of the protocol is computed by multilinear maps to achieve the optimization of the number of rounds of the secure multiparty computation protocol on the cloud, the communication sublinear overhead, and in the UC framework , the protocol security is achieved through UC security implementation. In future work, it is an important research direction to achieve adaptive security of secure multiparty computation protocols with guaranteed round count optimization and low communication overhead, by combining stronger cryptographic primitives and related techniques to achieve adaptive security of the protocols, while the rise of quantum cryptography also points to a direction for the development of secure multiparty computation.

Acknowledgements
This research was supported by both State Key Laboratory of Public Big Data, College of Computer Science and Technology that of Guizhou University.

Authors' contributions
Y.L. was a major contributor in writing the manuscript as a 1st Author and others were Co-Corresponding Authors. Y.C. and T.L. proposed some important ideas. C.T. and H.D. gave some suggestions for this paper. All authors read and approved the final manuscript.

Funding
This research was supported by Foundation of National Natural Science Foundation of China(61962009 and 62202118), and Top Technology Talent Project from Guizhou Education Department([2022]073).

Availability of data and materials
Data sharing is not applicable to this paper as no datasets were generated or analyzed during the current study.

Declarations

Competing interests
The authors declare no competing interests.

Received: 11 April 2023 Accepted: 31 December 2023
Published online: 22 January 2024

References

1. Zhou X, He Yang X, Ma J, Wang KIK (2021) Energy-efficient smart routing based on link correlation mining for wireless edge computing in iot. IEEE Internet Things J 9:14988–14997
2. Zhou X, Liang W, Yan K, Li W, Wang KIK, Ma J, Jin Q (2023) Edge-enabled two-stage scheduling based on deep reinforcement learning for internet of everything. IEEE Internet Things J 10:3295–3304
3. He Q, Tan S, Chen F, Xu X, Qi L, Hei X, Zomaya A, Jin H, Yang Y (2023) Edindex: Enabling fast data queries in edge storage systems. ACM SIGIR 675–685
4. Yuan L, He Q, Chen F, Zhang J, Qi L, Xu X, Xiang Y, Yang Y (2021) Csedge: Enabling collaborative edge storage for multi-access edge computing based on blockchain. IEEE Trans Parallel Distrib Syst PP:1–1
5. Qi L, Yang Y, Zhou X, Rafique W, Ma J (2022) Fast anomaly identification based on multiaspect data streams for intelligent intrusion detection toward secure industry 4.0. IEEE Trans Ind Inform 18:6503–6511
6. Zhou X, Xu X, Liang W, Zeng Z, Yan Z (2021) Deep-learning-enhanced multitarget detection for end-edge-cloud surveillance in smart iot. IEEE Internet Things J 8:12588–12596
7. Qi L, Lin W, Zhang X, Dou W, Xu X, Chen J (2022) A correlation graph based approach for personalized and compatible web apis recommendation in mobile app development. IEEE Trans Knowl Data Eng 35:5444–5457
8. Wu S, Shen S, Xu X, Chen Y, Zhou X, Liu D, Xue X, Qi L (2023) Popularity-aware and diverse web apis recommendation based on correlation graph. IEEE Trans Comput Soc Syst 10:771–782
9. Li Z, Xu X, Hang T, Xiang H, Cui Y, Qi L, Zhou X (2022) A knowledge-driven anomaly detection framework for social production system. IEEE Trans Comput Soc Syst 1–14
10. Dai H, Yu J, Li M, Wang W, Liu AX, Ma J, Qi L, Chen G (2022) Bloom filter with noisy coding framework for multi-set membership testing. IEEE Trans Knowl Data Eng 35:6710–6724
11. Xu X, Gu JF, Yan H, Liu W, Qi L, Zhou X (2023) Reputation-aware supplier assessment for blockchain-enabled supply chain in industry 4.0. IEEE Trans Ind Inf 19:5485–5494
12. Chaudhary R, Aujla GS, Garg S, Kumar N, Rodrigues JJ (2018) Sdn-enabled multi-attribute-based secure communication for smart grid in iiot environment. IEEE Trans Ind Inform 14:2629–2640
13. Luo Y, Chen Y, Li T, Wang Y, Yang Y, Yu X (2022) An entropy-view secure multiparty computation protocol based on semi-honest model. J Organ End User Comput 34:1–17
14. Li T, Wang Z, Yang G, Cui Y, Chen Y, Yu X (2021) Semi-selfish mining based on hidden markov decision process. Int J Intell Syst 36:3596–3612
15. Li T, Chen Y, Wang Y, Wang Y, Zhao M, Zhu H, Tian Y, Yu X (2020) Yang Y (2020) Rational protocols and attacks in blockchain system. Secur Commun Netw 8839047(1–8839047):11
16. Sun J, Chen Y, Li T, Liu J, Yang Y (2021) Psspr: A source location privacy protection scheme based on sector phantom routing in wsns. In: 2021 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech), AB, Canada, p 334–340
17. Li T, Wang Z, Chen Y, Li C, Jia Y, Yang Y (2021) Is semi-selfish mining available without being detected? Int J Intell Syst 37:10576–10597

18. Wang Y, Li T, Liu M, Li C, Wang H (2022) Stsiiml: Study on token shuffling under incomplete information based on machine learning. Int J Intell Syst 37:11078–11100
19. Zhao C, Zhao S, Zhao M, Chen Z, Gao CZ, Li H, Tan YA (2019) Secure multi-party computation: Theory, practice and applications. Inf Sci 476:357–372
20. Yao ACC (1982) Protocols for secure computations. In: 23rd Annual Symposium on Foundations of Computer Science (sfcs 1982), Chicago, IL, USA, pp 160–164
21. Goldreich O, Micali S, Wigderson A (1987) How to play any mental game. In: Proceedings of the nineteenth annual ACM symposium on Theory of computing, New York, NY, United States pp 218–229
22. Abraham I, Asharov G, Yanai A (2022) Efficient perfectly secure computation with optimal resilience. J Cryptol 35:66–96
23. Ananth PV, Choudhuri AR, Jain A (2017) A new approach to round-optimal secure multiparty computation. In: Katz, J., Shacham, H. (eds) Advances in Cryptology – CRYPTO 2017. CRYPTO 2017. Lecture Notes in Computer Science(), Springer, Cham, vol 10401, pp 468–499
24. Ananth PV, Choudhuri AR, Goel A, Jain A (2018) Round-optimal secure multiparty computation with honest majority. In: Shacham, H., Boldyreva, A. (eds) Advances in Cryptology – CRYPTO 2018. CRYPTO 2018. Lecture Notes in Computer Science(), Springer, Cham, vol 10992, pp 395–424
25. Cohen R, Garay JA, Zikas V (2020) Broadcast-optimal two-round mpc. Adv Cryptol EUROCRYPT 2020 12106:828–858
26. Quach W, Wee H, Wichs D (2018) Laconic function evaluation and applications. In: 2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS), Paris, France, pp 859–870
27. Patra A, Srinivasan A (2021) Three-round secure multiparty computation from black-box two-round oblivious transfer. In: Malkin, T., Peikert, C. (eds) Advances in Cryptology – CRYPTO 2021. CRYPTO 2021. Lecture Notes in Computer Science(), Springer, Cham, vol 12826, pp 185–213
28. Beaver D, Micali S, Rogaway P (1990) The round complexity of secure protocols. In: Symposium on the Theory of Computing, New York, NY, United States, pp 503–513
29. Gordon SD, Liu FH, Shi E (2015) Constant-round mpc with fairness and guarantee of output delivery. In: Gennaro, R., Robshaw, M. (eds) Advances in Cryptology -- CRYPTO 2015. CRYPTO 2015. Lecture Notes in Computer Science(), Springer, Berlin, Heidelberg, vol 9216, pp 63–82
30. Boyle E, Gilboa N, Ishai Y (2016) Breaking the circuit size barrier for secure computation under ddh. In: Robshaw, M., Katz, J. (eds) Advances in Cryptology – CRYPTO 2016. CRYPTO 2016. Lecture Notes in Computer Science(), Springer, Berlin, Heidelberg, vol 9814, pp 509–539
31. Garg S, Srinivasan A (2018) Two-round multiparty secure computation from minimal assumptions. J ACM 69:1–30
32. Hazay C, Orsini E, Scholl P, Soria-Vazquez E (2018) Tinykeys: A new approach to efficient multi-party computation. J Cryptol 35:1–66
33. Canetti R, Poburinnaya O, Venkitasubramaniam M (2017) Equivocating yao: constant-round adaptively secure multiparty computation in the plain model. In: Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, New York, NY, United States, pp 497–509
34. Chen Y, Dong S, Li T, Wang Y, Zhou H (2021) Dynamic multi-key fhe in asymmetric key setting from lwe. IEEE Trans Inf Forensic Secur 16:5239–5249
35. Mukherjee P, Wichs D (2016) Two round multiparty computation via multi-key fhe. In: Fischlin, M., Coron, JS. (eds) Advances in Cryptology – EUROCRYPT 2016. EUROCRYPT 2016. Lecture Notes in Computer Science(), Springer, Berlin, Heidelberg, vol 9666, pp 735–763
36. Brakerski Z, Halevi S, Polychroniadou A (2017) Four round secure computation without setup. In: Kalai, Y., Reyzin, L. (eds) Theory of Cryptography. TCC 2017. Lecture Notes in Computer Science(), Springer, Cham, vol 10677, pp 645–677
37. Garg S, Srinivasan A (2017) Garbled protocols and two-round mpc from bilinear maps. 2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS), Berkeley, CA, USA, pp 588–599
38. Micciancio D, Peikert C (2012) Trapdoors for lattices: Simpler, tighter, faster, smaller. IACR Cryptol ePrint Arch 2011:501
39. Garg S, Gentry C, Halevi S (2013) Candidate multilinear maps from ideal lattices. In: Johansson, T., Nguyen, P.Q. (eds) Advances in Cryptology – EUROCRYPT 2013. EUROCRYPT 2013. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, vol 7881, pp 1–17
40. Gentry C, Gorbunov S, Halevi S (2015) Graph-induced multilinear maps from lattices. In: Dodis, Y., Nielsen, J.B. (eds) Theory of Cryptography. TCC 2015. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, vol 9015, pp 498–527
41. Ciampi M, Ostrovsky R, Siniscalchi L, Visconti I (2017) Round-optimal secure two-party computation from trapdoor permutations. In: Kalai, Y., Reyzin, L. (eds) Theory of Cryptography. TCC 2017. Lecture Notes in Computer Science(), Springer, Cham, vol 10677, pp 678–710
42. Dachman-Soled D, Katz J, Rao V (2015) Adaptively secure, universally composable, multiparty computation in constant rounds. In: Dodis, Y., Nielsen, J.B. (eds) Theory of Cryptography. TCC 2015. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, vol 9015, pp 586–613
43. Canetti R, Goldwasser S, Poburinnaya O (2015) Adaptively secure two-party computation from indistinguishability obfuscation. IACR Cryptol ePrint Arch 2014:845
44. Asharov G, Jain A, López-Alt A, Tromer E, Vaikuntanathan V, Wichs D (2012) Multiparty computation with low communication, computation and interaction via threshold fhe. IACR Cryptol ePrint Arch 2011:613
45. Regev O (2005) On lattices, learning with errors, random linear codes, and cryptography. In: Symposium on the Theory of Computing, New York, NY, United States, pp 84–93
46. Cohen R, Shelat A, Wichs D (2019) Adaptively secure mpc with sublinear communication complexity. In: Boldyreva, A., Micciancio, D. (eds) Advances in Cryptology – CRYPTO 2019. CRYPTO 2019. Lecture Notes in Computer Science(), Springer, Cham, vol 11693, 30–60
47. Canetti R (2001) Universally composable security: a new paradigm for cryptographic protocols. In: Proceedings 2001 IEEE International Conference on Cluster Computing, Newport Beach, CA, USA, pp 136–145
48. Hazay C, Venkitasubramaniam M (2016) Composable adaptive secure protocols without setup under polytime assumptions. In: Hirt, M., Smith, A. (eds) Theory of Cryptography. TCC 2016. Lecture Notes in Computer Science(), Springer, Berlin, Heidelberg, vol 9985, pp 400–432

Publisher's Note