Received: May 27, 2020. Revised: July 2, 2020.

INASS

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Secure and Verifiable Multi-Party Computation Using Indistinguishability Obfuscation

Smita Chaudhari[1,2]* Gandharba Swain[1] Pragnyaban Mishra[1]

[1]*Department of Computer Science and Engineering,*
Koneru Laxmaiah Education Foundation, Vaddeswaram-522502, Guntur, Andhra Pradesh, India
[2]*Dr. D. Y. Patil Institute of Technology, Pimpri, Pune*
* Corresponding author's Email: smita.m.c@gmail.com

Abstract: In most practical cloud computing applications such as e-voting, auctions, health, and financial applications or cloud services in common, to prove the exactness of outsourced data is one of the major needs today. Most of the time, third party auditing is employed for this task. This auditing work is controlled by assigning the secret inputs to an entity trusted third party, or worker, who is liable for performing computations and hand over the result of the computation to the cloud users or clients. To verify the integrity of computations using traditional cryptographic techniques, the time required to generate and validate the proof is a major computation issue. This paper proposes an improved public auditing technique for multi-party computation to check the integrity of outsourced data using a cryptographic solution. Many researchers have given auditing protocols that generate and verify proof using a cryptographic solution. Most of these scheme uses Non-Interactive Zero-Knowledge Proof (NIZK) which are basically built on bilinear map technology. The verification time using these existing technique is computationally expensive which affect the performance of the auditing system. We propose an efficient protocol that verifies the result correctness using modern cryptographic technique Indistinguishability Obfuscation. The proposed system works in two phases, (i) auction and (ii) audit. During the auction phase, multiple clients share their encrypted bid value to the worker. The worker generates auction result and proof using Pedersen Commitment Scheme. The audit phase starts only after the completion of the auction phase which results in reduced verification time. During the Audit phase, clients can verify the integrity of results using NIZK with the IO technique. The results for reduced verification time in auction system have been presented. It is found that the performance of the proposed system has improved compared to the pertinent NIZK Proof technique. In our setting, we assumed that a worker is one of the trusted entity. By this notion, our protocol also guarantees privacy to the clients during the audit phase.

Keywords: Cloud computing, Public auditing, Pedersen commitment scheme, Secure multiparty computation, Indistinguishability obfuscation.

1 Introduction

Cloud computing is an infrastructure in which multiple computing resources such as servers, services, networks, applications, and storage are aided on an on-demand basis [1]. These resources are rapidly assigned and released with the least interaction of IT management and service provider. Even though this structure is shown to be an exceptional service prototype for the Internet, it has brought several complex design problems that can affect the security and efficiency of the complete system [2]. Generally, it offers two primary service models, computing and cloud data store. Cloud computing services require up-to-date and efficient cloud data store to satisfy user requirements, all cloud computing services require high-performance cloud data store. Hence cloud data store is the most significant component icloud computing system. The files in cloud data store are retrieved and used by multiple users that may cause integrity and privacy issues. As cloud server is not fully trusted entity, certain unintended flaws and data issues might not be reported to cloud user to retain their status.

Consequently, to guarantee the precision of his farm out data on cloud data store, the user needs some means. Auditing is the method in which without downloading the entire data, the exactness of data or files can be confirmed. This auditing responsibility can be delegated to outside party for example Trusted Third Party (TTP). TTP will validate the correctness of cloud data store on the side of users.

Different public auditing techniques for cloud data store are discussed in [3]. These approaches focus on different techniques used to validate the exactness of out posted data on cloud data store. Multiple approaches are furnished by investigators [4,5] for verification of cloud data store. Many researchers have given cryptographic solutions for auditing which includes homomorphic authenticators, Elliptic Curve Cryptography (ECC), RSA etc.

In public auditing protocol, mostly TTP or worker is used to validate the correctness of farm out files or data in support of user. Since there is no necessity of public key management and verification, many researchers has given solutions for auditing using Identity-based (ID-based) cryptography. ID-based auditing [6] includes generation of file tags, verification of file tag or block tag, unforgeability and privacy preserving. ID-based ring signature is a consolidation of ID-based cryptography and common ring signature in which user's public key is signer identity. Vector commitment is one of the most significant commitment schemes in cryptography. It allows a user to bind to a selected value, safekeeping it from others. Subsequently that committed value can be opened or revealed to the verifier. This scheme fulfills two properties: hiding and binding. Auditing technique using ID-based cryptography and Vector commitment is proposed in [7]. With this method, the verification time increases as number of users emerge. Most of these existing schemes are insecure and consuming high computation and communication costs. So, another scheme suggested is an identity-based scheme using the RSA assumption which contains large prime exponents in random oracle model [8]. These ID-based schemes use bilinear pairing for auditing. Most of these schemes are having high computation and communication costs. Another approach for auditing is homomorphic encryption by which we can do computations over encrypted data though preserving the confidentiality of data. Another protocol proposes very effective outposting of data even though there exists a robust attacker [9]. This protocol uses fully homomorphic encryption scheme. It also retains the secrecy of the user's sensitive data. But the researchers of this scheme also suggested future work which must devise a verifiable and efficient computation scheme other than homomorphic encryption. Some researchers have proposed and enhanced the functionalities [10] of third party auditor servers such as public verifiability, metadata generation, data dynamics and encryption- decryption of data and files in cloud data store. An auditing technique that is based on provable data possession is proposed which enables users to efficiently depute the data validation process to a TTP [11]. This scheme proved that it has a low computational as well as communication burden on the cloud users during audit.

The most of the current research on cloud computing is primarily concentrating on the service side. But very few researchers have focused on the data security and trust. For the forthcoming growth of cloud computing technology in government sectors, and commercial trade, data security and privacy preservation are playing major roles. In some situations, data owners need to compute certain functions mutually from the collection of their sensitive data. For example, to predict cyber threats, companies may have to analyze the associated information from other companies, or hospitals may need to do medical research on their combined patient record. In such cases, result is obtained by performing computation on the data shared by all parties. But at the same time, each party want to preserve the privacy of their own sensitive information. Furthermore, even though we are using TTP to perform secure computation, many companies are not willing to share their sensitive information since TTP may breach the trust and misuse of this information. Secure Multi-party Computation (SMC) [12, 26] is a computationally efficient technique that enables a client to outsource computation to external parties or cloud providers, convincing that the client's sensitive or shared information is not misused even if some parties are corrupted or cannot be completely trusted. In most practical cases of secure multiparty computation such as auctions, e-voting, benchmarking services, or cloud services in general, computations are performed by assigning all the sensitive inputs to a TTP or worker. Worker is responsible for computation and distribution of the result to all users. Many researchers have given an effective private bidding and auction schemes [13, 16] where user and worker are doing the interaction among themselves in many rounds to complete the process of Multi-party Computation (MPC) protocol. The main focus of researchers is to minimize these rounds of interactions and communication complexity to complete MPC task.

The trust in SMC is one of the major problems which encompass two main issues: the integrity of the computation, and the secrecy of the inputs. To verify

Table 1. Comparative study for Verifiable SMC

Sr. No.	Reference Paper	Uses IO	Public Verification	Secure Multiparty Computation
1.	[14]	×	✓	✓
2.	[15]	×	✓	✓
3.	[16]	✓	×	✓
4.	[22]	✓	✓	×
5.	[23]	✓	✓	×
6.	[24]	✓	✓	×
7.	[25]	✓	✓	×

the correctness of computation, most of the researchers have given approaches using cryptographic primitives. One such approach is proposed which is an efficient MPC scheme with a public verifiability [14]. They have come up with a publicly verifiable, secure computation scheme. The scheme offers an improved version of the SPDZ (pronounced "Speedz") protocol. In this protocol, even though every party involved is dishonest, someone can verify about the correctness of the result. Edouard and Olivier [15] present verifiable Multiparty computation with Perfectly Private Audit Trail [PPAT] for MPC using Pedersen commitment scheme. This scheme is single-phase: the users share their inputs asynchronously, and afterward all parties can collect the result. They presented three distinct applications: resolving a system of linear equations, an auction system and the exploration of the shortest path in a shared graph. This protocol uses non-interactive zero-knowledge (NIZK) proof to check the correctness of result. The time required for verification is increasing as the numbers of users are increased. To reduce this time, in our proposed work, we are using a new modern cryptographic construct Indistinguishability Obfuscation (IO) [17-20] to check the correctness of result. Using IO, even we obfuscate two programs with the unique functionality, they are still computationally not distinct with each other. IO can be used with one-way function (OWF) to construct multiple essential building blocks such as public-key encryption, ID-based encryption, as well as Chosen-Ciphertext Secure Public Key Encryption and NIZKs. Amit and Brent [21] have shown different basic cryptographic paradigms from IO and one-way functions. Some of the researchers [22-25] explore IO for constructing a proof-of-retrievability technique which offers public verification for cloud data store, but very few researchers have addressed the public verification in secure multiparty computation. Table 1 shows the comparative study of different research work for verifiable SMC.

The contribution of our wok is as follows:

- Compared to the auditing scheme used in PPAT, our proposed auction scheme works in two phases, (i) Auction, and (ii) Audit. This will drastically reduce the verification time of auditing process.
- PPAT scheme uses NIZK to check the correctness of proof. In our proposed work during audit phase, we have implemented NIZK using modern cryptographic technique IO. Any user can check the bid values and results during audit phase.
- We proved by experiments that the verification time is reduced by our scheme compared to existing one.

The organization of remaining contents of this paper is described as follows. Section 2 elaborates related work with our scheme just as verifiable Multi-Party Computation (MPC), Indistinguishability Obfuscation and Pedersen Commitment Scheme. Pedersen commitment scheme plays very important role of generating proof in our scheme. In section 3, we explain our proposed verifiability scheme for SMC. The architecture, workflow and proposed algorithms are elaborated in this section. Section 4 compares the performance of results with existing NIZK [15] scheme.

2 Related work

2.1 Multi-party computation and audit

MPC is a technique in which several parties share their secret input without revealing anything to each other. The required function is calculated on shared input and result of computation is shared to all the parties. Assume there are two millionaires who want to find which of them has more money without sharing exactly how much money they have. MPC is useful in such situations. The two main paradigm of MPC protocol are [26]: circuit garbling and secret sharing. In circuit garbling, two entities garbler and evaluator are involved. A known function f (x, y) is securely computed by these parties using shared input. A Boolean circuit g=f(a,.) is encrypted by garbler which generates garbled truth table. It is send to evaluator. The evaluator decrypts the garbled circuit using corresponding key and check g(b). In secret sharing, inputs are shared among several parties and these shares are used to perform computation. Suppose we have specific field elements a $\in$ F, fragment it up into two pieces $a=a1+a2$. Give party p1 and p2 the value of a1 and a2 respectively. Neither party is knowing the value of a, but collectively they can calculate it. It means for each i, party p_i has a_i,

where, $a= \sum_{i=1}^{n} a_i$. Different sharing such as multiplicative $a=a1 \times a2$ is also available but additive scheme is mostly suitable for MPC applications. The basic overview of secret sharing MPC is as follows:

1. Initially, all parties share their secret inputs i.e. input 'x' is shared so that $x^i = \sum_{j=1}^{n} x_j^i$ and party P_j holds x_j^i where n is number of parties.
2. The parties carry out addition and multiplication on these shared values. Parties may communicate certain values by doing computation at local sites. The result of an operation is revealed to all the parties.
3. At the end, the parties 'Open' the result of computation. This stage comprises each party sharing their 'final' share to each other (verifies that no errors were hosted by the attacker during this communication).

For verifiable multiparty computation, Perfectly Private Audit Trail (PPAT) protocol was proposed in [15]. By this protocol, the users share their inputs to the worker. The worker can also be the user or separate entity who mostly concern with the generation of result and correctness proof. The output of computation and a publicly auditable proof is generated at the conclusion of the protocol. Even though worker is corrupted, this protocol ensures the integrity of the output. In PPAT process, different users n secretly share inputs I and calculate the function $O = f(I^n)$ and send the result O to all the users. The worker can be corrupted by an attacker. This protocol guarantees that even if worker send corrupted output, it will not be accepted by the users because worker has to generate a proof for the correctness of result. This proof is published by worker on Public Bulletin (PB) Board. By verifying this proof, every user can ensure the integrity of the received result. To build the proof, protocol uses Pedersen Commitment scheme and NIZK is used to verify the proof.

2.2 Indistinguishability obfuscation

Barak et.al [20] attracted the attention of researchers on the conception of program obfuscation in 2001. By his theory, program obfuscation is a method which create computer programs "unintelligible" however maintaining their functionality. According to Amit and Brent in [21], they proposed two ways of general obfuscation, i) Virtual black-box obfuscation (VBO) and ii) Indistinguishability Obfuscation. VBO consist of obfuscated program which is nothing but a black box instantiating the program. Perhaps this concept has several immediate and native applications in cryptography, such as, to convert a normal private-key encryption scheme to a public-key encryption scheme [20]. But still, they proved that VBO is not possible to accomplish. Based on this impracticality, they have proposed second concept known as Indistinguishability Obfuscation, which is possibly feasible to implement. This concept states that even though we obfuscate any two different (same-size) functions that implement unique functionalities, they are still computationally not differentiable with respect to each other.

How to obfuscate the program is a major issue. The idea is to puncture the program (which is to be obfuscated). The concept states that we have to separate a key part of the program in such a manner that the functionality of the program still has to remains the same. And since we remove a key element, attacker can't win the security game. Suppose we want to transform a natural private key encryption scheme into a public-key encryption scheme by process of obfuscation. Consider simple private-key encryption scheme: Assume that the key K is a secret key of a pseudo-random function (PRF), r is a random string. We can represent PRF as $PR=PRF(K, r)$. For private-key encryption scheme, using a random string r, encrypt the message m and generate the encoded text $C = (r, PR \oplus m)$.

How to convert this private-key encryption scheme into public-key encryption scheme using IO? With respect to security point of view, we want to achieve here is that attacker should not be capable to regain message m*, if given challenge ciphertext $C^*= (r^*,e^*)$, encoding certain message m*. To achieve this, consider flawed solution as: Let's assume that obfuscated version of encryption function as a public key. So obfuscated encryption function becomes: $f_k(r, m) = (r, PR \oplus m)$. So we have to develop the function $f_k$using a "Punctured" PRF key such that it properly define the PRF at all other random strings than r, but not going to reveal any information about $PRF(K, r^*)$. Then it is not possible for attacker to crack the security of challenge cipher-text even though he is aware of this punctured PRF confidential key. It means we have to substitute original function $f_k(r, m) = (r, PR \oplus m)$ using obfuscation of punctured function in such a way that it is not differentiable to the attacker. Still, there is a trivial attack possible on this solution. Attacker simply feed input $(r^*, 0)$ to the obfuscated program and accordingly determine $PRF(K,r^*)$ and regain m^* from the challenge cipher text. So the idea is to search an alternative to "shift" the puncturing to a place which is not at all retrieved by the program functionally.

Using a Pseudo-Random Generator (PRG) which associates λ bits to 2λ bits, where λ is a security parameter, we can tackle this issue. We can rewrite new PRF using PRG as *N_PRF= (k, PRG(r))*. So revised private-key encryption function is, $f_k(r, m)$ = *(PRG(r), N_PRF* $\oplus$ *m)*. Because of keyless nature of PRG, the attacker can't differentiate the original security game in which the challenge cipher text was generated.

According to Tal and Alone [27], IO is one of the weaker primitive than VBO. Therefore, it is not possible to construct different cryptographic constructs using only IO. Amit and Brent [21] has given different cryptographic building blocks such as public-key encryption, short signature, NIZK etc. using IO and one-way function. In our proposed system, NIZK using IO is used to verify the correctness of result.

2.3 Commitment consistent encryption (CCE) scheme

This scheme uses both encryption and a commitment scheme. In this scheme, user share his input secretly through a cipher text while further verification is done by committing publicly on the equivalent input. It is feasible to generate a commitment on the encrypted message from any CCE ciphertext, and this commitment can be opened by the private key. A CCE is formed by different algorithms such as Gen, Enc, Dec, DerivCom, Open, Verify defined as below:

- Gen (1^{λ}): **Generates different security parameters.** Using a security parameter λ, generate parameters pp, pk and sk, where pp is public parameter, pk is public key and sk is the secret key.
- Enc_{pk} (m): **Encrypts the message.** By encrypting message m (selected from the plaintext **M** demarcated by pp) using public key pk, generate a ciphertext c.
- Dec_{sk}(c): **Decrypts the message.** Decrypt ciphertext c and generate a message m utilizing the secret key sk.
- $DerivCom_{pk}$ (c): **Computes the Commitment Value.** Using ciphertext c and pk, generate a commitment d.
- $Open_{sk}$(c): **Generates Opening Value for Commitment.** Generate an opening term o utilizing the secret key sk. The parameters of this term is used to open a commitment.
- $Verify_{pk}$(d, o, m): **Verify Commitment.** Verification result either success or fail, is generated inputting a message m, a commitment d and an opening term o with respect to key pk. It checks verifies the opening (o, m) related to pk and d.

We can use different commitment schemes to generate and verify the commitment in CCE encryption schemes such as Pedersen, vector etc. In our proposed work, we are generating and verifying the commitment using Pedersen technique which is explained in next section.

2.4 Pedersen commitment scheme

Strong Commitment schemes must have to possess two properties: information hiding and computationally binding. The Pedersen Commitment scheme permits a sending party to generate a commitment using a secret value. The verifier may then afterward expose the commitment. The value is disclosed in a verifiable way that binds them to their commitment. A Pedersen Commitment scheme consists of a three stages, (i) Setup, (ii) Commit, and (iii) Reveal.

Setup: The Receiver selects

- Large primes p and q in such a way that q divides p-1
- G is generator of the order-q subgroup of Z_p^*
- Random secret a from Z_q
- Calculate $h = g^a \ mod \ p$
  -values p, q, g, h are published publicly, a kept private.

Commit:

To commit to some $x \in Z_q$, sender calculates $C = g^x \ h^r mod \ p$ where r is random value such that r $\in$ Z_q. Sender sends this Commitment value C to the receiver. This is nothing but same as $g^x \ (g^a)^r \ mod \ p = g^{x+ar} mod \ p$

Reveal:

Sender exposes x and r to open the commitment, receiving party then validates that $C= g^x h^r \ mod \ p$

3 Proposed system

In this part, we first give the construction of PPAT algorithm for electronic auction. Then we show the modification in this protocol using IO. MPC arrangement for different entities is as shown in Fig. 1. It comprises of multiple clients. Each client C_i has his private input $x_i \in$ I, where i=1,2,…,n. The worker is a node who can also be a client or separate entity often addresses the correctness aspect through zero-knowledge proof. Public Bulletin Board PB is a place where worker publishes proof of the correctness of the output. Each client can verify it at any time. Each client submit one bid to worker node

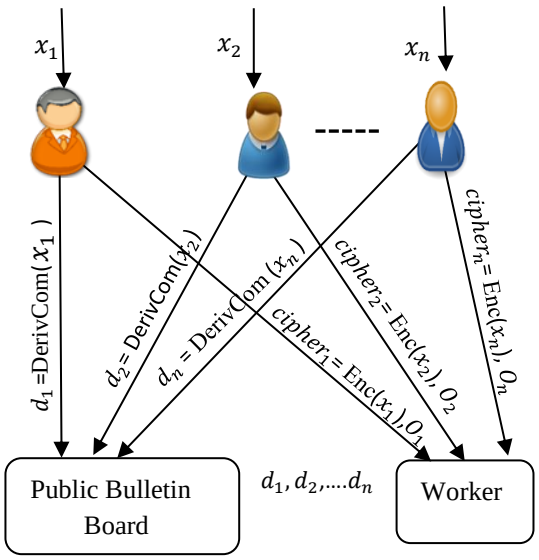

Figure. 1 Architecture of proposed system

W. The result for the auction system is nothing but a list of sorted bids. Each Client computes $c_i \leftarrow Enc\ (x_i)$. From c_i , C_i derives $d_i \leftarrow DerivCom\ (c_i)$ and computes $\pi_{ran}(d_i,$ I) as in Eq. (1). where π_{ran} is a proof which consists of relation as follows.

$$R_{ran}=\{(d, (x, O))|Verify(d, x, O) =1 \wedge x \in I\} \quad (1)$$

where O is an opening term to expose the commitment d. While c_i is sent to W, each C_i publishes d_i and π_{ran} (d_i, I) on PB. W computes the sorted list (x_1' ...x_n') from ($x_1 \ldots x_n$) using any known sorting algorithm. W reshuffles d_i from the sorted list, to generate ordered list of commitments $d_1' \ldots d_n'$. Later W calculates n-1 commitments $e_1 \ldots e_{n-1}$ where $e_i = d_i \geq d_{i+1}$ which requires n-1 proofs. Thus π_{cor} is a combined proof based on sorted commitments as in Eq. (2).

$$\pi_{cor} = ((e_1,O_1 , \pi_1) \wedge \ldots \wedge (e_{n-1},O_{n-1} , \pi_{n-1})) \quad (2)$$

W publishes π_{cor} on PB along with $d_1' \ldots d_n'$. Then each client validates π_{cor} to verify the result of the auction. Fig. 2 shows the workflow between client and worker.

The proposed system works in two phases auction and audit phase. Both the phases are independent of each other. Algorithm1 shows the auction phase of proposed system. In auction phase, different users bid for their values. Worker node generates winning bid using any suitable sorting algorithm. It also generates proof using Pedersen Commitment Scheme and publishes it on PB.

After auction phase, audit phase will start where clients can verify the bid. In PPAT protocol, auction

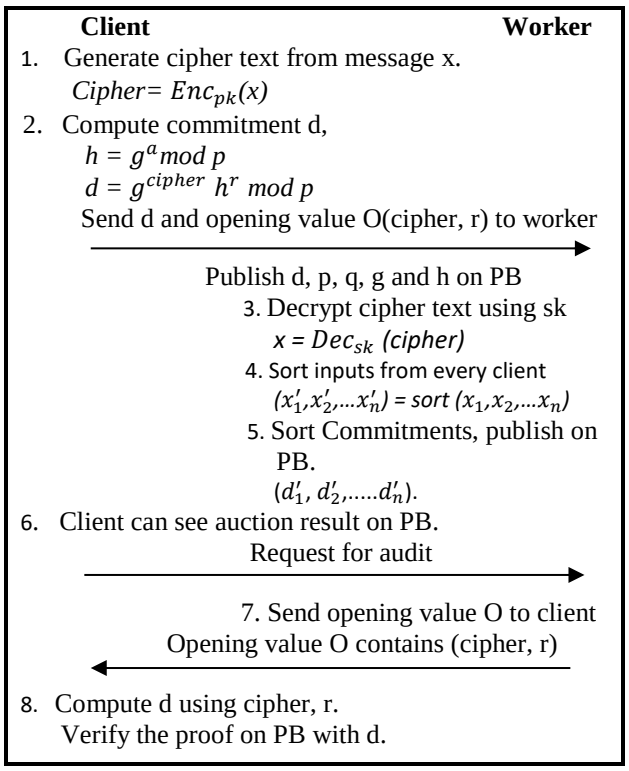

Figure. 2 Workflow of auction/audit phase

system uses NIZK which is mostly built on bilinear groups. In the proposed system, we have modified PPAT protocol for auction system which uses NIZK using modern cryptographic construct Indistinguishability Obfuscation. In this proposed NIZK scheme, an obfuscated function is having parameters instance and witness and verify the instance if the witness relation holds. To produce a proof π, using (authentic) witness w_0 , a prover initially builds a NIZK proof of statement x. Afterward, a prover can assert that π was evidence generated utilizing witness $w_1 \neq w_0$ even though he is not aware of the witness w_1 when he initially generated the proof. Algorithm 2 shows the original audit program with witness w_0 . In audit phase, opening value O is the input comprises of encrypted bid value given by each client and a random value r. Using O and r, verifier calculates d. It will be compared with d value of PB. If a relation holds, audit program generates the accept result otherwise rejected.

4 Results and discussions

Now we give the performance analysis of our proposed auditing scheme by comparing it with auditing scheme used in PPAT. The implementation of this system is realized in Python. The main components of the system are auction-goers or clients, Cloud Service Provider(CSP) and Worker. Clent entity implemented as Android app from where users

Algorithm1: Auction Phase

Input: Message x,Public Key pk and Secret Key sk,
Large p & q in such a way that q divides p-1.
g is generator of the order-q
Random Secret a from Z_q
C: Set of Clients (1, 2,...n)
W: Worker Node
PB: Public Bulletin Board

Output: Proofs generated by client and worker node

1. For each Client C_i, $i \leftarrow 1$ *to* n
   - Generate cipher by encrypting message x.
     $cipher = Enc_{pk}(x)$
   - Calculate h = g^a mod p
   - Generate commit d using random value r, $d = g^{cipher}\ h^r\ mod\ p$
   - Send d and opening value O to the worker node.
     Where O is a tuple (cipher, r).
   - Publish d on PB.
2. Make g,p,q and h as public.
3. W decrypt ciphertex using his own secret key sk
   $x = Dec_{sk}(cipher)$
4. W sorts the input of every client using any suitable sorting algorithm.
   $(x'_1, x'_2, ...x'_n) = sort\ (x_1, x_2, ...x_n)$
5. W rearranges d_i to generate a sorted list of commitments (d'_1, d'_2,.....d'_n).

Algorithm2: Audit Phase

Input: Opening Value O is a tuple (cipher, r)
Output: Successful/Unsuccessful verification

1. Test if $d = g^{cipher}\ h^r\ mod\ p$
2. Outputs accept if true, reject if false.

can bid for the auction. Auction values get stored on CSP using Firebase Cloud Server. Worker is implemented on system having Windows 8.1with an Intel Core i5-5200U CPU functioning at 2.20 GHz, 2201 Mhz, 4.0 GB RAM, 2 Cores and 4 Logical Processors. The elliptic curve which we have used is having its base size as 256 bits and its embedding degree 6.

We have implemented auditing system for auction. Auction and audit phases work independent with each other. Audit phase will start after the completion of auction phase only. Different users can give their bid values using android app. For obtaining the results, bid values are stored on Firebase server. The results were obtained up to 1000 users. The

Algorithm3: Audit Phase* (Obfuscated)

Input: Opening Value O is a tuple ($cipher^*$, r)
Output: Successful/Unsuccessful verification

1. If $cipher = cipher^*$,
   test if $g^{cipher^*}\ h^r\ mod\ p = d^*$,
   Outputs accept if true, reject if false.
2. Test if $d = d^*$
3. Outputs accept if true, reject if false

Table 2. Verification time for auction system

Number of Clients	Verification time for client(in sec) using NIZK scheme in PPAT[15]	Verification time for client (in sec) using proposed system
5	0.236	0.0591
10	0.394	0.0917
50	1.973	0.1146
100	4.17	0.171
500	19.364	0.3164
1000	42.08	4.025

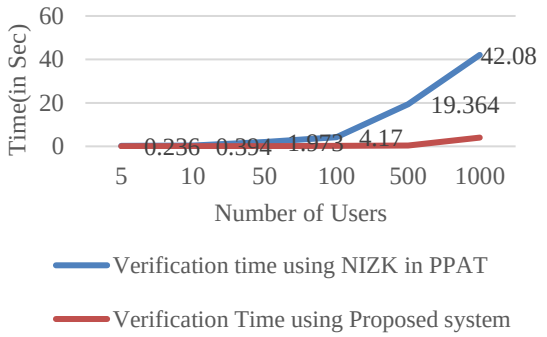

Figure. 3 Performance of proposed NIZK system

parameter we have chosen is verification time required for user to check the correctness of result.

In proposed system, integrity of data is exploited by hacker entity that is having capability to modify the inputs and result of auction. If values are modified, our auditing system will correctly generate verification unsuccessful message to verifier. Table 2 shows the verification time in seconds for the proposed NIZK system using IO compared with NIZK protocol used in PPAT algorithm for auction system.

According to Table 2, we observe that the verification time for our proposed NIZK using IO is reduced as compared to NIZK scheme used in PPAT for different users. We have not considered the size of input for this system while calculating the result as every client submits only the bidding value which is mostly very small in size. Fig. 3 shows the graphical

representation of performance of our proposed system. As we can observe from the graph, the verification time of proposed NIZK system is constant compared to NIZK in PPAT. In PPAT, as number of users are increased, verification time also increases.

5 Conclusion

This paper proposed a secure and verifiable auditing scheme for secure multiparty computation. We have implemented the verifiable auction system to carry out our experiments. Users can validate the correctness of the bidding value and bidding result using auditing technique. To verify the correctness, PPAT scheme uses NIZK, which results in increased verification time for user. Our implementation uses modern cryptographic technique such as Indistinguishability Obfuscation. Even though IO is one of the weaker primitive and not possible to have real life implementation, combined with OWF, can create useful cryptographic construct. By using these constructs, it is possible to implement IO for different applications. This paper also shows such an application such as verifiable auction system using IO. Result show that using our proposed verification scheme, for 100 users, verification time has reduced from 4 seconds to 0.171 seconds. As number of users are increased in PPAT scheme, verification time is also increased. Our proposed system shows constant verification time even though users are increased. The constant verification time is the result of two different phases: i) auction and ii) audit in our proposed scheme. Audit phase is completely independent of auction phase. Because of this, the verification time for our scheme is drastically reduced as compared with NIZK scheme used in PPAT protocol. Moreover, scheme is secure since the encrypted bidding values from each user are used in auditing. By assuming worker as one of the trusted entity, our scheme also maintains privacy. But if worker take advantage of this trust, with the help of any client he can create collusion attack. This may create privacy issues in auditing system. So our future work is to address collusion attack and come up with efficient and better auditing scheme which maintain privacy also.

Conflicts of Interest

The authors state that there are no conflicts of interest about the publication of this paper.

Author Contributions

Conceptualization, Smita and Gandharba; methodology, Smita; software, Smita; validation, Smita, Gandharba and Pragnyaban; formal analysis, Smita; investigation, Smita; resources, Smita, Gandharba and Pragnyaban; data curation, Smita, Gandharba and Pragnyaban; writing—original draft preparation, Smita; writing—review and editing, Smita, Gandharba and Pragnyaban; visualization, Smita; supervision, Smita; project administration, Smita; funding acquisition, Smita, Gandharba and Pragnyaban.

Acknowledgments

This research work is an independent work and no financial assistance has been received for this work.

References

[1] C. Wang, K. Ren, W. Lou, and J. Li, "Toward Publicly Auditable Secure Cloud Data Storage Services", *IEEE Nerwork*, Vol. 24, No. 4, pp. 19-24, 2010.

[2] T. K. Babu and C. D. Guruprakash, "A Systematic Review of the Third Party Auditing in Cloud Security: Security Analysis, Computation Overhead and Performance Evaluation", In*: Proc. of 3rd International Conf. on Computing Methodologies and Communication (ICCMC)*, pp. 86-91, 2019.

[3] S. Chaudhari and S. K. Pathuri, "A Comprehensive Survey on Public Auditing for Secure Cloud data store", *International Journal of Engineering & Technology*, Vol. 7, No.27, pp. 565-569, 2018.

[4] S. Hohenberger, V. Koppula, and B. Waters. "Universal signature aggregators", In: *Proc. of Annual International Conf. on the Theory and Applications of Cryptographic Techniques*, Berlin, pp. 3-34, 2015.

[5] S. Rizvi, K. Karpinski, B. Kelly, and T. Walker, "Utilizing third party auditing to manage trust in the cloud", In: *Proc. of Complex Adaptive System*, San Jose, CA, pp. 191-197, 2015.

[6] J. Hong, M. Xie, B. Kang, C. Li, and L. Si, "ID-Based Public Auditing Protocol for Cloud data store Data Integrity Checking with Strengthened Authentication and Security", *Wuhan University Journal of Natural Sciences*, Vol. 23, No. 4, pp. 362-368, 2018.

[7] S. Singh and S. Thokchom. "Public integrity auditing for shared dynamic cloud data" , In: *Proc. Of 6th International Conf. on Smart*

Computing and Communications, ICSCC, Kurukshetra, India, pp. 698-708, 2018.

[8] Z. Xu, L. Wu, M. K. Khan, K. R. Choo, and D. He, "A secure and efficient public auditing scheme using RSA algorithm for cloud data store", *The Journal of Supercomputing*, Vol. 73, No. 12, pp. 5285-5309, 2017.

[9] R. Gennaro, C. Gentry, and B. Parno, "Non-interactive verifiable computing: Outsourcing computation to untrusted workers", In: *Proc. of Annual Cryptology Conf.* Berlin, pp. 465-482, 2010.

[10] N. Nagar and S. Ugrasen, "Reliable and Enhanced Third Party Auditing in Cloud Server Data Storage", *International Journal of Security and Its Applications*, Vol. 11, No. 7, pp. 59-72, 2017.

[11] L. E. Ghoubach, R. B. Abbou, and F. Mrabti, "A secure and efficient remote data auditing scheme for cloud data store", *Journal of King Saud University-Computer and Information Sciences*, 2019. https://doi.org/10.1016/j.jksuci.2019.02.011

[12] F. He and T. Wang, "Research and application of Secure Multiparty Computation in Several Computational Geometry Problems", In: *Proc. of International Conf. on Industrial Control and Electronics Engineering(ICICEE)*, pp. 1434-1437, 2012

[13] C. Cachin, "Efficient private bidding and auctions with an oblivious third party", In: *Proc. of the 6th ACM Conf. on Computer and Communications Security*, Singapore, pp. 120-127, 1999.

[14] C. Baum, I. Damgård, and C. Orlandi, "Publicly auditable secure multi-party computation", In: *Proc. of International Conf. on Security and Cryptography for Networks*, pp. 175-196, 2014.

[15] E. Cuvelier and O. Pereira, "Verifiable Multi-party Computation with Perfectly Private Audit Trail", In: *Proc. of International Conf. on Applied Cryptography and Network Security*, Guildfold, UK, pp. 367-385, 2016.

[16] S. Garg, C. Gentry, S. Halevi, and M. Raykova, "Two-round secure MPC from indistinguishability obfuscation", In: *Proc. of Theory of Cryptography Conf.* Berlin, pp. 74-94, 2014.

[17] S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, and B. Waters, "Candidate indistinguishability obfuscation and functional encryption for all circuits", *SIAM Journal on Computing*, Vol. 45, No. 3, pp. 882-929, 2016.

[18] S. Banescu, M. Ochoa, N. Kunze, and A. Pretschner, "Idea: benchmarking indistinguishability obfuscation–a candidate implementation", In: *Proc. of International Symposium on Engineering Secure Software and Systems*, pp. 149-156, 2015.

[19] Y. Shi, H. Fan and Q. Liu, "An Obfuscatable Designated Verifier Signature Scheme", *IEEE Transactions on Emerging Topics in Computing*, Vol.5, No.2, pp. 271-285, 2017.

[20] B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, S. Vadhan, and K. Yang, "On the (im) possibility of obfuscating programs", In: *Proc. of Annual International Cryptology Conf.*, Berlin, pp. 1-18, 2001.

[21] A. Sahai, and B. Waters, "How to use indistinguishability obfuscation: deniable encryption, and more", In: *Proc. of the Forty-Sixth Annual ACM Symposium on Theory of Computing*, pp. 475-484, 2014.

[22] C. Guan, K. Ren, F. Zhang, F. Kerschbaum, and J. Yu, "Symmetric-key based proofs of retrievability supporting public verification", In: *European Symposium on Research in Computer Security*, Vienna, Austria, pp. 203-223, 2015.

[23] M. Liu, Y. Wu, J. Chang, R. Xue, and W. Guo, "Verifiable proxy re-encryption from indistinguishability obfuscation", In: *Proc. of International Conf. on Information and Communications Security*, pp.363-378, 2015.

[24] Y. Zhang, C. Xu, X. Liang, H. Li, Y. Mu, and X. Zhang, "Efficient public verification of data integrity for cloud data store systems from indistinguishability obfuscatio", *IEEE Transactions on Information Forensics and Security*, Vol. 12, No. 3, pp. 676-688, 2016.

[25] S. Kumawat and S. Paul, "A new constant-size accountable ring signature scheme without random oracles", In: *Proc. of International Conf. on Information Security and Cryptology*, pp. 157-179, 2017.

[26] P. Pullonen and S. Siim, "Combining Secret Sharing and Garbled Circuits for Efficient Private IEEE 754 Floating-Point Computations", In: *Proc. of International Conf. on Financial Cryptography and Data Security*", Berlin, pp. 172-183, 2015.

[27] T. Moran and A. Rosen, "There is no Indistinguishability Obfuscation in Pessiland", *IACR Cryptology Eprint Archieve*, p. 643, 2013.