Block-SMPC: A Blockchain-based Secure Multi-party Computation for Privacy-Protected Data Sharing

Yuhan Yang
Department of Automation, Shanghai Jiao Tong University and the Key Laboratory of System Control and Information Processing, Ministry of Education of China, Shanghai 200240, China
yuhanyang@sjtu.edu.cn

Lijun Wei
Department of Automation, Shanghai Jiao Tong University and the Key Laboratory of System Control and Information Processing, Ministry of Education of China, Shanghai 200240, China
sjtu_weilijun@sjtu.edu.cn

Jing Wu
Department of Automation, Shanghai Jiao Tong University and the Key Laboratory of System Control and Information Processing, Ministry of Education of China, Shanghai 200240, China
jingwu@sjtu.edu.cn

Chengnian Long*
Department of Automation, Shanghai Jiao Tong University and the Key Laboratory of System Control and Information Processing, Ministry of Education of China, Shanghai 200240, China
longcn@sjtu.edu.cn

ABSTRACT
With the rapid development of Internet of Things, the demand of data sharing is not only to ensure the data integrity, but also to protect user's individual privacy. Secure multi-party computation, as a technology paradigm based on cryptography technology, enables privacy protection in data sharing among multiple parties, while the implement of secure multi-party computation is limited due to the inefficient, complex computation protocol and frequent interaction. Fortunately, the emergence of blockchain technology provides excellent properties of decentralization, verifiability and high reliability for secure multi-party computation. In this paper, we propose a blockchain-based secure multi-party computation architecture for data sharing, where we design an aggregator consortium for data storage, verification and joint computation. Moreover, we introduce a straightforward secure multi-party computation scheme based on homomorphic encryption. In addition, we analyze and discuss this scheme in terms of data security, access flexibility, attack resistance and potential applications.

CCS Concepts
•Security and privacy →Systems security →Distributed systems security

Keywords
Privacy protection; Blockchain; Secure multi-party computation; Homomorphic encryption

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Permissions@acm.org.
ICBCT'20, March 12–14, 2020, Hilo, HI, USA
© 2020 Association for Computing Machinery.
ACM ISBN 978-1-4503-7767-6/20/03…$15.00
https://doi.org/10.1145/3390566.3391664

1. INTRODUCTION
Privacy protection in data sharing has been spotlighted as a critical problem in most industrial smart applications [1]. Especially with the rapid growth of the number of devices in the Internet of Things (IoT) system, the terminal devices collect a huge amount of data in the edge network. Privacy protection is a fundamental issue when sharing these data for intelligent analysis and computation, which receives great attention in industry and academia. Secure multi-party computation (SMPC) [2], presents a novel solution for multi-party data sharing while protecting user's private information. It is proposed by Yao in 1982, who solved the typical case "The Millionaires' Problem" using the garbled circuit method. SMPC provides opportunity for multiple parties to cooperate and share data, and ensures the confidentiality of individual privacy in a public channel by leveraging the cryptography. However, despite these considerable advantages, when using SMPC for data sharing, it is limited due to: (i) in traditional SMPC architecture, data from diverse participants needs to be aggregated for analysis and filtration, which causes the risk of the single point of failure; (ii) most SMPC scheme is inefficient because of the frequent interaction among multiple participants; (iii) some verifiable data transmission methods might be embedded in the encryption schemes to guarantee the correctness and integrity in data sharing, which makes the SMPC scheme more complex; (iv) some SMPC protocol may require the data providers to keep online for maintaining the protocol execution, while these data providers are usually high mobility devices with unstable network conditions and cannot stay online throughout.

For the problems above, blockchain appears to be a novel and feasible solution by providing a tamper-resistant ledger for reliable data storage. In order to avoid single point of failure and ensure data integrity, some researchers combine blockchain technology with SMPC, in which blockchain is regarded as a medium to store and share data in a public distributed channel [3]. Moreover, a number works focus on promoting the cooperation among multi-parties and designing incentive mechanism based on

smart contract [4], [5]. Despite that the current works solve problems of single point of failure, data security and cooperative incentive in SMPC to some extent by introducing blockchain technology, the complexity of SMPC scheme and operation burden of users are not be considered. In practical scenario, the feasibility of SMPC architecture and the effectiveness of computation protocol are also need to be guaranteed.

In this paper, we propose Block-SMPC, which combines blockchain technology and SMPC to achieve efficient and feasible privacy-protected data sharing, while providing flexible access for users. According to the essential roles in SMPC (i.e. data provider, data aggregator and computing unit), we design a dynamic aggregator consortium for reliable data storage and electing one aggregator to cooperate with the computing unit when performing multi-party computation. In addition, we propose a homomorphic encryption-based SMPC scheme by following the encryption mechanism in [6], which decreases the interaction frequency between users and other components, and ensures data confidentiality through the authority separation of homomorphic key and ciphertext.

2. RELATED WORK

To address the data security and improve the efficiency of SMPC, most works design SMPC protocol for data sharing based on cryptography such as garbled circuit [7], [8], threshold secret sharing and homomorphic cryptosystem [6], [9], [10], [11], [12]. To be specially, Kamara *et al.* [7] considered to outsource the complex computation tasks of participants to the cloud server based on Yao's garbled circuit, and designed server-aided MPC protocols to ensure the efficiency in terms of computation and communication. Carter *et al.* [8] solved the problem of outsourced secure function evaluation (SFE) by proposing a new SFE protocol that allowed mobile devices to securely outsource the majority of computation and ensure the security in malicious adversary attacks. Asharov *et al.* [9] constructed a SMPC protocol to against full malicious attackers using threshold fully homomorphic encryption (TFHE), and showed a relatively efficient fully homomorphic key construction scheme.

In order to ensure data verifiability and provide multi-party access control, Baum *et al.* [10] gave the concept of publicly auditable secure computation and proposed a protocol to enhance the so-called SPDZ protocol. Maltitz *et al.* [11] proposed fine-granular access control on data querying. Moreover, to reduce interaction frequency and the complexity of encryption algorithm, L'opea-Alt *et al.* [12]introduced an encryption scheme based on multikey fully homomorphic encryption, which allows users to hold long-term public-secret key pairs. Peter *et al.* [6] leveraged two non-colluding servers to design a multi-party computation protocol under simple interaction.

Similar to our work, some researches combined blockchain technology with SMPC for multi-party data sharing and computation [4], [3], [5], [13]. An early work to implement multi-party computation with Bitcoin is in [13], which showed the properties of Bitcoin that can be used in SMPC. Kumaresan *et al.* [4] concentrated on the fairness of multi-party computation protocol and proposed the incentive model based on Bitcoin to promote the cooperation between participants. Furthermore, Zhou *et al.* [3] developed blockchain-based IoT system "BeeKeeper" and achieved secure storage through threshold secret sharing. Gao *et al.* [5] designed an open reputation system based on blockchain, where all parties are considered as foresighted and non-cooperative players in a game and work under a certain incentive mechanism. These works only focus on the fairness of SMPC protocol or cooperative incentive among participants, however, the efficiency and feasibility are failed to be considered. Specially, we propose a blockchain-based SMPC architecture according to the fundamental roles in SMPC, where multiple data aggregators form an aggregator consortium to achieve efficient, secure and verifiable data sharing. Moreover, we introduce a homomorphic encryption-based SMPC mechanism to enhance the feasibility of SMPC.

3. BLOCKCHAIN-BASED SMPC ARCHITECTURE

In this section, we introduce our proposed blockchain-based SMPC architecture as shown in Figure 1, which contains two networks (i.e., the blockchain network and the SMPC network) and three parties: an aggregator consortium, a computing server and users. The blockchain network consists of the aggregator consortium and users. The SMPC network is constituted by the aggregator consortium and the computing server. Blockchain, as a public ledger, stores all the secret data of users and is maintained by the aggregator consortium. Considering a specific application scenario of electronic election, where each voter (user) encrypted the individual voting result and upload it to blockchain. The aggregator and computing server jointly perform the statistical process, then the final vote of each candidate will be disclosed to the blockchain network. Now we present detailed introduction of each component in our proposed architecture.

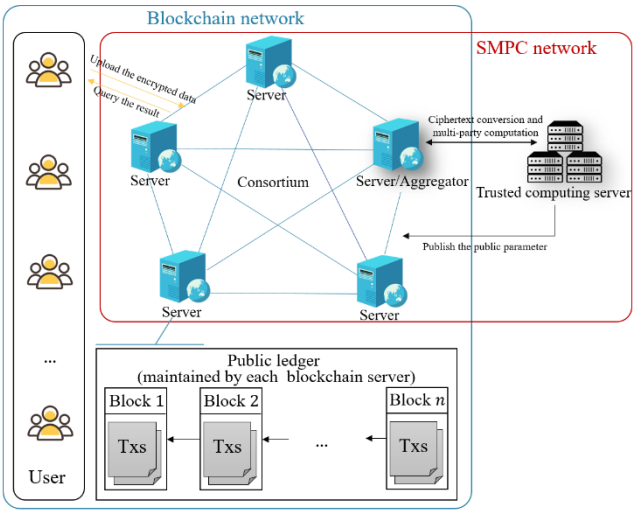

Figure 1. Blockchain-based SMPC framework

3.1 Users

The users are lightweight nodes in the blockchain network and don't synchronize the complete blockchain. As the data providers and SMPC requesters, the users encrypt the data locally

and upload the ciphertext to blockchain. However, in practical application scenario, users cannot promise to participate in the SMPC protocol execution throughout, and they may misbehave when submitting data to the aggregator. Therefore, it is necessary to decrease the interaction among users and other components, and the SMPC protocol should allow them to resubmit data at any time.

3.2 Aggregator Consortium

The aggregator consortium is composed by a certain number of servers with abundant resource, in which each server is regarded as a data aggregator to collect users' secret data. Besides, it is responsible for synchronizing and maintaining the complete blockchain. When performing multi-party computation, the selected aggregator downloads the necessary data from blockchain and executes the SMPC protocol by cooperating with the computing server.

3.3 Computing Server

Most homomorphic cryptosystems usually require a pair of public/secret key for encryption. To this end, following the encryption mechanism in [6], we deploy a computing server to convert the ciphertexts encrypted under different secret keys to the ciphertexts encrypted by a uniform key. Moreover, the computing server separates the management authority of homomorphic key and ciphertext with data aggregator for data confidentiality.

4. HOMOMORPHIC ENCRYPTION-BASED SMPC SCHEME

In this section, we introduce the homomorphic encryption method and the SMPC mechanism based on homomorphic encryption.

4.1 Homomorphic Encryption

Homomorphism is a pivotal property of the encryption mechanism in our SMPC scheme, which ensures the computing result performed over the ciphertext is equal to the result performed over the plaintext. To be specific, the additive homomorphism is guaranteed if for two ciphertexts c_1, c_2, we have

$$En_{pk}(c_1) \cdot En_{pk}(c_2) = En_{pk}(c_1 + c_1), \tag{1}$$

where $\cdot$ is an operation in the encryption domain, and $En_{pk}(c_1)$ represents that the ciphertext c_1 is encrypted by the public key pk. ElGamal [14] is a typical homomorphic encryption system, it satisfies the multiplication homomorphism and guarantees the additive homomorphism through modifying the encryption and decryption algorithm. In our work, we follow [15] and introduce an additively homomorphic encryption scheme, which is a variant of ElGamal cryptosystem.

Initialization. Generate a public parameter triplet (N, k, g). N is a safe-prime modulus of bitlength k, and it can be calculated by $N = pq$, where p and q are primes that can be given by $p = 2q' + 1$ and $q = 2p' + 1$. Note that p' and q' are two primes. Next, we pick $g \in \mathbb{Z}_{N^2}^*$ as the generator of $\mathbb{G}$, which is a cyclic group of quadratic residues modulo N_2. If a participant in the cryptosystem generates its own secret key as $sk = a$, the public key can be given as

$$pk = g^a \mod N^2, \tag{2}$$

which depends on the public parameter triplet (N, k, g). Moreover, we define a master key $MK = (p', q')$ for decrypting all ciphertext encrypted by any key, and it is held by the computing server secretly.

Encryption. Given a plaintext message m and let $\mathbb{Z}_N$ be the message space, we randomly choose $r \in \mathbb{Z}_N^2$. The ciphertext defined by (A, B) can be calculated as

$$A = g^r \mod N^2, \tag{3}$$

$$B = h^r(1 + mN) \mod N^2.$$

Decryption by sk. The ciphertext (A, B) encrypted by public key pk can be decrypted by the corresponding secret key sk as

$$m = (B/A^a - 1 \mod N^2)/N. \tag{4}$$

Decryption by MK. Ciphertext encrypted by any public or secret key can be decrypted by the master key MK. Firstly, we calculate $a \bmod N$ as

$$a \mod N = \frac{h^{p'q'} - 1 \mod N^2}{N} \cdot k^{-1} \mod N. \tag{5}$$

Similarly, $r \bmod N$ is shown as follows

$$r \mod N = \frac{A^{p'q'} - 1 \mod N^2}{N} \cdot k^{-1} \mod N. \tag{6}$$

Then the plaintext message can be decrypted by

$$m = \frac{(B/(g^\gamma))^{p'q'} - 1 \mod N^2}{N} \cdot \epsilon \mod N, \tag{7}$$

where γ is the inverse of $p'q'$ modulo N and $\epsilon = ar \bmod N$.

4.2 Homomorphic Encryption-based SMPC Mechanism

Based on the proposed blockchain-based SMPC architecture, we present our homomorphic encryption-based SMPC mechanism in detail. As illustrated in Figure 2, we divide the mechanism into five parts: *data collection and verification, data blinding, ciphertext converting, multi-party computation and result return.*

- **Data collection and verification.** In order to share data secretly, we recall the homomorphic encryption algorithm in the section 4.1. The computing server generates the public parameter triple (N, k, g) and discloses it to the blockchain network. Each user queries the aggregator to obtain the public parameter, and then generates its own public/secret key pair. Considering a scenario that includes n users $U = (1,2, \dots, n)$, in which each one generates its key pairs as (pk_i, sk_i), where $i \in U$. Each user encrypts the shared data using the public key and submits it to the aggregator as a transaction. The aggregators are responsible for verifying all submitted transactions and ensuring the transactions are well-formed and valid. Each valid transaction will be recorded in blockchain by the aggregators who synchronize the complete blockchain after a certain consensus process.
- **Data blinding.** In order to prevent the computing server from obtaining and decrypting the secret data, we design the data blinding process. Specifically, an aggregator will be elected in the aggregator consortium to download the necessary data from blockchain and blind it before performing multi-party computation. We define by $M = (m_1, m_2, \dots, m_n)$ the secret data recorded in the blockchain, and the ciphertexts $m_1, m_2, \dots, m_n$ are encrypted by different public keys of n users. The aggregator blinds m_i by importing a random blind value v as

  $$Blind(m_i) = Add(m_i, En_{pk_i}(v)), \tag{8}$$

  where Add represents the addition gate in the homomorphic cryptosystem. Note that the blinding process is executed directly on M by the aggregator due to the additive homomorphism of the encryption algorithm.

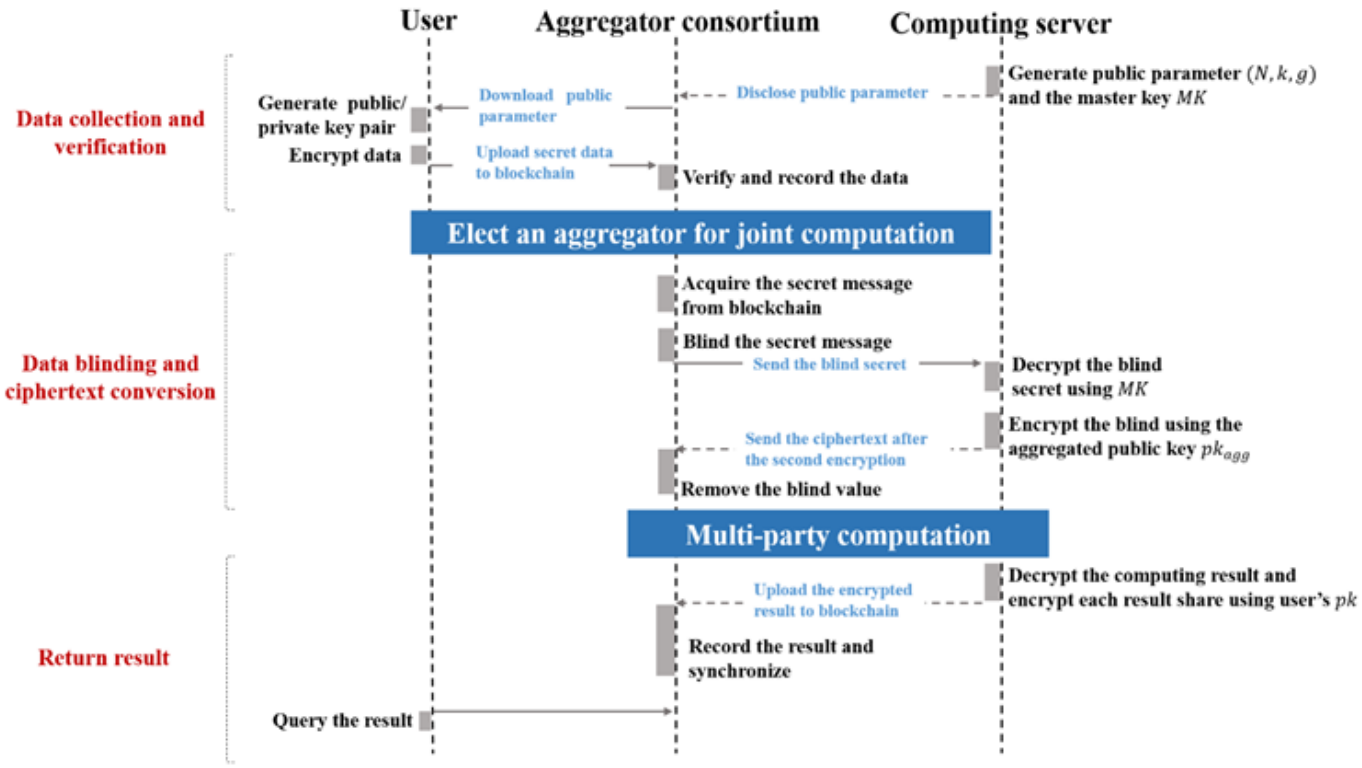

Figure 2. Homomorphic encryption-based SMPC mechanism

- **Ciphertext conversion.** The homomorphic encryption requires that the ciphertexts are encrypted under the same public key, hence converting the ciphertexts is necessary before multi-party computation. Specifically, the selected aggregator sends the blinded data to the computing server, and the computing server decrypt it using the master key. The decrypted blinded data is only visible to the computing server and will not be disclosed. According to the decryption algorithm introduced in **Decryption by** MK, the computing server obtains the decrypted blinded data from different users. Then these data are re-encrypted by an extra aggregated public key for homomorphic computation. We define by pk_{agg} the aggregated public key, which can be calculated based on all public keys of users. Especially, the secret key defined by sk_{agg} corresponding to pk_{agg} is based on the secret keys of users. Therefore, it is almost impossible to decrypt the ciphertext encrypted by pk_{agg}, unless all users disclose their own secret keys and collude to infer sk_{agg}. Of course, the computing server can decrypt this ciphertext using the master key MK.
- **Multi-party computation.** After obtaining the ciphertexts encrypted by pk_{agg} through ciphertext conversion, the fundamental addition and multiplication can be achieved. Given that the encryption algorithm is guaranteed to the additive homomorphism, the addition can be performed directly on the ciphertext that re-encrypted under the aggregated public key pk_{agg}. For multiplication, it requires the cooperation between the computing server and the aggregator, which can be found from [16] in detail.
- **Result return.** The multi-party computation is executed based on the interaction between aggregator consortium and the computing server, while the users just need to upload their individual secret data to the blockchain network. Similarly, the computation result is calculated under the aggregated public key encryption, hence the result should be decrypted by the computing server using the master key and re-encrypted using user's public key. The process requires the computing server to recall the ciphertext conversion mechanism, and upload the encrypted result to the blockchain network through the aggregator that interacts with it. Each user accesses the blockchain to download the computation result at any time, which achieves users' flexible participation in the multi-party computation protocol

5. DISCUSSION

In this section, we analyze and discuss the Block-SMPC in terms of data security, access flexibility, attack resistance and potential applications.

5.1 Data security and Verifiability

In order to prevent users' data from being tampered with, some SMPC schemes introduce public verifiable data transmission mechanism that requires extra proof for verification during data sharing. It provides verifiability to other participants but increases the complexity of the cryptosystem. In our scheme, users encrypt their individual data using the secret key before uploading it, and data recorded in the blockchain will be verified by the aggregators. Hence, data security and verifiability are both guaranteed.

5.2 Privacy Protection

There are three lines of defense for privacy protection in our scheme. Firstly, user's individual data is recorded in blockchain in ciphertext. No participant has ability to read the plaintext except the users themselves. Secondly, we separate the management authority of homomorphic key and ciphertext in the SMPC

mechanism. Moreover, the secret key corresponding to the aggregated public key used for ciphertext conversion is based on all users' secret keys, which makes the decryption is almost impossible.

5.3 Flexible Access for User

In most multi-party computation applications, users cannot promise to participate in the SMPC protocol execution throughout. Therefore, in our scheme, users don't need to interact with other components frequently, while the only necessary thing is to encrypt the raw data and upload it to blockchain. After finishing multi-party computation, they can download the computation results from blockchain at any time. Furthermore, our scheme allows a user to resubmit its secret data if it misbehaves in submitting or fails to upload, and the selected aggregator in each round just need to query and download the latest data from the user in blockchain. This provides flexible access for users and makes our scheme scale well in multi-users scenario.

5.4 Avoiding Collusion in Semi-honest Model

Collusion attack is a crucial problem in practical scenarios, especially when the malicious parties can obtain more profit through colluding. Our proposed SMPC scheme can avoid collusion in semi-honest model, where all participants execute the protocol honestly but may record the intermediate information and infer users' inputs. For malicious user, if it wants to obtain the intermedia data, it must collect enough secret keys of others and infer the aggregated secret key to decrypt the ciphertext in homomorphic encryption. For malicious aggregator, collusion in aggregator consortium does nothing for inferring useful knowledge. In the meanwhile, the aggregator selected to cooperate with the computing server is according to the election mechanism, which is not controlled by any aggregator. For the computing server, it is usually an authority like government or a trusted supervision department that does not have enough motivation to collude and interrupt the execution of multi-party computation. It will not read the plaintext if it follows the ciphertext conversion mechanism honestly.

5.5 Application Scenario

The SMPC scheme is suitable for applications with small data volume and high privacy protection requirement. The electronic election is a typical applicable scenario, in which the individual voting results of diverse voters are required to keep confidential, while the final vote of each candidate and the voting process should be transparent. In our proposed SMPC scheme, the voting results can be encrypted locally and uploaded to blockchain. Then the final voting result is disclosed in the blockchain network after performing the statistical process by the aggregator and computing server. Moreover, digital health is another application with high privacy protection requirement. Hey, Charlie [17] is a health management platform to help users recover from opioid addiction and enhance the communication between patients and clinicians. It tracks and collects the behavior data (e.g. the communication and the travel pattern) of patients, which helps them obtain the fastest and most convenient treatment. SMPC ensures the collected behavior data is confidential and the analysis process will not violate patients' privacy

6. CONCLUSION

In this paper, we propose Block-SMPC, a blockchain-based SMPC scheme, where we design a distributed and scalable data sharing architecture to overcome the single point of failure in data aggregation. Our proposed scheme guarantees the data integrity and verification by combining blockchain, and avoids the collusion attack in semi-honest model by constructing the dynamic aggregator consortium. Moreover, we introduce a homomorphic encryption-based multi-party computation mechanism to promote the privacy security through the authority separation of homomorphic key and ciphertext, while decreasing the interaction frequency between users and other components.

For future work, we schedule to design a detailed multi-party computation protocol based on the proposed Block-SMPC, and evaluate the efficiency through an experiment system based on blockchain. In addition, we will analyze the security and robustness of our proposed blockchain-based SMPC scheme in mathematical.

7. ACKNOWLEDGMENTS

This work was supported in part by the National Nature Science Foundation under Grants 61673275, 61873166 and Science and Technology Commission Shanghai Municipality under Grant 19511102102.

8. REFERENCES

[1] Chen, D., and Zhao, H. 2012. Data security and privacy protection issues in cloud computing. In *2012 International Conference on Computer Science and Electronics Engineering* (Hangzhou, China, March 23 - 25, 2012). IEEE. 647-651. DOI= 10.1109/ICCSEE.2012.193

[2] Yao, Andrew C. 1982. Protocols for secure computations. In *23rd annual symposium on foundations of computer science* (Chicago, USA, November 03-05, 1982). IEEE. 160-164. DOI= 10.1109/SFCS.1982.38

[3] Zhou, Lijing, et al. 2018. Beekeeper: A blockchain-based iot system with secure storage and homomorphic computation. *IEEE Access* 6 (Jun. 2018), 43472-43488. DOI= 10.1109/ACCESS.2018.2847632

[4] Kumaresan, Ranjit, and Iddo Bentov. 2014. How to use bitcoin to incentivize correct computations. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. (Scottsdale, Arizona, USA, November 03 - 07, 2014). ACM. 30-41. DOI= 10.1145/2660267.2660380

[5] Gao, Hongmin, et al. 2019. BFR-MPC: A Blockchain-Based Fair and Robust Multi-Party Computation Scheme. *IEEE Access.* 7 (Aug. 2019), 110439-110450. DOI= 10.1109/ACCESS.2019.2934147

[6] Peter, Andreas, Erik Tews, and Stefan Katzenbeisser. 2013. Efficiently outsourcing multiparty computation under multiple keys. *IEEE transactions on information forensics and security*. 8.12 (Dec. 2013), 2046-2058. DOI= 10.1109/TIFS.2013.2288131

[7] Kamara, Seny, Payman Mohassel, and Mariana Raykova. 2011. Outsourcing Multi-Party Computation. *IACR Cryptology ePrint Archive* (2011), 272.

[8] Carter, Henry, et al. 2016. Secure outsourced garbled circuit evaluation for mobile devices. *Journal of Computer Security* 24.2 (Apr. 2016), 137-180.

[9] Asharov, Gilad, et al. 2012. Multiparty computation with low communication, computation and interaction via threshold FHE. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. (Cambridge, UK, April 15 - 19, 2012). Springer, Berlin, Heidelberg. 483-501

DOI= https://xs.scihub.ltd/https://doi.org/10.1007/978-3-642-29011-4_29

[10] Baum, Carsten, Ivan Damgård, and Claudio Orlandi. 2014. Publicly auditable secure multi-party computation. In *International Conference on Security and Cryptography for Networks*. (Amalfi, Italy, September 03 - 05, 2014). Springer, Cham. 175-196. DOI= https://xs.scihub.ltd/https://doi.org/10.1007/978-3-319-10879-7_11

[11] von Maltitz, Marcel, Dominik Bitzer, and Georg Carle. 2019. Data Querying and Access Control for Secure Multiparty Computation. In *IFIP/IEEE Symposium on Integrated Network and Service Management*. (Arlington, VA, USA, April 08 - 12, 2019). IEEE, 171-179.

[12] López-Alt, Adriana, Eran Tromer, and Vinod Vaikuntanathan. 2012. On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*. (New York, USA, May 19 - 22, 2012). ACM. 1219-1234. DOI= 10.1145/2213977.2214086

[13] Andrychowicz, Marcin, et al. 2014 Secure multiparty computations on bitcoin. In *IEEE Symposium on Security and Privacy*. (San Jose, CA, USA, May 18 - 21, 2014). IEEE, 443-458. DOI= 10.1109/SP.2014.35

[14] ElGamal, Taher. 1985. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE transactions on information theory*. 31.4 (Jul. 1985), 469-472. DOI= 10.1109/TIT.1985.1057074

[15] Bresson, Emmanuel, Dario Catalano, and David Pointcheval. 2003. A simple public-key cryptosystem with a double trapdoor decryption mechanism and its applications. In *International Conference on the Theory and Application of Cryptology and Information Security*. (Warsaw, Poland, May 04 - 08, 2003). Springer, Berlin, Heidelberg. 37 - 54. DOI= https://xs.scihub.ltd/https://doi.org/10.1007/978-3-540-40061-5_3

[16] Cramer, Ronald, et al. 2001. Multiparty computation from threshold homomorphic encryption. *International conference on the theory and applications of cryptographic techniques*. (Innsbruck, Austria, May 06 – 10, 2001). Springer, Berlin, Heidelberg, 280-300. DOI= https://xs.scihub.ltd/https://doi.org/10.1007/3-540-44987-6_18

[17] Hey, Charlie: https://heycharlie.org/