A Protocol for Checking Equality of Data Using Hash in Ideal Model of Secure Multiparty Computation

D.O.I - 10.51201/Jusst12654

http://doi.org/10.51201/Jusst12654

Rashid Sheikh
Research Scholar: Mewar University, Chittorgarh, India
Associate Professor
Acropolis Institute of Technology and Research, Indore, India

Durgesh Kumar Mishra
Sri Aurobindo Institute of Technology, Indore, India

Meghna Dubey
Mewar University, Chittorgarh, India

Abstract- The ideal Secure Multiparty Computation (SMC) model deploys a Trusted Third Party (TTP) which assists in secure function evaluation. The participating joint parties give input to the TTP which provide the results to the participating parties. The equality check problem in multiple party cases can be solved by simple architecture and a simple algorithm. In our proposed protocol *Equality Hash Check*in ideal model, we use a secure hash function. All the parties interested to check equality of their data supply hash of their data to the TTP which then compared all hash values for equality. It declares the result to the parties.

Keywords- Secure multiparty computation, Secure equality check, Ideal SMC model.

Introduction

When multiple parties with their private data want to compute some common function of their data jointly such that the privacy of their data is preserved from one another, the problem is called Secure Multiparty Computation (SMC) [1]. In this scenario all the participating parties do not have trust on one another. But the common function evaluation is in their common interest. Mainly, there are two goals of SMC problem, privacy and correctness of the result. Owing to the heavy use of the Internet which is associated with the distributed system, the scenario of SMC has become highly relevant. The parties geographically distributed on distributed sites

may wish to evaluate a function providing their data. But at the same time they are worried about privacy of their data. During online transaction many sites cooperatively work to process the transactions. But each site does not necessarily want to show the exact value of the data it shares with other party. Therefore in this age of large number of online transactions the SMC is has become highly relevant. Parties frequently need to perform joint computation on their sensitive data while keeping confidentiality of the data. Consider a situation where two or banks wish to know cooperatively the details of a customer from their individual databases. Knowing the details about the customer is in mutual interest of all the participating banks. But no want wants to share their database to other banks as they are competitor in the same sector.

The earlier solutions provided by the researchers for providing solutions to SMC problem used circuit evaluation techniques where the researchers used combinational logic circuits and their logic functions. But the analysis of these solutions showed that they were highly complex and expensive. Later researchers moved on to the cryptographic techniques with use of random numbers. Recently, anonymity techniques are also proposed in which the identity of the parties holding the data is hidden from other parties to achieve the privacy. In this chapter, we are going to focus on the various research work done in the area of the SMC, the research gaps, and our attempts to fill this gap. We also provide some future scope in the area of SMC.

The development of networking, the Internet, and distributed system has provided huge opportunities for joint computations where multiple parties perform SMC on their secret data. Due to the concern of the privacy and worldwide regulations made by made by different countries, there exist many scenarios where SMC becomes applicable. Consider following scenarios which will let you understand the practical applicability of the SMC solutions:

1. A person having his DNA pattern wants to about the genetic diseases associated with his DNA pattern. He wants to do some query from a server storing different DNA patters and the diseases associated with those DNA patterns. But at the same time the person does not want to disclose his exact

DNA patter to the server. This privacy-preserving database query can be implemented using SMC solutions.

2. A group of mobile service providers wish to compute together to prepare list of total subscribers active in an area within some specified time interval. This could be due to some police investigation and asked by the intelligence of a country. The companies do not want to disclose their customer details. This is a case of privacy-preserving union of sensitive databases and can be solved using SMC solutions.
3. A group of students wish to know their average marks obtained in an examination but know student wish to disclose his actual marks to other students of the group. This is a case of privacy preserving sum computation which is called as secure sum in the literature. The sum in this example is divided by number of students to get the secure average.
4. Five real brothers who live separately wish to compute their total wealth but know brother wish reveal his wealth to other brothers. This problem can also be solved using secure sum solutions.
5. A bank wants to some loan details of a suspicious customer from other bank but the bank don't want to disclose actual customer details to other bank. This privacy-preserving query can be solved using SMC solutions.
6. A group of police stations in a country wish to search details of a criminal from their databases but no police station wish to show its database to other police station. This is a case of privacy preserving data mining.
7. A client computer in a payment system wish to learn that a QR code is matching or not without showing the actual QR code to the server. This is a case of privacy-preserving matching or equality check.

Formally, consider multiple parties P_0 to P_{k-1} with private data d_0 to d_{k-1} respectively. These k parties want to evaluate function $f(d_{0,\ldots,} d_{k-1})$ without revealing their private data to each other. Based on the type of f many specific SMC problems are devised by researchers and thereafter many real life applications emerged. We pointed out these works in our publications [2, 3].

Architecture of *Equality Hash Check* protocol Ideal Model

In the protocol *Equality Hash Check*in ideal model all the joint parties are required to compute hash of their private data and supply to the TP as shown in the Fig.1.

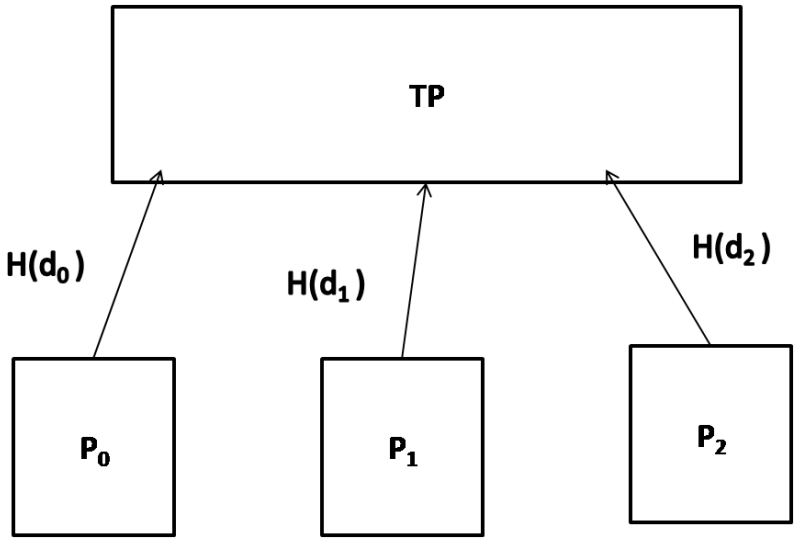

Figure 1:Architecture of *EqualityHashCheck* protocol Ideal Model

Informal Description of *Equality Hash Check*protocol in Ideal Model

The *Equality Hash Check* protocol works in a simple manner. All the participating parties who wish to check the equality of their individual data use a common hash function. All the parties supply hash code of their data to the TP. The TP checks all these hash value for equality. The result of equality check is announced to all the parties. The privacy of the data among participating parties is preserved as these parties do not communicate with each other. Similarly, the data is also hidden from the TP as it receives hash value, not the actual data. The protocol is suitable for semi-honest adversary.

Formal Description of *Equality Hash Check* protocol Ideal Model

The algorithm for *Equality Hash Check* can be described as follows:

Equality Hash Check algorithm in Ideal Model

Input: Parties P_0 to P_{k-1} with data $d_0$to d_{k-1} respectively. A hash function H( ) . A Third Party (TP) assists for comparison of hash values.

Requires: All parties wish to know that their data are equal or not without disclosing individual data to one another.

Step1: For i = 0 to k-1

P_i computes H(d_i) and sends it to TP

Step2: The TP compares all the hash values for equality and declares result to all the parties.

Step3: Stop

Performance analysis of *Equality Hash Check* protocol in Ideal Model

The *Equality Hash Check* protocol is simplest of its kind as the parties send hash code of their data to the TP. The TP compares these hashes to announce the result of equality check. The comparison of *Equality Hash Check* in ideal model and real model is given in the table 1. There is an obvious reason of presence of TP that the implementation of *Equality Hash Check* in ideal model is complex and expensive. From the table it is clear that the amount of computation and communication both are less in ideal model as compared to the real model *Equality Hash Check.*

Table-1: Comparison of *Equality Hash Check* in real and ideal model

S. No.	Criteria	Real Model	Ideal Model
1.	TP	No TP exists	TP is used for comparison
2.	Encryption-Decryption	No encryption and decryption is performed.	No encryption and decryption is performed.
3.	Randomization	No random number is used	No random number is used
4.	Computations	Only hashing and comparison by pair of parties	Only hashing by parties and comparison by TP
5.	Communications	Only one message exchange: send GO or sendhash to next party.	Each party sends hash to the TP
6.	Threat of attack	Normally, the hash function is infeasible	Normally, the hash function is infeasible

Conclusion

The privacy preserving function evaluation is the need of today's world as the sensitive data from multiple sources may be shared by multiple sites for joint computation. Secure equality check is one of the cases where a set of parties need to compute the data for equality. For example two police stations situated far apart want to check some biometric data of a criminal such that both the stations do not want to disclose actual data to one another. In this case two-party equality check would be useful. In our work we have extended the two-party equality check to multiparty case. Our protocol secure *Multi Equality Check*uses additive homomorphic cryptosystem where the sum of ciphertext is equal to the ciphertext of the sum. Our work is suitable for semi-honest adversaries.

The protocol *Multi Equality Check* uses public-private key generation using homomorphic encryption, encryption, and decryption. We proposed a novel equality

check scheme using hash function. The protocol *Equality Hash Check* ensures checking equality among multiple parties with the help of comparison of hash functions. It has an advantage of less computation as compared to the *Multi Equality Check*. We have proposed *Equality Hash Check* for both real and ideal model of SMC. In real model the parties compare their hashes pair-by-pair until all the parties are traversed. In the ideal model the hashes are provided to a TP. It is the TP which checks whether all the hashes are equal and provides result to all the parties. The privacy is preserved from TP as it receives hash of data not the actual data. The hash function is an irreversible function. That mean given the hash, it is infeasible to compute data from the hash. Similarity, privacy is preserved among parties as they do not communicate with each other. But, the cost of maintaining the TP is an overhead.

In future the equality check protocols can be devised for malicious adversaries.

References

[1] W. Du and M.J. Atallah, "Secure Multiparty Computation Problems and Their Applications: A Review and Open Problems," In proceedings of new security paradigms workshop, Cloudcroft, New Maxico, USA, page 11-20, Sep. 2001.

[2] R. Sheikh and D. K. Mishra, "Secure Multi-Party Computation: A Research Proposal," in the proceedings of the International Conference on Computer and Communication, ICCC2012, pages 876-881, Bhopal, India, Jan 2012.

[3] R. Sheikh, B. Kumar and D. K. Mishra, "Secure Multi-party Computation: From Millionaires Problem to Anonymizer," in the Information Security Journal: A Global Perspective, Taylor and Francis, Vol. 20, Issue 1, pages 25-33, USA, 2011.

[4] C. Clifton, M. Kantarcioglu, J. Vaidya, X. Lin, and M. Y. Zhu, "Tools for Privacy-Preserving Distributed Data Mining," J. SIGKDD Explorations, Newsletter, Vol.4, No.2, ACM Press, pages 28-34, Dec. 2002.

[5] R. Sheikh, B. Kumar and D. K. Mishra, "Privacy-Preserving k-Secure Sum Protocol," in International Journal of Computer Science and Information Security, Vol. 6 No.2, pages 184-188, USA, Nov. 2009.

[6] R. Sheikh, B. Kumar and D. K. Mishra, "A Distributed k-Secure Sum Protocol for Secure Multi-party Computation," in Journal of Computing, USA, Vol. 2, Issue 3, pages 68-72, Mar. 2010.

[7] R. Sheikh, B. Kumar and D. K. Mishra, "Changing Neighbors k-Secure Sum Protocol for Secure Multi-party Computation," in International Journal of Computer Science and Information Security, Vol. 7 No.1, USA, Jan. 2010.