Behavioral Identification of Trusted Third Party in Secure Multiparty Computing Protocol

Zulfa Shaikh[1] and Poonam Garg[2]

[1]Faculty of Computer Applications, Acropolis Institute of Technology & Research, Indore (M.P.)
shaikh.zulfa@gmail.com

[2]Deptt of Information Technology, Institute of Management Technology, Ghaziabad
pgarg@imt.edu

ABSTRACT

We present a solution for identification and reduction of malicious conduct by Trusted Third parties (TTPs) in Secure Multiparty Computing Protocol. This paper also proposes a secured protocol for computation and defines encryption to be performed before sending inputs for computation. Our protocol uses e-envelopes for sharing keys between parties and TTPs. This key sharing is done on the basis of RSA algorithm. This ensures that parties send their data in encrypted manner to TTPs in order to maintain privacy and security of inputs. Also, single and multi trusted third party model is compared and the probabilistic evidences for them have been analyzed with security analysis graphs.

KEYWORDS

Secure Multiparty Computation (SMC), Trusted Third Party (TTP), Single TTP, multi TTP, privacy, security, and correctness.

1. INTRODUCTION

In the fast growing Internet world, there is an enormous sharing of data and so in today's development, security and privacy is a biggest challenge. Secure multi-party computation (SMC) problem [3] is the problem of *n* parties to compute a private function of their inputs. The computation performed by TTP should be such that it announces the correct results of computation. For computing correct results privacy and security has to be maintained in the protocol as parties may try to misuse the other party's data. Consider n inputs of parties inputs *x1, x2, …, xn*, where *xi* is the data of party *Pi* and the TTP will compute a function $f(x1, x2, \ldots, xn) =$ y and will send the results to respective parties or announces publicly. Security is meant to achieve correctness of the result of computation and keeping the party's input private even some of the parties are corrupted. In real world scenario, trust on third party performing the computation is doubtful, so need is to design and develop a protocol where party's privacy can be maintained and malicious conduct of third party can be identified and reduced. In figure 1, the general SMC framework has been defined.

DOI : 10.5121/ijsptm.2012.1306

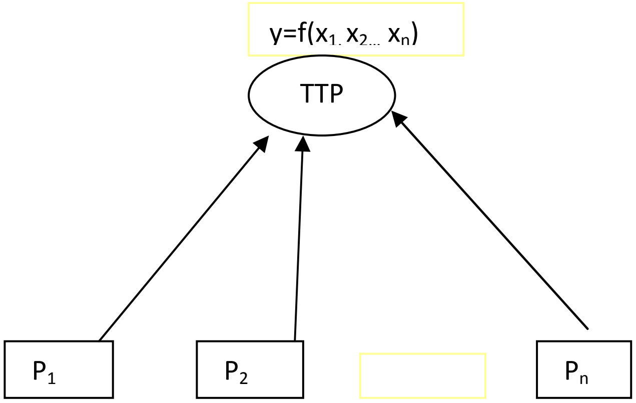

Figure 1. General SMC Model

In this paper, SMC model with single third party and multi third party have been defined. In single third party SMC model, all the parties involved in computation hand their inputs to the single third party for computation whereas in multi third party SMC model, same computation is performed by number of trusted third parties selected at runtime. The paper helps in identifying and reducing malicious conduct by TTPs in multi third party SMC model. This protocol also proposes secured computation by introducing encryption algorithm at party layer.

2. BACKGROUND

SMC problem is the problem of n parties to compute a private function of their inputs in a secure method, where security means the correct result computed by the TTPs for maintaining the privacy of the parties as some of the parties may want to misuse the other party's data. We assume that the inputs are *x1,x2,…xn* where xi is the data of party Pi and the TTP will compute a function f(x1,x2,….xn)=y and announce the result y [1]. Security is meant to achieve correctness of the result of computation and keeping the party's input private even if some of the parties are corrupted. In figure 1, trusted third party is used for doing the computation on the inputs provided by the parties. According to [2], the major problem with this approach is that it is difficult to find the third party which is trusted by all the parties providing the inputs and to control the function of adversaries.

Yao's introduced the SMC problem in [3].The first solution uses a centralized TTP which is selected by majority of honest party, which shows synchronous system with cryptography [4].After handing up the inputs to a trusted third party, security increases but there is a chance that the trusted third party behaves like a malicious adversary. It was demonstrated analytically as well as experimentally, the performance characteristics and security and proved that for the range of numbers; Yao's protocol is secure [5]. The idea was extended to multiparty computation by many researchers [6]. They used circuit evaluation protocols for secure computation. Earlier research focused on theoretical studies. Later, some real life applications emerged like Private Information Retrieval (PIR) [7, 8], Privacy-preserving data mining [9, 10], Privacy-preserving geometric computation [11], Privacy-preserving scientific computation [12], Privacy preserving statistical analysis [13] etc. A detailed review of SMC research is provided by Du et al. in [14] where they developed a framework for problem discovery and converting normal problem to

SMC problem. A review of SMC with special focus on telecommunication systems is given by Oleshchuk et al. in [15].

Aiming at privacy preserving computing of statistical distribution, which is frequently encountered in statistics, and based on the intractability of computing discrete logarithm and using rigorous logic, they proposed the solution. [16] Presented the protocols allowing the players to securely solve standard computational problems in linear algebra such as determinant of matrices product, rank of a matrix, and determine similarity between matrices. [17] Presented TASTY, a novel tool for automating, i.e., describing, generating, executing, benchmarking, and comparing, efficient secure two-party computation protocols. They used TASTY to compare protocols for secure multiplication based on homomorphic encryption with those based on garbled circuits and highly efficient multiplication. [18] Presented a hybrid-secure MPC protocol that provides an optimal trade-off between IT robustness and computational privacy. [19] Presented a solution to the Secure Multi-party Computation (SMC) problem in the form of a protocol that ensures zero-hacking. The solution comprises of a protocol with several trusted third parties (TTPs) where there is a possibility of threat to the security.

3. PROPOSED WORK

The basic need during SMC is to obtain correct results of computation maintaining privacy and security in the protocol. As trust on third parties performing joint computation is doubtful in real scenario. So the need is to define more secure protocol announcing the right result of computation. The objectives of present study are:-

- To define and compare single and multi third party computing model.
- To analyze and store the behavior of third parties in several rounds of computation.
- To identify and reduce malicious conduct of trusted third parties in order to obtain correct results of computation.
- To make computation secured.

3.1 Proposed Protocol: SMC_Encryption using e-envelopes

1. Parties (P_1, P2,…,P_n) have inputs (x_1,x_2,…x_n).
2. The inputs (x_1, x2…x_n) before sending for computation to TTPs must be encrypted.
3. The proposed encryption uses RSA algorithm for sharing keys and establishing session.
4. TTPs have its own private key and send public keys to parties.
5. Parties using public key of TTPS creates e-envelopes for session keys.
6. TTPs using their private keys decrypt the session key.
7. Then with the help of session keys parties can send cipher text to TTPs.
8. TTPs with the help of session key will decrypt the cipher text.

This proposed encryption algorithm will make the SMC protocol more secured and computation will be efficient in terms of correctness. As the trusted third party performing computation must provide correct results.

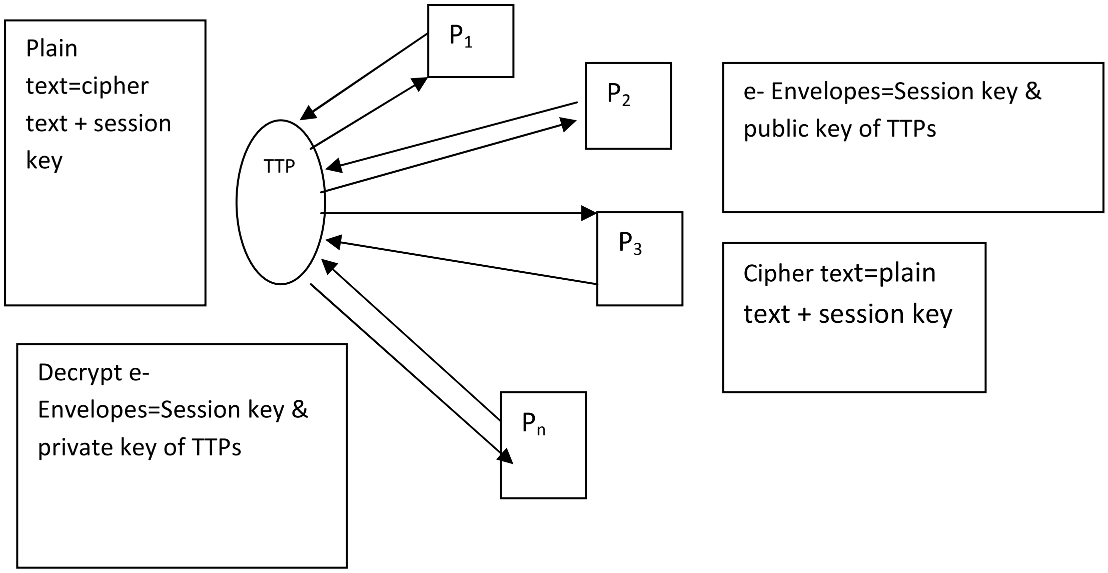
Plain text=cipher text + session key
P_1
P_2
e- Envelopes=Session key & public key of TTPs
TTP
P_3
Cipher text=plain text + session key
Decrypt e-Envelopes=Session key & private key of TTPs
P_n

Figure 2: SMC_E-envelopes

3.2 Architectural Framework:

3.2.1. Single TTP Computation Model

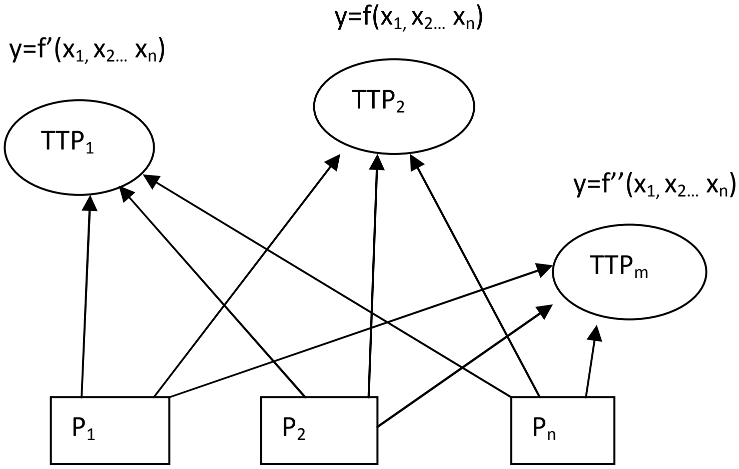
$y=f(x_1, x_2 \ldots x_n)$
$y=f'(x_1, x_2 \ldots x_n)$
TTP_2
TTP_1
$y=f''(x_1, x_2 \ldots x_n)$
TTP_m
P_1
P_2
P_n

Figure 3. SMC_ Single TTP Computation Model

3.2.2. Multi TTP Computation Model

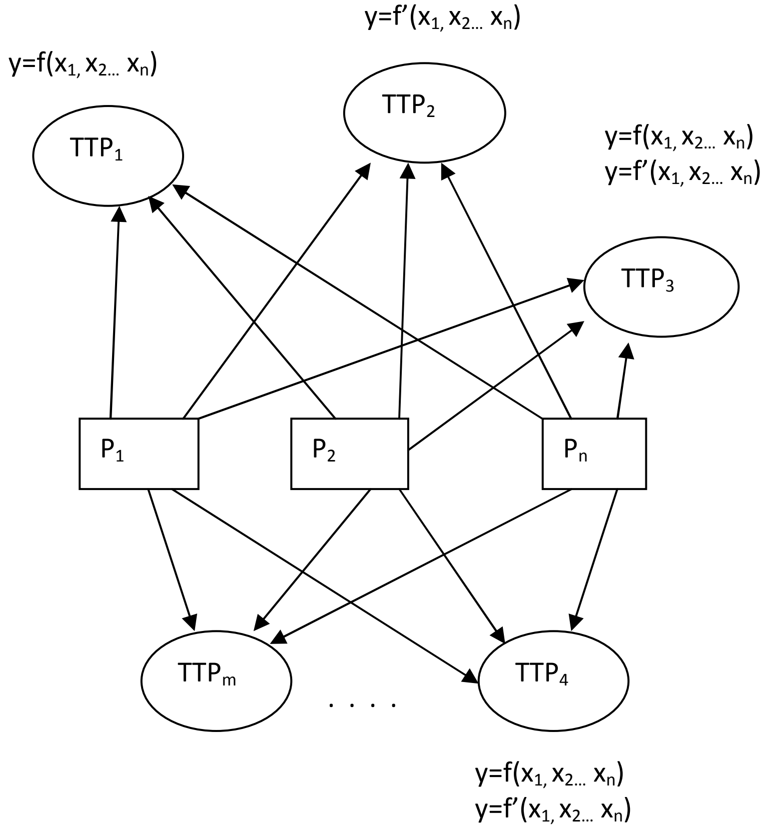

Figure 4. SMC_ Multi TTP Computation Model

In figure 1 and 2 the architectural framework of SMC model is designed. The model works in two different environments: one is the single TTP, selected for performing the computation from a pool of TTPs and second, multiple TTPs performing computation on single function.
The assumptions in both the models are:

- Due to critical mission data no party will share its information with other parties involved in computation.
- Parties provide their inputs to TTP or TTPs for computation.
- TTPs are selected at runtime from a pool of TTPs.

The major concern with this computation is that, what if TTPs involved in computation are malicious? The results will be:

- Correctness in the output cannot be ensured.
- The malicious TTPs may affect security and privacy issues of SMC as well.

Comparing single TTP with multi TTP model the advantage of multi TTP model is that the protocol does not rely on one TTP but on majority providing the identical results. This will give more clear identification in correctness of results. On the other hand, in single TTP model the protocol selects a single TTP for computation at runtime from a pool of TTPs. This makes the protocol inefficient as if, a TTP computing the function is malicious then it may announce the incorrect result of computation and there is no alternative, other than relying on the malicious TTP.

3.3 Probability Analysis:

***Case 1:* Probability of malicious conduct in single and multi TTP model before selection of TTPs**

In single TTP model if the TTPs are selected at runtime using randomization function (R_f) then the probability of malicious conduct is:

$$P\ (TTP_{single}) = 1/m \tag{1}$$

Here m is the total number of TTPs involved in the model.

Multi TTP model will have the probabilistic analysis of malicious conduct by TTPs, during selection, is:

$$P\ (TTP_{multi}) = r/m \tag{2}$$

where m is the total number of TTPs in the model and r is the number of TTPs that will perform computation.

***Case 2:* Probability of malicious conduct in single and multi TTP model after selection of TTPs**

Probability of malicious conduct in single TTP model is:

$$P\ (TTP_{single}) = 1 \tag{3}$$

Probability of malicious conduct in multi TTP model is:

$$P\ (TTP_1) = P\ (TTP_2) = P\ (TTP_3) =1/3$$

Here $TTP_{1,}$ TTP_2 and TTP_3 are selected for computation on a particular function f.

In generalized form, suppose m is the total number of TTPs performing the computation on a particular function f and r is the number of TTPs that can perform malicious conduct out of m TTPs selected at runtime is:

$$P\ (TTP_{multi}) = r/m \tag{4}$$

If all the TTPs performing the computation are malicious then r=m, hence

$$P\ (TTP_{multi}) = m/m =1 \tag{5}$$

If this is the case single TTP and multi TTP behavior will be identical and more often multi TTP model will have high computation cost with no effectiveness.

In single TTP model the probability of malicious conduct is either 0 or 1.Contrary in multi TTP model it increases gradually as malicious TTPs increases.

3.4 Graph Analysis

After selection of TTPs

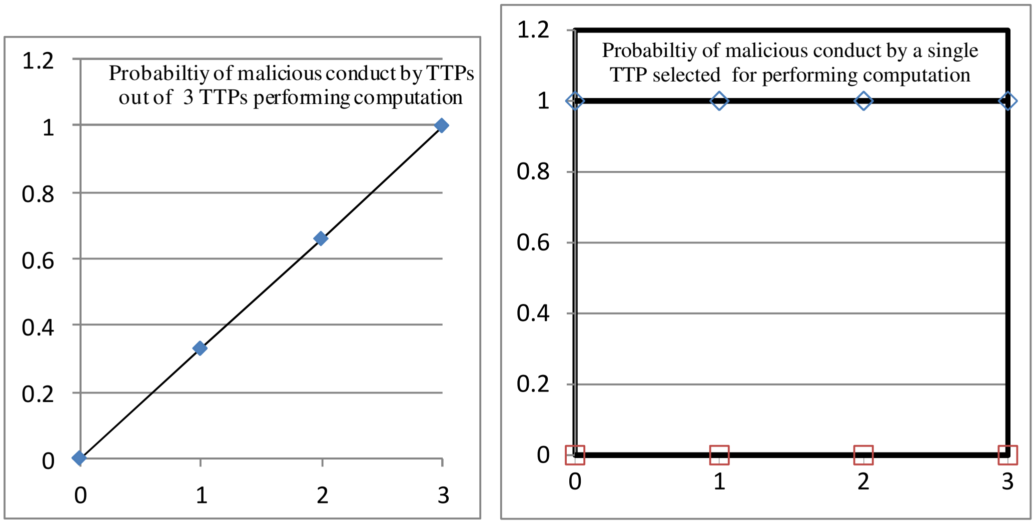

Figure 5. Graph analysis for multi and single third party computation

In single TTP model the probability of malicious conduct is either o or 1.Contrary in multi TTP model it increases gradually as malicious TTPs increases. In graph, 3 TTPs are performing computation on particular function. So the probability of being malicious during computation will be 1/3, 2/3 and 3/3. In this paper, we have also worked on identification and reduction of malicious conduct of TTPs in multi TTP environment. There are following cases considering different behaviors of Third party.

***Case 1:* (number of TTP= m)> ½ is providing identical and correct results**

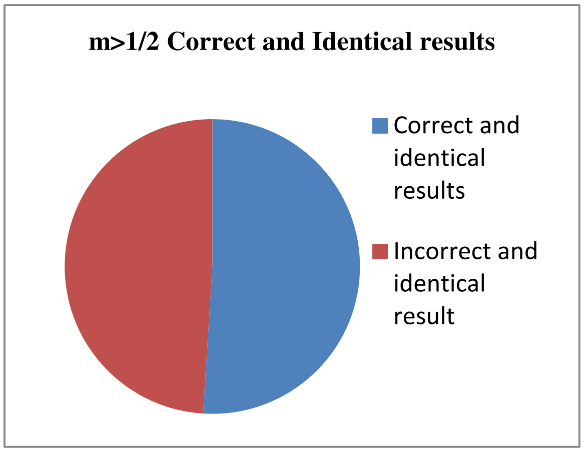

Figure 6. Correct and Identical results

If this is the result of computation where number of TTPs >50% is giving identical and correct results and remaining TTPs are providing some other results then in real model, we have to consider the majority providing the same and correct results.

***Case 2:* m>1/2 (Majority is providing identical but wrong results of computation)**

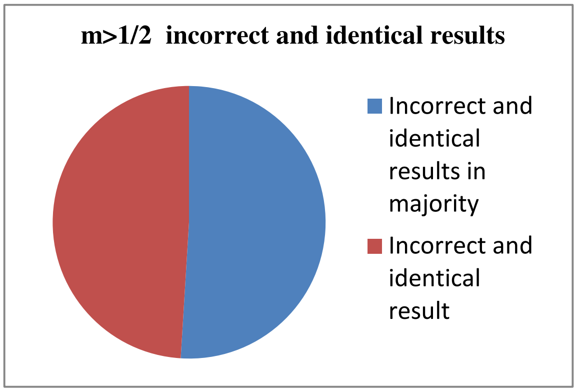

Figure 7. Incorrect and Identical results

If this is the case of computation then it is difficult to identify the correctness in result as majority of TTPs are providing same wrong outputs. This case leads to protocol failure, and this kind of scenario is rarely possible in real world where majority of TTPs are giving same wrong results.

***Case 3:* m=1/2 (half of the TTPs providing identical and correct results and remaining half giving identical and wrong outputs)**

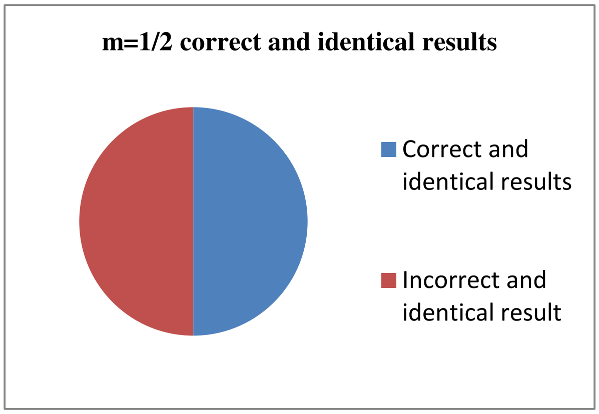

Figure 8.Correct and Identical Results

This case also leads to system unacceptability and failure as equal number of TTPs providing same results in both the half, and so almost difficult to identify the correct results.

***Case 4:* Majority of TTPs giving identical and correct results and remaining TTPs giving different outputs in groups.**

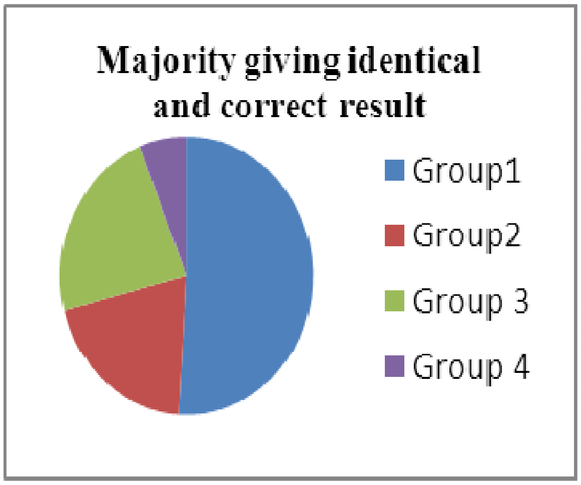

Figure 9. Majority giving identical and correct results

In this case it is almost reliable to go with majority.

3.5 Identification and Reduction of Malicious conduct by TTPs

Case 1: m>1/2 (providing identical and correct results of computation)

Here our aim is to identify the malicious TTP resulting in protocol disruption in multi TTP computing model. The TTP performing malicious conduct is identified in several rounds of computation.

Consider a scenario where total number of TTPs used in the model=5. The number of TTPs performing the computation out of 5 TTPs is selected at runtime.

Table 1

TTPs performing computation in different rounds and their analysis

Round	Name of TTPs selected	Identical results in majority	Correct result	Malicious conduct	No of TTPs giving same result	Groups of TTPs
(No of TTPs=3) I	TTP$_1$	√	√	x	2	A
	TTP$_2$	x	x	√		B
	TTP$_3$	√	√	x		A
(No of TTPs=4) II	TTP$_2$	x	x	√	3	A
	TTP$_3$	√	√	x		B
	TTP$_4$	√	√	x		B
	TTP$_5$	√	√	x		B

Note: Correct result parameter depends on majority of TTPs giving identical results.

The identification of TTPs is made on the basis of following steps:

1. Find the name of TTPs not in majority of correct results in several rounds of computation.
2. Now the TTPs not in majority of identical results are considered to be marked.
3. Find the number of times that TTP is out of majority in several rounds of computation.
4. The TTP is then stored in SMC_multi_trouble zone with the value (number of times) parameter.

In the above scenario if TTP2 is not in the majority of identical results in several rounds of computation, the TTP$_2$ is considered to be malicious trying to deviate from the protocol. From table 1, following observations are drawn:

Round I- TTP2 is not in the majority of identical results and is marked.
Round II- TTP2 is not again in the majority of identical results.

In this way TTPs deviating from the protocol and behaving as an “Odd Man Out” can be identified by saving their all records of computation performed in several rounds.

Now on the basis of parameters used in table 1 we divided the TTPs to lie in either of two zones: SMC_Multi_Safezone and SMC_Multi_Troublezone.

Table 2

Behavior identification of TTPs

Name of TTP	SMC_Multi_Troublezone	SMC_Multi_Safezone
TTP_1	-	1
TTP_2	2	-
TTP_3	-	2
TTP_4	-	1
TTP_5	-	1

This gives identification of TTP behaving maliciously in the protocol and therefore its involvement in computation has to be reduced if the value parameter of TTP in trouble zone is highest and computation should be performed with the TTPs giving highest value in safe zone.

Case 2: Majority providing incorrect and identical output of computation

Table 3

Majority giving identical but wrong output of computation

Round	Name of TTPs selected	Identical results in majority	Correct result	Malicious conduct	No of TTPs giving same result	Groups of TTPs
(No of TTPs=3) III	TTP_1	√	x	√	2	A
	TTP_2	√	x	√		A
	TTP_3	x	√	x		B

From table 2, Round III- $TTP_2$is in the majority of identical results but it is the case when computation done is incorrect.In this case, where majority is giving identical but wrong output of computation then protocol will go with the majority and this scenario leads to system failure as correctness parameter of SMC does not work.

The only solution is to see the behavior of TTPs after each round of computation in “Table: Behavior identification of TTPs” if there are entries of TTPs performing the computation.The reference to “Behavior identification of TTPs” is made after every computation.

- Find the number of times a TTP involved in computation is in the SMC_Multi_Troublezone (TTP_2 = highest trouble zone value=2).
- Find the number of times a TTP involved in computation is in the SMC_Multi_Safezone (TTP_3 = highest safe zone value=2).

- The trust on SMC_Multi_Safezone column with high values will be more than SMC_Multi_Troublezone.

The behavior of TTPs performing computation is identified through table" Behavior----". From the table, the conclusions are, TTP_2 has the highest troublezone value as 2 whereas TTP_3 has the highest safe zone value as 2. So the trust on TTP_3 will be high.

- Now, the same computation has to be re-performed with other remaining TTPs in the pool with highest safezone value.
- If the remaining output matches with TTP_3 and is in the majority then correctness can be ensured.
- The entries have to be updated in Table 3 for the computing TTPs.

Case 3: Equal number of TTPs giving identical results.

Table 4

Equal number of TTPS giving identical results

Round	Name of TTPs selected	Identical results in majority	Correct result	Maliciou s conduct	No of TTPs giving same result	Groups of TTPs
(No of TTPs= 4) IV	TTP_2		x	√	2,2	A
	TTP_3	**identical**	x	√		A
	TTP_5		√	*x*		*B*
	TTP_1	***identical***	√	*x*		*B*

In this case it is almost difficult to identify the correct result but if there is an entry in trouble and safe zone column of the table 3, then reference to table is the only solution in identification of wrong conduct and steps of case 2 has to be followed.

4 RESULTS AND CONCLUSIONS

When most of the operations are jointly performed today, there is a need of more secured protocols which can maintain privacy and assure correctness. . This paper has defined a secured protocol for computation and proposes an encryption to be performed before sending inputs for computation. The protocol uses e-envelopes for sharing keys between parties and TTPs. This ensures that parties send their data in encrypted manner to TTPs in order to maintain privacy and security of inputs. In this paper single and multi third party SMC environment is defined, compared and analyzed. The need of using multi TTPs computing model is that of privacy concern as parties providing inputs for computation may not be able to know the third party performing computation as the TTPs are selected at runtime from the pool of TTPs. While using multi third party environment for computation, different cases were studied for identification of malicious conduct by TTPs .The behavior of TTPs is analyzed, in several rounds. Analyzing the behavior of TTPs, by looking at the highest count in trouble_zone column of Behavior Identification of TTPs, the involvement of that TTP in computation is reduced and the highest safe_ zone count TTP is given more rights at computation. This reduces the malicious TTPs and increases the system acceptability.

REFERENCES

[1] C.Clifton, M. Kantarcioglu, J. Vaidya, X. Lin and Y. Michael. (2002), Tools for privacy preserving distributed data mining , SIGKDD Explorations Volume – 4,Issue – 2, 1-8.

[2] J.Vaidya, and Chris Clifton. (2003), Leveraging the Multi in Secure Multi-Party Computation, in the proceeding of the 2003 ACM workshop on privacy in electronic society, ACM Press.

[3] A.C.Yao. (1982), Protocol for secure computations, in Proc. 23rd IEEE Symposium on the Foundation of Computer Science (FOCS), IEEE, 160-164.

[4] O.Goldreich, S. Micali, A Wigderson. (1987), How to play any mental game- a complete theorem for protocol with honest majority, in the proceeding of 19th ACM symposium on the theory of computing (STOC), 218-229.

[5] I.Ioannidis and A. Grama.(2003), An efficient protocol for Yao's Millionaires Problem, in the Proceeding of 36th Hawaii International Conference on System Sciences,HICSS'03, IEEE Press, 6-11.

[6] O.Goldreich, S. Micali, and A. Wigderson. (1987), How to play any mental game, in STOC '87: Proceedings of the nineteenth annual ACM conferenceon Theory of computing, New York, NY, USA: ACM, 218-229.

[7] B.Chor and N.Gilbao. (1997), Computationally Private Information Retrieval (Extended Abstract), in proceedings of 29th annual ACM Symposium on Theory of Computing, El Paso, TX USA.

[8] B. Chor, E. Kushilevitz, O. Goldreich, and M. Sudan. (1995), Private Information Retrieval, in proceedings of the 36th Annual IEEE Symposium on Foundations of Computer Science, Milwaukee WI, 41-50.

[9] Y. Lindell and B. Pinkas. (2000), Privacy preserving data mining, in advances in cryptography-Crypto2000, lecture notes in computer science, vol. 1880,2000.

[10] R. Agrawal and R. Srikant. (2000), Privacy-Preserving Data Mining, in proceedings of the 2000 ACM SIGMOD on management of data, Dallas, TX USA, 439-450.

[11] M. J. Atallah and W. Du. (2001), Secure Multiparty Computational Geometry, in proceedings of Seventh International Workshop on Algorithms and Data Structures(WADS2001), Providence, Rhode Island, USA,165-179.

[12] W. Du and M.J. Atallah. (2001), Privacy-Preserving Cooperative Scientific Computations, in 14th IEEE Computer Security Foundations Workshop, Nova Scotia, Canada, pages 273-282, Jun. 11-13 2001.

[13] W. Du and M.J.Atallah. (2001), Privacy-Preserving Statistical Analysis, in proceedings of the 17th Annual Computer Security Applications Conference, New Orleans, Louisiana, USA, 102-110.

[14] W. Du and M.J. Atallah. (2001), Secure Multiparty Computation Problems and Their Applications: A Review and Open Problems, in proceedings of new security paradigm workshop, Cloudcroft, New Maxico, USA, 11-20.

[15] V. Oleshchuk, and V. Zadorozhny. (2007), Secure Multi-Party Computations and Privacy Preservation: Results and Open Problems, Telektronikk: Telenor's Journal of Technology, vol. 103, no.2.

[16] Q. Zheng, S. Shan Luo, Y. Xin. (2010), Research on the Secure Multi-Party Computation of some Linear Algebra Problems, Applied Mechanics and Materials, Trans Tech Publication, Switzerland, Vols. 20-23, 265-270.

[17] W. Henecka, S.K. Ogl. (2010), TASTY: tool for automating secure two-party computations, in the Proceedings of the 17th ACM conference on Computer and Communications Security.

[18] C. Lucas, D. Raub, U. Maurer. (2010), Hybrid-secure MPC: trading information-theoretic robustness for computational privacy, PODC '10 Proceeding of the 29th ACM SIGACT-SIGOPS symposium on Principles of distributed computing.

[19] Mishra,D.K.,Chandwani,M. (2008),A zero-hacking protocol for secure multiparty computation using multiple TTP , TENCON 2008 - 2008 IEEE Region 10 Conference.