# Secure Multiparty Computation Protocol: Basic Building Blocks Methods

**Prof. Anand R. Padwalkar**
Department Of Computer Application
Shri Ramdeobaba College Of Engg
& Management, Nagpur, India
anandpadwalkar@gmail.com

**Prof. Prajakta Pande**
Department Of Computer Application
Shri Ramdeobaba College Of Engg
& Management, Nagpur, India
mepnp@rediffmail.com

**Ms. Vidhi Dave**
M.Tech. TGPCET,
Nagpur ,India
vidhiamehta@rediffmail.com

***Abstract*: In a modern information-driven society, the everyday life of individuals and companies is full of cases where various kinds of private information is an important resource. While a cryptographer might think of PIN-codes and keys in this context, this type of secrets is not our main concern here. Secure Multiparty Computation (SMC) allows parties to compute the combine result of their individual data without revealing their data to others. SMC also allows parties with similar background to compute results upon their private data, minimizing the threat of disclosure. On SMC many eminent researchers give their protocols especially in modifier tokens to compute result. As the data involved in computation was encrypted, without revealing the data right result can be computed and privacy of the parties is maintained.**

**In this paper, we survey the basic paradigms and notions of secure multiparty computation and discuss their relevance to the field of privacy-preserving data mining. In addition to reviewing definitions and constructions for secure multiparty computation, we discuss the issue of efficiency and demonstrate the difficulties involved in constructing highly efficient protocols. Finally, we discuss the relationship between secure multiparty computation and privacy-preserving data mining, and show which problems it solves and which problems it does not secure sum computation, researchers show their interest. The process involves encrypting data in a manner that it does not affect the result of the computation. Virtual parties are created by all organizations and encrypted data is distributed among them. Modifier tokens are generated along encryption which are assigned to virtual parties, and finally used in the computation. The computation function uses the acquired data and are responsible for generating the basic methods to be used for extraction of the given set of data.**

***Keywords*: Secure Multiparty Computation (SMC); no loss secured approach, Third Party (TP); Secure Sum Protocol; Hybrid, Pool of function.**

## I. INTRODUCTION

A Common approach to describing intuitively the meaning of security in multiparty protocols for general function computation uses the notion that the secrecy ensured by a trusted party should be the standard by which a general protocol is measured. That is, the execution of a protocol should reveal nothing more than is revealed in the situation where a trusted party is available, namely, nothing more than the value of $f(x_1,\ldots x_n)$. Other properties, such as correctness, independence of input selection, and fairness are equally important, and the literature contains a variety of approaches to describing and defining these desired properties.

Privacy-preserving data mining considers the problem of running data mining algorithms on confidential data that is not supposed to be revealed even to the party running the algorithm. There are two classic settings for privacy-preserving data mining (although these are by no means the only ones). In the rest, the data is divided among two or more different parties; the aim being to run a data mining algorithm on the union of the parties' databases without allowing any party to view another individual's private data. In the second, some statistical data that is to be released (so that it can be used for research using statistics and/or data mining) may contain confidential data; hence, it is first modified so that (a) the data does not compromise anyone's privacy, and (b) it is still possible to obtain meaningful results by running data mining algorithms on the modified data set. In this paper, we will mainly refer to scenarios of the first type.

Secure Multiparty computation problem is not a problem of single party as the name itself says it is the problem of multiple parties' i.e. n parties. In SMC problem, n parties want to compute their private data or function as input in secure mode means data of individual party cannot be disclose or reveal to other and correct result is computed.

The aim of *secure multiparty computation* is to enable parties to carry out such distributed computing tasks in a secure manner. Whereas distributed computing classically deals with questions of computing under the threat of machine crashes and other faults. Two important requirements on any secure computation protocol are *privacy* and *correctness*. The privacy requirement states that nothing should be learned beyond what is absolutely necessary; the correctness requirement states that each party should receive its correct output. The setting of secure multiparty computation encompasses tasks as simple as coin-tossing and broadcast, and as complex as electronic voting, electronic auctions, electronic cash schemes, contract signing, anonymous transactions, and private information retrieval schemes. Due to its generality, the setting of secure multiparty computation can model almost every cryptographic problem.

## II. LITERATURE SURVEY

*Secure multi-party computation* (also known as secure computation or multi-party computation (MPC)) is a subfield of cryptography. The goal of methods for secure multi-party computation is to enable parties to jointly compute a function over their inputs, while at the same time keeping these inputs private. For example, two millionaires can compute which one is richer, but without revealing their net worth. In fact, this very example was initially suggested by Andrew C. Yao in a 1982 paper [1] and was later named the *millionaire problem*.

In this problem two millionaires wanted to know who is richer among them without disclosing their wealth to each other. The solution provided by Yao was for semi honest. Semi honest parties' means they want to know other information also. Then Clifton et al introduce tools for privacy preserving distributed data mining. He gave four efficient methods for privacy preserving computation that can be used to support data mining. They used circuit evaluation protocols for secure computation. All these are the theoretical aspects of SMC. Privacy preserving data mining using SMC has great importance and many applications have been developed. Du et al. reviewed the various industrial problems and listed.

A new concept was put forward by D.K.Mishra through his multi-layer protocols. Initially a two-layer protocol along with a tentative architecture for its implementation was proposed. This two-layer protocol was improvised by a three layer protocol in which an anonymizer layer was added in between the parties and the third party. This new layer hides the information of the parties from the third party, who computes the data and provides the result. After theoretical studies few practical problems of SMC was introduced i.e. Privacy information retrieval problem (PIR), Privacy preserving Statistical analysis, Privacy Preserving Scientific Computation, Privacy preserving Data Mining, Privacy Preserving Geometric Computation etc. In PIR problem there is a client and a server, client want to hack the information from the server without letting know I to server and server do not want that client ever know the binary sequence. Beside this, Lindall et al [2] and Agrawal et al [3] respectively provide cryptographic technique and solutions for SMC and for mining association rule, provide fast and secure algorithm. Through PORTIA project of Rebecca Wright some of the problems of SMC and privacy preserving data mining got the solution. Many eminent researchers provided their views and solutions of problems for SMC. After this Sheikh et al worked on the real model of SMC. In which **they** proposed many protocols for secure sum computation. In these protocols they used random numbers for privacy of input data of individual parties.

## III. HYBRID TECHNIQUE OF SECURE MULTIPARTY PROTOCOL

In this protocol Hybrid model of secure multi-parties computation is proposed as shown in figure 1. In Hybrid model third party and individual parties both do computation partially at their end. In this protocol each party divides its data in three segments and with each segment parties adds different random number.

Steps:

1. Each party send its sum of first segment *D11, D21, D13,….Dn1* and random no. *r11, r21, r31…..rn1* to third party.

2. (i) Third party do sum of all the first segments received from all the parties *P1, P2, P3….Pn* i.e. *S*.
(ii) Third party send sum *S* to party *P1*.

3. Party *Pi* subtracts its random no. *ri1* and add its second segment *Di2* and its random no. *ri2* and then send sum to next party *Pi+1*. This step repeat till *i=n*.

4. Party *Pn* send sum *S* to *Pn-1*.

5. Party *Pn-1* subtracts its random no. *ri2* and add its third segment *Di3* and its random no. *ri3* and send sum to previous party *Pi-1*. This step repeat till *i=1*

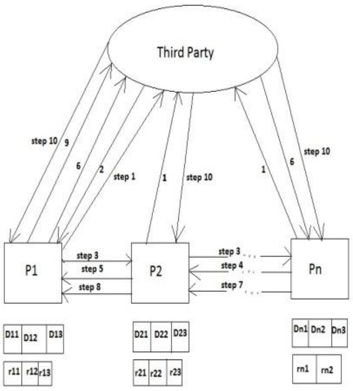

**Fig 1: Hybrid Secure Sum Architecture**

6. Party *P1* send sum *S* to *TP* and *TP* send this sum to *Pn*.

7. Party *Pn* subtracts its random no. *rn2* and add its third segment *Dn3* and send sum to *Pn-1*.

8. Party *Pi-1* subtracts its random no. *ri3* and send sum to *Pi-2*. Repeat this step till *i=1*.

9. Party *P1* sends sum *S* to *TP*.

10. Third party *TP* broadcast the sum *S* to *P1, P2,* P3,….Pn.

## IV. TOOLS: BASIC BUILDING BLOCKS

As building blocks for our algorithms, we use protocols for privacy-preserving computation of a minimum min($x$, $y$) and set union $S1 \cup S2$. In the minimum problem, the parties have as their respective private inputs integers $x1$ and $x2$ which are representable in $n$ bits. They wish to privately compute $m$ = min($x1$, $x2$). Because this problem is efficiently solved by a simple circuit containing $O(n)$ gates, it is a good candidate for Yao's generic method [1].

An implementation of this functionality with Yao's garbled circuit requires 2 communication rounds with $O(n)$ total communication complexity and $O(n)$
computational complexity.

**Privacy-Preserving Set Union**

In the set union problem, parties $P1$ and $P2$ have as their respective private inputs sets $S1$ and $S2$ drawn from some finite universe $U$. They wish to compute the set $S = S1 \text{ U } S2$ in a privacy-preserving manner, *i.e.*, without leaking which elements of $S$ is in the intersection $S1 \cap S2$. We will define $|S1| = s1$, $|S2| = s2$, $|S| = s$, and $|U| = u$. In this section, we give two solutions for privacy-preserving set union: the iterative method, and the tree-pruning method. Both require communication

and computational complexity that is logarithmic in $u$, provided $s$ is small (note that even if we are not concerned about privacy, computing the set union requires at least $O(s \log u)$ bandwidth, although it can be done in 1 round).

As a result, each element in $U$ is given an integer label with log $u$ bits. In addition, we need a label representing ∞, for which can simply use the integer $u$+1. The protocol proceeds as follows:

*Step 1*. Set $S = \Phi$.

*Step 2*. $P1$ selects $m1$ as the canonically smallest element in $S1$, or sets $m1 = \infty$

if $S1 = \Phi$. $P2$ likewise selects $m2$ as the canonically smallest element in $S2$, or

sets $m1 = \infty$ if $S1 = \Phi$.

*Step 3*. Using a protocol for private minimum, $P1$ and $P2$ privately compute $m$ = min($m1$,$m2$).

*Step 4*. If $m = \infty$, stop and return $S$. Otherwise, $S = S \text{ U } \{m\}$ and the parties remove $m$ from their input sets (it may be present in one or both). Then return to step 2.

The protocol preserves privacy because, given the output set $S$, a simulator can determine the value of $m$ at each iteration. The protocol used for computing the minimum is private, so there exists an efficient algorithm that can simulate its execution to the party $P1$ given its input and the output $m$ (likewise for $P2$). The simulator for the iterative method protocol uses the simulator for the minimum protocol as a subroutine, following the standard hybrid argument.

**The iterative method** protocol requires $s$ + 1 iterations, and in each iteration the minimum of two (log $u$)-bit integers is privately computed. Using Yao's method, this requires a circuit with 2 log $u$ inputs and $O(\log u)$ gates. The 2 log $u$ oblivious transfers can all take place in parallel, and since Yao's method requires a constant number of rounds the whole protocol takes $O(s)$ communication rounds. The total communication and computational complexity for the iterative method is $O(s \log u)$.

**Tree pruning method**. Before the tree pruning protocol begins, the participants agree on a (log $u$)-bit binary label for each element in the universe (note that a canonical total ordering would automatically provide such a label). The basic idea of the protocol is that the participants will consider label prefixes of increasing length, and use a privacy-preserving Bit-Or protocol to determine if either participant has an element with that prefix in his set. Initially, the single-bit prefixes "0" and "1" are set "live." The protocol proceeds through log $u$ rounds, starting with round 1. In the $i$th round, the participants consider the set P of $i$-bit "live" prefixes. For each prefix $p \epsilon P$, each participant sets his respective 1-bit input to 1 if he has an element in his set with prefix $p$, and to 0 if he does not have any such elements. The participants then execute a privacy-preserving Bit-Or protocol on their respective 1-bit inputs. The tree-pruning protocol preserves privacy because, given the output set $S$, a simulator can determine the output of each of the Bit-Or protocols. As in the case of the iterative method protocol, we can construct a simulator for the tree-pruning protocol that uses a simulator for the Bit-Or protocol as a subroutine, and prove its correctness using a hybrid argument. The construction is simple and is omitted for brevity. The tree-pruning protocol requires log $u$ iterations, and in each iteration the pair wise Bit-Or of at most $2s$ bits is computed. These computations can all take place in parallel, so the protocol requires $O(\log u)$ communication rounds. Each iteration requires $O(s)$ communication and computational complexity, so the entire protocol has complexity $O(s \log u)$. Both the iterative method and tree pruning protocols have the same complexity, but different numbers of rounds. The iterative method requires fewer rounds when $s = o(\log u)$.

## V. CONCLUSIONS

Secure Multi-Party Computation is a well researched. Quite a few protocols already exist, and work is going-on on another handful. These protocols for secure computation achieve remarkable results: it has been shown that generic constructions can be used to compute any function securely, and it has also been demonstrated that some functions can be computed even more efficiently using specialized constructions. Still, a secure protocol for computing a certain function will always be more costly than a naive protocol that does not provide any security. Our protocol also reduces the complexities that are encountered in three and four layer protocols. Subsequent enhancement of the protocol is expected the function domain is being further developed and the transforming functions that leverage the proposed architecture in different areas are being fine-tuned.

We believe that further research in this area is crucial for the development of secure and efficient protocols in this field. Of course, this must go hand in hand with research on privacy in general and the question of what information leakage is acceptable and what is not.

## REFERENCES

[1] Yao Andrew C., "Protocols for secure computations," *Proc. of 23$^{rd}$ Annual Symposium Foundations of Computer Science*, pp. 160-164.

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

[3] G. Aggarwal, N. Mishra and B. Pinkas. Secure Computation of the k-th Ranked Ele-ment. In *EUROCRYPT 2004*, Springer-Verlag (LNCS 3027), pages 40{55, 2004.

[4] W. Aiello, Y. Ishai and O. Reingold. Priced Oblivious Transfer: How to Sell Digital Goods. In *EUROCRYPT 2001*, Springer-Verlag (LNCS 2045), pages 119-135, 2001.

[5] Y. Aumann and Y. Lindell. Security Against Covert Adversaries: E±cient Protocols for Realistic Adversaries. In 4*th TCC*, Springer-Verlag (LNCS 4392), pages 137-156, 2007.

[6] D. Beaver. Foundations of Secure Interactive Computing. In *CRYPTO'91*, Springer-Verlag (LNCS 576), pages 377{391, 1991.

[7] D. Beaver, S. Micali and P. Rogaway. The Round Complexity of Secure Protocols. In 22*nd STOC*, pages 503{513, 1990.

[8] M. Bellare and S. Micali. Non-Interactive Oblivious Transfer and Applications. In *CRYPTO'89*, Springer-Verlag (LNCS 435), pages 547{557, 1989.

[9] W. Du, and M. J. Attalah, "Secure Multi – Party Computation Problems and Their Applications: A Review

[10] Open Problems." *Tech Report CERIAS Tech Report 2001-51*, Centre for Education and Research in Information Assurance and Security and Department of Computer Sciences, Purdue University, West Lafayette,IN 47906, 2001.

[11] J. Vaidya, and C. Clifton, "Leveraging the Multi in Secure Multi-Party Computation." *WPES'03 October 30, 2003*, Washington DC, USA, ACM Transaction 2003, pp 120- 128.

[12] M. J. Atallah and W. Du., "Secure Multi-Party Computation Geometry." *Seventh International Workshop on Algorithms and Data Structures (WADS 2001)*, 116 Providence, Phode Island, USA, Aug 8-10, 2001, pp 136-152.

[13] Jean-Sebastien Coron and_Ecole Normale Superieure, "Resistance Against Di_erential Power Analysis for Elliptic Curve Cryptosystems", [Published in C_ .K. Ko_c and C. Paar, Eds., Cryptographic Hardware and Embedded Systems, vol. 1717 of Lecture Notes in Computer Science, pp. 292 Springer-Verlag, 1999.]

[14] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, IT-22(6):644–654, 1976.

[15] S. Goldwasser and S. Micali. Probabilistic encryptionand how to play mental poker keeping secret all partial information. In *Proceedings of the 14th ACM Symposium on Theory of Computing (STOC'82)*, pages 365–377, San Francisco, CA, USA, May 1982.

[16] M. O. Rabin. Digitalized signatures and public-key functions as intractable as factorization. Technical Report LCS/TR-212, Massachusetts Institute of Technology, 1979.

[17] R. L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and publickey cryptosystems. *Communications of the ACM*, 21(2):120–126, February 1978.