Minimizing Storage and Communication Costs PSEASMC Model

[1]S.Samson Dinakaran, [2]Dr.M.Devapriya

[1]Assistant Professor, [2]Assistant Professor
[1]NDepartment of Computer Science,
[1]VLB Janakiammal College of Arts and Science, Coimbatore, Tamil Nadu, India

Abstract : Secure Multiparty Computation (SMC) is the most widely recognized security issue. The issue is if expect there is no Trustworthy Third Party (TTP) accessible to accumulate the data sources, compute the function and allocate the outcomes to every different party who could misconduct at all. To handle this issue, a smart card-based security model, specifically TrustedPals has been proposed and built-in with the failure detector and consensus in asynchronous network configuration. However, the Asynchronous SMC (ASMC) model uses fixed-size of payloads in fixed time frames which may influence the tradeoff among safety measures and execution. So, Performance and Security Enhanced ASMC (PSEASMC) model was suggested that improves the tradeoff objective function by means of both execution and safety measures together to modify the tradeoff among safety measures and execution. On the other hand, the consensus was not resolved because of high storage and communication costs. Hence in this paper, Sublinear communication and Quantum protocol in PSEASMC model (SQPSEASMC) is proposed. This model is built up against reactive deception of a greater part of the parties for any layered Boolean and Arithmetic Circuits. In this model, a One-Time Truth-Table (OTTT) protocol is used to enable various parties to commonly assess N-party functionality by sharing between Alice and Bob. This secret sharing procedure is accomplished depend on the oblivious transmit of a bit-string message from Alice to Bob. The oblivious transmit based on bit-string protocol uses hash functions to verify whether Bob receives the message or not. Besides, the functionality can end on the probability that it receives an erroneously organized message. This protocol can reduce both capacity and communication complexities with expanded protection from malicious adversaries. Finally, the experimental results show that the proposed SQPSEASMC model achieves better performance than the PSEASMC to solve the consensus by minimizing storage and communication cost.

***IndexTerms* - Secure multiparty computation, Failure detector, Consensus, PSEASMC, One-time truth-table.**

I. INTRODUCTION

In general, Multi-Party Computation (MPC) or called Secure Multiparty Computation (SMC) is a piece of cryptography with the objective of broadening systems for parties so as to commonly decide the function over their data sources while safeguarding those information sources. Other than the customary cryptographic strategies where cryptography ensures defence and reliability of transmission or storage space and the assailant is outside the arrangement of parties, the cryptography in the SMC model may keep the parties confidentiality from one another. Over the earlier periods, these SMC procedures have been quickly stretched out with reduced overheads. For example, contaminated circuit investigation has been accomplished by using SMC system at rates of 1.15 billion gates/seconds and secret sharing maintained privacy-preserving zone facilities were accessible at Real World Crypto 2015.

In spite of these consequences, SMC can't be used successfully in the greater part of the applications where real-time execution is required. Particularly, this is appropriate for techniques depend on fully homomorphic encryption or secret sharing. Let a lot of parties who need to properly decide some regular elements of their nearby information sources while saving their neighbourhood information as classified as realistic, yet who don't belief in one another, not the channels by which they commune. This is the extremely customary security issue in SMC i.e., it might be utilized to explain different real-time difficulties, for e.g., circulated determination, confidential request, signature sharing or unscrambling tasks and so on. Tragically, resolving SMC without any more concerns is exceedingly high expense in terms of the amount of messages to be transmitted, sum of redundancy and amount of synchronous rounds. To clarify these issues, TrustedPals [1] model has been structured which is a smart card-based security system that enables progressively profitable solutions for the SMC challenges. Hypothetically, this model comprises of a disseminated framework where functions were privately transmitted with tamper-proof security modules. In state-of-the-art procedures, the functions were implemented as a Java desktop application and also security modules were forecasted by Java Card Technology empowered smart cards [2], tamper-proof Subscriber Identity Modules (SIM) [3] like those used in smartphones or capacity gadgets with fundamental components. In this model, the function F is embedded as a Java function and is disseminated in the systems in an essential phase. At that point, functions offer their key value to their security module and the model accomplishes the protected dissemination of the key values. Finally, all security modules decide the function and return the outcome to their function. The system security modules initiate mystery and authentic channels between one another and act as a safe overlay in the dissemination stage.

Accordingly, this model permits the removal of the security issue in SMC to an issue of fault-tolerant synchronization. Additionally, the decrease from security to fault-tolerance provides a novel arrangement of verification requirements with respect to an integration of a fault-tolerant algorithm into a secure framework. This model and its execution with the synchronous system settings i.e., a design wherein all essential timing parameters of the system are limited and known. This makes TrustedPals vulnerable against the unexpected changes in the system delay and thus this is not appropriate for system execution. Cortinas et al. [4] investigated how to make TrustedPals relevant in a system design with less synchrony. As well, they demonstrated how to resolve the ASMC encouraged by the present consequences in fault-tolerant distributed computing. They used an asynchronous consensus algorithm and embody timing assumptions in a failure detector [5]. However, it produces predetermined size messages in fixed time frames. For that reason, the size of the payload was needed to decide to find a satisfactory tradeoff among safety measures and execution with the goal that a message size offers better security in cost of poor execution.

Therefore, an adaptive model for the tradeoff among safety measures and execution was proposed in the failure detector operation [6]. In this model, Performance Enhanced ASMC (PEASMC) and Security Enhanced ASMC (SEASMC) were proposed to quantify the execution and safety measures by improving the set of metrics. Too, Performance and Security Enhanced ASMC (PSEASMC) was proposed to compute the best tradeoff by diminshing the tradeoff objective function which has both performance and security metrics together rather than consequently changing with one security design then onto the next until the required tradeoff was accomplished. Along these lines, a satisfactory tradeoff among security and performance was accomplished depend on the elected payload size efficiently. On the other hand, the consensus in this model was not resolved because of high storage and communication endeavours. Additionally, the bit complexity of the payload was high because of the usage of unbounded buffers.

Hence in this article, SQPSEASMC model is proposed in which the correlated randomness model is utilized with polynomial storage and communication sublinear in the circuit size s for a large class of circuits. This protocol consists of two models such as layered Boolean and layered Arithmetic Circuits. Both models can perform against passive corruption of a majority of the parties by using the OTTT protocol that facilitates multiple parties to mutually evaluate an N-party functionality by sharing between Alice and Bob. Here, the secret sharing process is performed based on the oblivious transfer of a bit-string message from Alice to Bob. The oblivious transfer based on bit-string protocol uses hash functions to verify whether Bob receives the message or not. Also, the functionality can terminate if it receives any inaccurately formatted message. This protocol can reduce both storage and communication complexities with increased security against malicious adversaries. As a result, the storage and communication complexities in the trusted systems are minimized with the highest security against malicious opponents.

The remainder of the paper is composed as pursues: Section II investigates the past looks into on SMC for various purposes. Section III explains the proposed SQPSEASMC model. Section IV compares the investigational outcomes of the proposed and the existing models. Section V concludes the whole research work.

II. LITERATURE SURVEY

Boyle et al. [7] proposed an efficient method for securely and privately processing large datasets over multiple-parties with parallel distributed algorithms. Particularly, load-balanced statistically secure computation protocols were demonstrated to compute Parallel RAM (PRAM) programs, handle the malicious players when preserving up to polylogarithmic factors the computation, parallel time and memory complexities of PRAM program, aside from a one-time execution of a broadcast protocol per party. However, memory per party and the computation time were high.

Zyskind et al. [8] proposed Enigma i.e., decentralized computation platform with guaranteed privacy. This computational model was proposed based on a highly optimized version of SMC guaranteed by a verifiable secret-sharing scheme. For storage, a modified distributed hash table was used for holding secret-shared data. An external blockchain was used as the controller of the network that manages the access control, identifies and serves as a tamper-proof log of events. In this scheme, the requirement for a trusted third party was removed and autonomous control of personal data was enabled. However, the data storage was time-limited i.e., the data will be removed within a specific amount of time.

Li et al. [9] proposed a general framework, namely Preserving Multiparty Data Privacy (PMDP) in cloud computing. This framework can protect numeric data computing and publishing with the assistance of untrusted cloud servers and achieve delegation of storage simultaneously. This framework was constructed based on the different cryptographic primitives i.e., SMC and differential privacy mechanism that ensures its security against semi-honest participants without collision. However, the computational complexity of this framework was high.

Wang et al. [10] proposed a new constant round protocol for MPC of Boolean circuits to secure against an arbitrary number of malicious corruptions. In this protocol, an efficient preprocessing phase i.e., an optimized multiparty version of their TinyOT protocol was designed that facilitates the parties to create the authenticated information. However, the computational complexity and running time of this protocol were high.

Zhu et al. [11] proposed an extreme-scale actively-secure MPC named NANOPI to mitigate the problem of large-scale computations without significantly sacrificing performance. However, the computational and communication complexities of this protocol were high. Ishai et al. [12] studied the minimal number of point-to-point messages needed for general SMC in the setting of computational security against semi-honest, static adversaries who may corrupt an arbitrary number of parties. The aim of reducing the message complexity of protocol was encouraged by scenarios in which transmitting or receiving a message gas a high cost which is not very sensitive to the size of the message. However, the achievable security and required setup for extending the positive outcomes were essential to accommodate malicious adversaries.

Maltitz et al. [13] proposed a management and orchestration framework for SMC in dynamic environments. This framework can support the discovery of nodes, a trust establishment between them and realize robustness of SMC session by handling nodes failures and communication interruptions. However, the computational time complexity of this framework was high. Boyle et al. [14] studied the bottleneck complexity as a new communication efficiency measure for MPC. The main outcome of this study was a compiler that transforms any efficient protocol with a determined transmission pattern to compute any functionality into a secure MPC protocol when preserving the bottleneck complexity of the underlying protocol. However, it requires a variable transmission pattern to further reduce the computational complexity.

III. PROPOSED METHODOLOGY

3.1 One-Time Truth Table (OTTT) Protocol

The main core of this SQPSEASMC protocol is the OTTT which facilitates multiple parties for mutually estimating the function $f: X_1 \times X_2 \times \ldots \times X_N \longmapsto Z$ by distributing between each party a tangled version of the truth table of f. The OTTT protocol to estimate an arbitrary N-party functionality f in the correlated randomness model against a reactively corrupted majority is given in below:

OTTT Protocol:

Functionality:

- ✓ Public parameters: an N-party functionality $f: X_1 \times X_2 \times \ldots \times X_N \longmapsto Z$, where $(X_i, +)$ and $(Z, +)$ are groups.
- ✓ The parties $(P_1, \ldots, P_N)$ hold particular inputs $x = (x_1, \ldots, x_N)$;

- ✓ Output: Every party P_i learn $z = f(x)$.

Preprocessing:

- ➢ Sample $(r_1, \dots, r_N) \xleftarrow{\$} X_1 \times X_2 \times \dots \times X_N$.
- ➢ Consider M is the truth-table of f permuted with the shifts r i.e., for any $x \in X_1 \times X_2 \times \dots \times X_N$, $M|_{x+r} = f(x)$.
- ➢ Assume $(M_i)_{i \le N}$ is a random secret sharing of M.
- ➢ Obtain (r_i, M_i) to every party P_i.

Protocol(x):

- ➢ Every party P_i with input x_i transmits $u_i \leftarrow x_i + r_i$.
- ➢ Every party P_i transmits $z_i \leftarrow M_i|_u$.
- ➢ All parties reconstruct $z \leftarrow \sum_{i=1}^{N} z_i$.

Consider N parties communicating over asynchronous and authenticated broadcast channel. An N-party functionality $F: X_1 \times X_2 \times \dots \times X_n \longmapsto Z_1 \times Z_2 \times \dots \times Z_N$ denotes a mapping from the N input of each party to N outputs. These functionalities represent non-reactive computation processes. Randomized N-party functionalities are secret sharing functionalities for functions over an abelian group $(\mathbb{G}, +)$: a protocol calculates secret shares of a function $g: \mathbb{G} \longmapsto \mathbb{G}$ if it calculates the randomized N-party functionality on input $(x_1, \dots, x_N) \in \mathbb{G}^N$, outputs N uniformly random group elements $(z_1, \dots, z_N) \in \mathbb{G}^N$ subject to $\sum_{i=1}^{N} z_i = g(\sum_{i=1}^{N} x_i)$. This represents the condition where the parties hold secret shares of an input to the deterministic function and need to receive secret shares of the output of the function.

For any N-party functionalities f represented by a layered Boolean or arithmetic circuit C of size s with n inputs and m outputs and for any integer k, there is an absolutely secure protocol which realizes f in the preprocessing model against semi-honest parties without honest majority with communication $n + N \cdot \left(m + \lceil s/k \rceil\right)$ and storage $n/N + \left(m + \lceil s/k \rceil\right) \cdot \left(2^{2^k} + 1\right)$. In this protocol, storage is referred to as the number of correlated random coins stored by each party at the end of the preprocessing phase.

The protocol for the semi-honest model with correlated randomness is given below:

Δ_L Protocol:

Functionality:

- ✓ Public parameters: c-local function $g: \mathbb{F}_2^n \longmapsto \mathbb{F}_2^m$ and m size-c subsets $S_j \subset [n]$ of the bits in the input on which j^{th} output bit of g depend.
- ✓ Input: the parties $(P_1, \dots, P_N)$ hold random shares $(x_1, \dots, x_N)$ of an input x over $\mathbb{F}_2^n$;
- ✓ Output: the parties provide equally random shares of $g(x)$.

$\Delta_L.Preprocessing(g)$:

- ➢ Sample $(r_1, \dots, r_N) \xleftarrow{\$} \mathbb{F}^n \times \dots \times \mathbb{F}^n$. Assign $r \leftarrow \sum_{i=1}^{N} r_i$.
- ➢ $for(j = 1\ to\ m)$
- ➢ Consider $g_j \leftarrow limit(g, j)$.
- ➢ Consider M_j is the truth-table of g_j permuted with the shift $r[S_j]$ i.e., for any $y \in \mathbb{F}_2^c$, $M_j|_{y+r[S_j]} = g_j(y)$. //M_j is of size 2^c.
- ➢ Assume $\left(M_j^i\right)_{i \le N, j \le m}$ is random secret sharing of M_j.
- ➢ $for(i = 1, \dots, N)$
- ➢ Obtain $\left(r_i, \left(M_j^i\right)_{j \le m}\right)$ to each party P_i.

$\Delta_L.Protocol(g, x)$:

- ➢ Every party P_i with share x_i transmits $u_i \leftarrow x_i + r_i$. Consider $u \leftarrow \sum_i u_i$.
- ➢ $for(j = 1\ to\ m)$
- ➢ Every party P_i provides $z_{i,j} \leftarrow M_j^i|_{u[S_j]}$.

3.2 Sublinear Protocol for Layered Boolean Circuits

A basic PSEASMC model is proposed in the correlated randomness model against reactive fraud of a majority of the parties for any layered Boolean circuit with sublinear communication in the circuit size s. Consider C is a layered Boolean circuit with n inputs and m outputs of size s and depth $d = d(n)$ with layers $(L_1, \dots, L_d)$. For $i = 1, \dots, d$, consider w_i denotes the width of the layer L_i and k is an integer. The circuit C is split into $d' = \lceil d/k \rceil$ chunks $(ch_i)_{i \le d'}$, each chunk has k consecutive layers. Consider $t \in [k]$ is selected so that the sum of the widths of tth layer of each chunk is bounded by $\lceil s/k \rceil$. For $i = 1, \dots, d'$, t_i is the index of tth layer in ch_i; it holds that $\sum_{i=1}^{d'} w_{t_i} \le \lceil s/k \rceil$.

For $i = 1, \dots, d'$, assume m_i is the number of output nodes between the layers $L_{t_{i-1}}$ and L_{t_i} $(\sum_i m_i = m)$. For any $i \le d'$ and $j \le w_{t_i} + m_i$, $n_{i,j}$ is denoted as the jth node of the layer $L_{t_i} \in ch_i$ if $j \le w$ and the $(j - w)^{th}$ output node between the layers $L_{t_{i-1}}$ and L_{t_i}. Associate two sets to each $n_{i,j}$: assume $A_{i,j}$ is the set of ancestors of $n_{i,j}$ which belong to $L_{t_{i-1}}$ and $I_{i,j}$ is the set of input nodes between the layers $L_{t_{i-1}}$ and L_{t_i} which are ancestors of $n_{i,j}$. Consider $\alpha_{i,j}$ (resp. $\tau_{i,j}$) is the size of the set $A_{i,j}$ (resp. $I_{i,j}$).

The proposed SQPSEASMC model is initiated by evaluating C on an input x in a chunk-by-chunk manner. The parties can estimate a chunk i while they calculate all the values related to the nodes of the layer L_{t_i} including all the values related to the output nodes between the layers $L_{t_{i-1}}$ and L_{t_i}. Every chunk can be estimated during a round. The bit-string of the shares of the values on L_{t_i} denoted as $y_{i,l}$ is computed by the party P_l in the ith round and the reconstructed value as $y_i = \oplus_{l=1}^{N} y_{i,l}$. Likewise, the bit-string of the shares of the values on the output links between $L_{t_{i-1}}$ and L_{t_i} calculated by the party P_l in the ith round and the reconstructed output string as $z_i = \oplus_{l=1}^{N} z_{i,l}$. For any $i \le d'$ and $j \le w_{t_i} + m_i$, consider $f_{i,j}$ is the following function: on input, the sub-string $x[I_{i,j}]$ of the input string x and the bit-string $y_{i-1}[A_{i,j}]$, $f_{i,j}$ outputs the value related to the node $n_{i,j}$. Consider $\delta_i \leftarrow w_{t_i} + m_i$ is the number of functions $f_{i,j}$ for predefined i. At last, the following function is denoted by $f_i: \mathbb{F}_2^{w_{t_i}+n} \longmapsto \mathbb{F}_2^{\delta_i}$: on input the string y_{i-1} related to the different layer of $(i-1)^{th}$ chunk and the input string x, f_i outputs $\left(f_{i,j}(x[I_{i,j}], y_{i-1}[A_{i,j}])\right)_{j \le \delta_i} = (y_i, z_i)$. By constructing this protocol, it is observed that f_i is a 2^k-local function.

Δ_B Protocol:

Functionality:

- ✓ Public parameters: a layered Boolean circuit C of size s and depth d with n input gates and m output gates and an integer k
- ✓ The parties $(P_1, \dots, P_N)$ hold particular inputs $x = (x_1, \dots, x_N)$ of length n/N;
- ✓ Output: all the parties learn $C(x)$.

$\Delta_B.Preprocessing(C)$: $for\left(i = 1\ to\ d' = \left\lceil d/k \right\rceil\right)$

- ➢ Perform $\Delta_L.Preprocessing(f_i)$.

$\Delta_B.Protocol(C, x)$:

- ➢ $for(i = 1\ to\ d')$
- ➢ All parties carry out $\Delta_L.Protocol(f_i, (y_{i-1}, x))$.
- ➢ Every party P_l acquires output $(y_{i,l}, z_{i,l})$.
- ➢ Output: all the parties transmit $z_{i,l}$.
- ➢ All the parties reconstruct the output $z = \left(\sum_l z_{i,l}\right)_{i \le d'}$.

3.3 Sublinear Protocol for Layered Arithmetic Circuits

The proposed SQPSEASMC protocol model can be extended to the arithmetic circuits by demonstrating the normal analog of the OTTT protocol, adapted to arithmetic functions. This protocol consists of the following features: Alice holds an n-variate polynomial P of degree θ, Bob holds a vector of input $x \in \mathbb{F}^n$ and both parties share a common random string. They transmit a single synchronized message to a third party with optimal communication. This facilitates the third party to learn $P(x)$. The process of this protocol is given below:

- The shared randomness is $r \in \mathbb{F}^n$ and a random n-variate polynomial P of degree θ.
- Alice transmits $(x', u) \leftarrow (x + r, R(x + r))$.
- Bob transmits the polynomial $Q(X) = P(X - r) + R(X)$.
- The third party outputs $Q(x') - u$.

Δ_A Protocol:

Functionality:

- ✓ Public parameters: an arithmetic function $f: \mathbb{F}^n \longmapsto \mathbb{F}^m$ of depth d over a finite field $\mathbb{F}$ and m size-2^d subsets $S_j \in [n]$ of the coordinates of the input on which j^{th} coordinate of the output of f depends.
- ✓ The parties $(P_1, \dots, P_N)$ hold additive shares $x = (x_1, \dots, x_N)$ of an input $x \in \mathbb{F}^n$;
- ✓ Output: the parties provide equally random shares of $f(x)$.

Preprocessing:

- ➢ Sample $(r_1, \dots, r_N) \xleftarrow{\$} \mathbb{F}^n \times \dots \times \mathbb{F}^n$. Assign $r \leftarrow \sum_{i=1}^{N} r_i$.
- ➢ $for(j = 1\ to\ m)$
- ➢ Consider $f_j \leftarrow limit(f, j)$.
- ➢ Consider $P_j(X)$ is the normal-form of f_j observed as a 2^d-variate polynomial of degree 2^d over $\mathbb{F}$.
- ➢ Assume $Q_j(X) \leftarrow P(X - r[S_j])$ is the polynomial P shifted with $r[S_j]$.
- ➢ Sample $N - 1$ equally random degree-2^d n-variate polynomials $\left(R_j^i(X)\right)_{i \le N-1}$ and assign $R_j^N(X) \leftarrow Q_j(X) + \sum_{i=1}^{N-1} R_j^i(X)$.
- ➢ Output: $\left(r_i, \left(R_j^i(X)\right)\right)_{j \le m}$ to each party P_i.

Protocol(x):

- ➢ Every party P_i with share x_i transmits $u_i \leftarrow x_i + r_i$. Consider $u \leftarrow \sum_i u_i$.
- ➢ $for(j = 1\ to\ m)$
- ➢ Every party P_i provides $z_{i,j} \leftarrow R_j^i(u[S_j])$.

3.4 Secure Computation of Secret Shares using Oblivious Transfer Protocol

This protocol can achieve an oblivious transfer of a bit-string message from Alice to Bob i.e., ideal functionality for the secure computation of secret shares on an input $x \in \mathbb{F}_2^n$ shared between N parties and improve the security against the semi-honest corruption of a majority of the parties. The oblivious transfer protocol has two different phases such as transmitting phase and opening phase. In the transmitting phase, Alice transmits encoded secret information to all other parties. In the opening phase, Alice reveals adequate data so that all other parties can decode the secret with probability $\frac{1}{2}$.

In this protocol, the hash function is used after the opening phase to verify whether all parties receive the message or not. A hash function generates a digest of a message i.e., a string of smaller size such that the chance of creating random strings with similar hash value is insignificant and the hash values are almost uniformly distributed over the set of all potential digests. Given a computational basis $\beta_0 = \{|0\rangle, |1\rangle\}$, initially, Alice encodes each bit mes_i of the message $mes = mes_1, \dots, mes_l$ into the state $|mes_i\rangle$ of the related $q$bit. After that, he/she randomly selects a bit value α and for each mes_i, a rotation angle φ_i and rotates $|m_i\rangle$ by $(-1)^\alpha \varphi_i$. In the transmitting phase, he/she transmits the $q$bits to each other parties. For each $q$bit i, the encoding quantum states are as:

$$|0_i^{(\alpha)}\rangle = R((-1)^\alpha \varphi_i)|0\rangle \quad (1)$$

$$|1_i^{(\alpha)}\rangle = R((-1)^\alpha \varphi_i)|1\rangle = R(\pi)|0_i^{(\alpha)}\rangle \quad (2)$$

In the above equations, $R(\varphi)$ is defined by $R(\varphi)|0\rangle = cos(\varphi/2)|0\rangle + isin(\varphi/2)|1\rangle$ which is mutually orthogonal and so completely noticeable, provided that the path α and the angle φ_i of the rotation are evaluated. Hence, Bob or all N parties cannot decipher the message mes unless given additional data about the encoding bases $\beta_i = \left\{|0_i^{(\alpha)}\rangle, |1_i^{(\alpha)}\rangle\right\}$.

In the opening phase, Alice provides Bob with partial data: he/she transmits the so-called secret key, a string of rotation angles $\varphi = (\varphi_1, \dots, \varphi_l)$, however not the rotation path α. Oblivious to the rotation path, Bob can only estimate it which he/she will receive perfectly in 50% of the cases. Encrypted in quantum states of $q$bit, Alice transmits the message mes combined with its digest $dig = h(mes)$, given by a properly selected hash function $h: \{0,1\}^l \to \{0,1\}^\omega$ with $\omega = \lfloor\sqrt{l}\rfloor$. Depending on the decrypting the states of $q$bits transmitted by Alice, Bob reconstructs the string which is a concatenation of the form $mes' dig'$ where mes' and dig' are not essentially the message and its hash value $dig = h(mes)$. Bob verifies $dig' = h(mes')$. If this case is true, then he/she is encouraged that the received message mes' is really the intended message mes.

Bit-String Oblivious Transfer Protocol:

Security Parameter: $l, \eta \in \mathbb{N}$ with $\theta_\eta = \pi/2^{\eta-1}$;

Message to Transmit: $mes = m_1, \dots, m_l$;

Hash Function: $h: \{0,1\}^l \to \{0,1\}^\omega$ with $\omega = \lfloor\sqrt{l}\rfloor$;

Secret Key: $s = (s_1, \dots, s_{l+\omega})$ where each $s_i \in \{0, \dots, 2^\eta - 1\}$.

<u>Transmitting Phase:</u>

- Alice selects the hash function h and a bit $\alpha \in \{0,1\}$ randomly and arranges the following state with $dig = h(mes)$:

$$|\psi\rangle = \bigotimes_{i=1}^{l} R(mes_i\pi + (-1)^\alpha \times s_i\theta_\eta)|0\rangle \bigotimes_{i=1}^{\omega} R(dig_i\pi + (-1)^\alpha \times s_{i+l}\theta_\eta)|0\rangle \quad (3)$$

$$= \left(\bigotimes_{i=1}^{l} \left[\cos\left(\frac{mes_i\pi + (-1)^\alpha \times s_i\theta_\eta}{2}\right)|0\rangle + sin\left(\frac{mes_i\pi + (-1)^\alpha \times s_i\theta_\eta}{2}\right)|1\rangle\right]\right) \otimes \left(\bigotimes_{i=1}^{\omega} \left[cos\left(\frac{dig_i\pi + (-1)^\alpha \times s_{i+l}\theta_\eta}{2}\right)|0\rangle + sin\left(\frac{h_i\pi + (-1)^\alpha \times s_{i+l}\theta_\eta}{2}\right)|1\rangle\right]\right) \quad (4)$$

- Alice transmits the state $|\psi\rangle$ to Bob.

<u>Opening Phase:</u>

- Alice transmits the secret key $s = (s_1, \dots, s_{l+\omega})$ and the security parameter η to Bob.
- Bob verifies if s is expected to be a possible output of a random process.
- Bob selects $\alpha' \in \{0,1\}$ randomly and applies $R((-1)^\alpha s_i\theta_\eta)$ to each $q$bit of $|\psi\rangle$.
- Bob applies the measurement operator $M^{\otimes(l+\omega)} = (0 \times |0\rangle\langle 0| + 1 \times |1\rangle\langle 1|)^{\otimes(l+\omega)}$.
- Consider $mes' dig'$ is the message that Bob reconstructs. He/she verifies $dig' = h(mes')$.
- If yes, then Bob is approximately convinced that $mes' = mes$; otherwise, he/she identifies that mes' is an incorrect message.

IV. RESULTS AND DISCUSSIONS

In this part, the experimental results of the proposed SQPSEASMC model is presented and evaluated with the existing PSEASMC model by using Java. The assessment is prepared in terms of differential privacy, latency and accuracy which are portrayed beneath:

- **Differential Privacy:** Confidentiality is estimated by differential privacy. The required confidentiality level of the ith party is meant as ε_i. An algorithm A is ε_i-differentially secret for the ith party if for $i \in [k]$ and all $x_i, x_i' \in \{0,1\}, x_{-i} \in \{0,1\}^{k-1}$ and $\tau \in \mathcal{T}$ as:

$$\mathbb{P}(\tau|x_i, x_{-i}) \le e^{\varepsilon_i}\mathbb{P}(\tau|x_i', x_{-i}) \quad ()$$

This condition promises no opponent can understand the secrecy information x_i with high satisfactory certainty. On the chance that the algorithm is ε_i-differentially secrecy for all $i \in [k]$, in that case it is said that the protocol is $\{\varepsilon_i\}$-differentially secrecy for all parties.

- **Accuracy Measure:** For the ith party, let an accuracy measure $w_i: \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}$ with the goal that $w_i\left(f_i(x), \hat{f}_i(\tau, x_i)\right)$ measures the exactness when the function to be figured is $f_i(x)$ and the estimation is $\hat{f}_i(\tau, x_i)$. At that moment, the average accuracy for this i^{th} party is characterized as:

$$Acc_{avg}\left(P, w_i, f_i, \hat{f}_i\right) = \frac{1}{2^k} \sum_{x \in \{0,1\}^k} \mathbb{E}_{\hat{f}_i, P_{x,\tau}}\left[w_i\left(f_i(x), \hat{f}_i(\tau, x_i)\right)\right] \quad ()$$

Where $\mathbb{E}$ indicates the probability which is assumed over the arbitrary transcript τ sharing as P and furthermore any uncertainty in the decision function $\hat{f}_i$. In the same way, the worst-case accuracy is characterized as:

$$Acc_{wc}\left(P, w_i, f_i, \hat{f}_i\right) = \min_{x \in \{0,1\}^k} \mathbb{E}_{\hat{f}_i, P_{x,\tau}}\left[w_i\left(f_i(x), \hat{f}_i(\tau, x_i)\right)\right] \quad ()$$

- **Latency:** Latency is characterized as the amount of rounds needed to interface all legitimate parties with high likelihood after at most $\lceil log\lceil (t+1)/32 \rceil \rceil + 1$ iterations of the while loop.

Table 1 gives the performance comparison of proposed SQPSEASMC and existing PSEASMC regarding differential privacy, average accuracy, worst-case accuracy and latency.

Table.1 Comparison of Performance Metrics

	No. of Corrupt Parties	**PSEASMC**	**SQPSEASMC**
Differential Privacy	25	0.60	0.65
	50	0.64	0.69
	75	0.67	0.73
	100	0.69	0.77
	125	0.71	0.81
	150	0.73	0.85
	No. of Corrupt Parties	**PSEASMC**	**SQPSEASMC**
Average Accuracy (%)	25	75.4	78.3
	50	76.7	80.5
	75	78.4	82.9
	100	79.5	85.1
	125	80.9	87.6
	150	82.0	89.4
	No. of Corrupt Parties	**PSEASMC**	**SQPSEASMC**
Worst-case Accuracy (%)	25	56.0	58.2
	50	57.0	60.1
	75	58.2	62.3
	100	59.4	64.7
	125	60.7	66.5
	150	62.5	68.9
	No. of Corrupt Parties	**PSEASMC**	**SQPSEASMC**
Latency (Number of Rounds)	25	12	10
	50	14	12
	75	16	14
	100	18	16
	125	20	18
	150	22	20

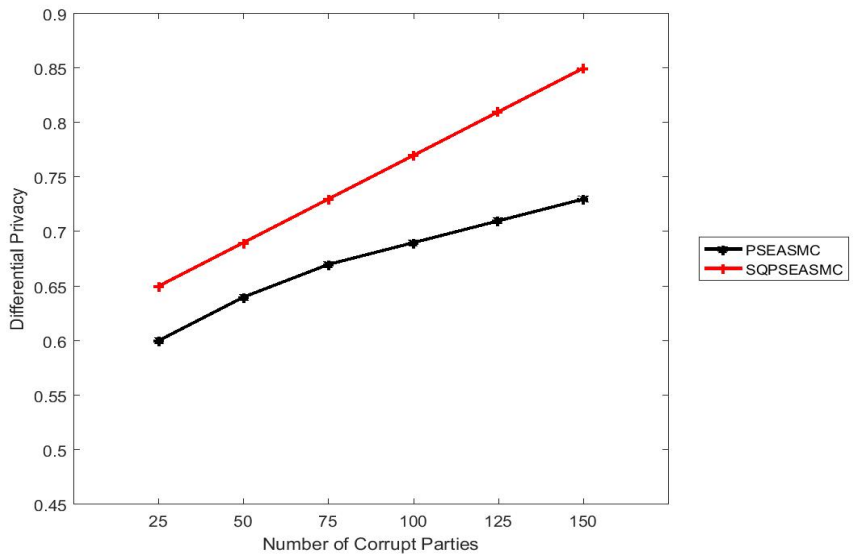

Figure.1 Comparison of Differential Privacy

Figure 1 exhibits the assessment of SQPSEASMC and PSEASMC in terms of differential privacy. For instance, consider the number of fraudulent parties is 150. At that point, the differential privacy for SQPSEASMC is 16.44% boosted than PSEASMC.

From this investigation, it is shown that the SQPSEASMC model can increase the secrecy for all N parties than the PSEASMC model.

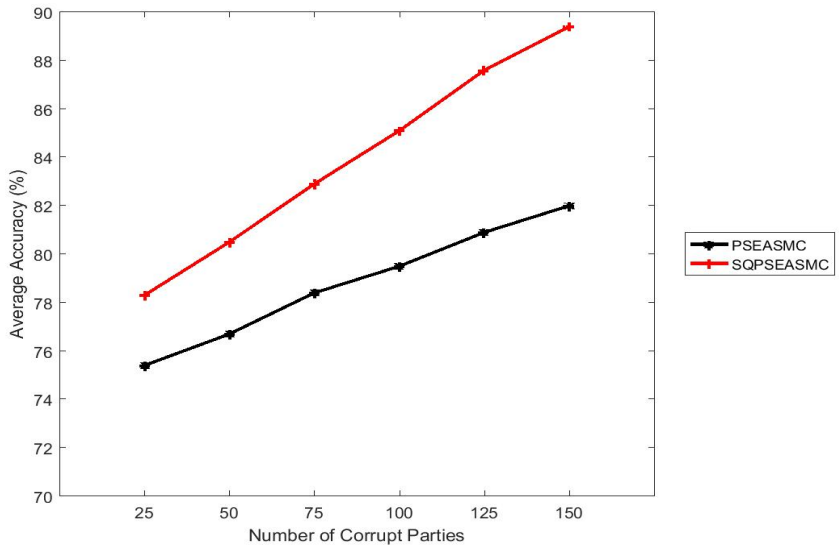

Figure.2 Comparison of Average Accuracy

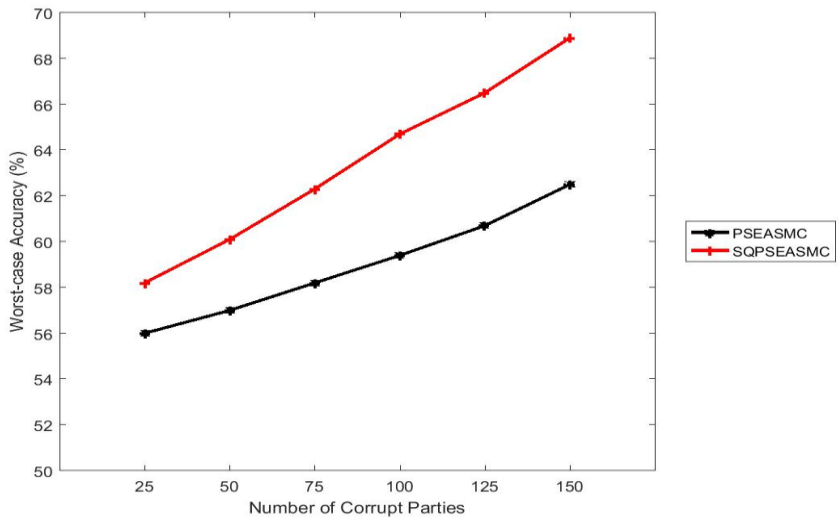

Figure.3 Comparison of Worst-case Accuracy

Figure 2 and 3 represent the assessment of average and worst-case accuracy for proposed and existing models, correspondingly. On the chance that the amount of dishonest parties is 150, in that case the average accuracy of proposed SQPSEASMC model is 9.02% increments than the PSEASMC model. Also, the worst-case accuracy of SQPSEASMC is 10.24% higher than PSEASMC model. Along these lines, it is inferred that the proposed SQPSEASMC model accomplishes higher exactness in both average and worst-case scenario for securely computing secret shares of the N-party functionality.

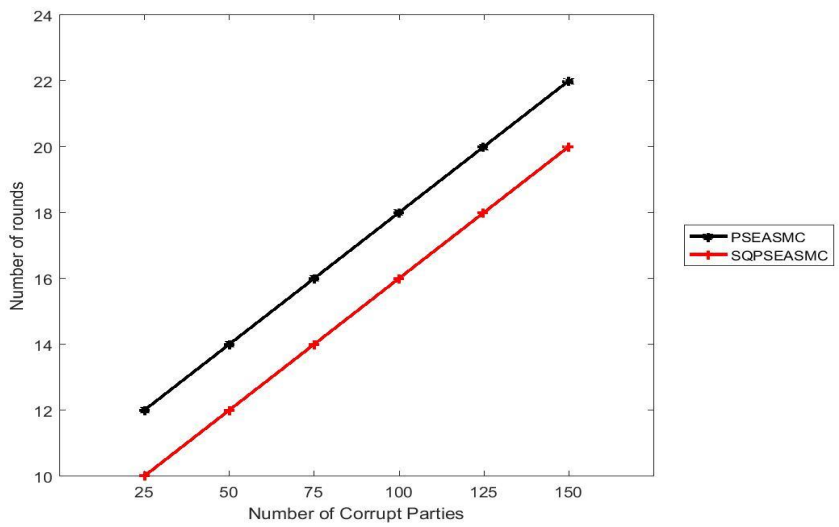

Figure.4 Comparison of Latency

Figure 4 demonstrates the assessment of SQPSEASMC and PSEASMC regarding latency. For instance, consider the amount of fraudulent parties is 150. For this condition, the number of rounds (latency) of proposed SQPSEASMC model is 9.09% reduced

than the PSEASMC model. In this way, it is seen that the proposed SQPSEASMC model significantly limits the number of rounds to interface all genuine parties with maximum secrecy.

V. CONCLUSION

In this article, the SQPSEASMC model is proposed in which the correlated randomness model is considered by means of polynomial storage and communication sublinear in the circuit size s for a large class of circuits. In this SQPSEASMC model, two sublinear models such as layered Boolean and layered Arithmetic circuits are designed that can execute against passive corruption of a majority of the parties by using OTTT protocol. Mainly, the OTTT protocol allows N parties to equally compute an N-party functionality by secret shares between Alice and Bob. The computation of secret shares between N parties is achieved by using the oblivious transfer based bit-string protocol which utilizes hash functions to authenticate whether Bob receives the correct message or not. If it receives an incorrect message, then N-party functionality can be terminated. By using this model, it is noticed that both storage and computational complexities are efficiently reduced. As well, the security against a semi-honest corruption of a majority of the parties is significantly increased. Finally, the experimental results proved that the proposed SQPSEASMC model has maximum differential privacy, average and worst-case accuracy and minimum latency than the existing PSEASMC model.

REFERENCES

1. Fort, M., Freiling, F., Penso, L. D., Benenson, Z., & Kesdogan, D. (2006, September). TrustedPals: Secure multiparty computation implemented with smart cards. In *European Symposium on Research in Computer Security* (pp. 34-48). Springer, Berlin, Heidelberg.
2. Keersebilck, P. (2004). Smart card technology based on java. In *7th International Conference on Development and Application Systems* (pp. 398-402).
3. Leavitt, N. (2005). Will proposed standard make mobile phones more secure?. *Computer*, *38*(12), 20-22.
4. Cortinas, R., Freiling, F. C., Ghajar-Azadanlou, M., Lafuente, A., Larrea, M., Penso, L. D., & Soraluze, I. (2012). Secure failure detection and consensus in trustedpals. *IEEE Transactions on Dependable and Secure Computing*, *9*(4), 610-625.
5. Atif, M. (2011). Formal modeling and verification of distributed failure detectors. *Faculty of Mathematics and Computer Science, TU/e*, *10*.
6. Dinakaran, S. S., & Devapriya, M. (2019). Security and performance enhanced asynchronous secure multiparty computation (ASMC). *International Journal of Engineering Research & Technology*, *8*(06), 724-730.
7. Boyle, E., Chung, K. M., & Pass, R. (2015, August). Large-scale secure computation: Multi-party computation for (parallel) RAM programs. In *Annual Cryptology Conference* (pp. 742-762). Springer, Berlin, Heidelberg.
8. Zyskind, G., Nathan, O., & Pentland, A. (2015). Enigma: Decentralized computation platform with guaranteed privacy. *arXiv preprint arXiv:1506.03471*.
9. Li, J., Wei, J., Liu, W., & Hu, X. (2017). PMDP: A Framework for Preserving Multiparty Data Privacy in Cloud Computing. *Security and Communication Networks, 2017*.
10. Wang, X., Ranellucci, S., & Katz, J. (2017). Global-scale secure multiparty computation. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security* (pp. 39-56). ACM.
11. Zhu, R., Cassel, D., Sabry, A., & Huang, Y. (2018). NANOPI: Extreme-Scale Actively-Secure Multi-Party Computation. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security* (pp. 862-879). ACM.
12. Ishai, Y., Mittal, M., & Ostrovsky, R. (2018, March). On the message complexity of secure multiparty computation. In *IACR International Workshop on Public Key Cryptography* (pp. 698-711). Springer, Cham.
13. von Maltitz, M., Smarzly, S., Kinkelin, H., & Carle, G. (2018, April). A management framework for secure multiparty computation in dynamic environments. In *NOMS 2018-2018 IEEE/IFIP Network Operations and Management Symposium* (pp. 1-7). IEEE.
14. Boyle, E., Jain, A., Prabhakaran, M., & Yu, C. H. (2018). The Bottleneck Complexity of Secure Multiparty Computation. In *45th International Colloquium on Automata, Languages, and Programming (ICALP 2018)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.