CHAPTER 5

SECURE MULTI-PARTY COMPUTATIONS USING ENCRYPTION

The scenario of the world is changing, people are more interested toward outsourcing work to a third party rather than maintaining their own resources, in this circumstance there is a requirement of insuring security from the service provider as it may lead to security breaches and party may not be interested in such service providers. Secondly some time organizations or individuals are interested in getting joint result of computation for their growth and analysis of future prospective of the field then they need to have SMC mechanisms which ensure the correct computation with the privacy of individual input.

In different real life situation such as voting, medical analysis, data mining etc., SMC is necessary to perform a joint computation over private data. In recent years much attention has been given to the computation security, as there is enormous growth in cloud computing, organizations are migrating from on premise infrastructure towards cloud environment.

This chapter presents a protocol for secure sum problem using asymmetric encryption protocol (*JCAE*) to maintain confidentiality during secure multi-party computations.

5.1 INTRODUCTION

Confidentiality of individual participants is very essential in various real life applications such as medical science and financial analysis. This protocol focuses on implementation of an asymmetric secure sum computation protocol using anonymization and public-key encryption

where all parties have access to TTP who (1) doesn't add any contribution to computation (2) doesn't know who is the owner of the input received (3) has large number of resources (4) decryption key is known to TTP to get the actual input for computation of final result. In this environment, concern is to design a protocol which deploys TTP for computation. It is demonstrated that the protocol is very proficient (in terms of secure computation and individual privacy) for the parties than the other available protocols. The solution incorporates protocol using asymmetric encryption scheme where any party can encrypt a message with the public key but decryption can be done only by the possessor of the decryption key (private key). As the protocol works on asymmetric encryption and packetization it ensures following: (1) Confidentiality (from unauthorized data access) (2) Security (3) Anonymity (identity).

The basic concept of SMC was initiated by Yao's (1986). Yao's (Yao, 1982) proposed two party knowledge exchange tool using cryptography, they have shown how two parties can generate a random number $R = PcQ$ such that prime no's P, Q cannot be obtained by individual party but it can be recovered jointly if needed. Goldreich *et al.* (1987) extended the problem proposed by (Yao, 1982) and presented a polynomial time algorithm to solve mental game problem, provided the majority of participants are honest. Rabin *et al.* (1981) proposed protocol work accurately if 2/3 of the parties are honest.

Cloud computing is the field where heterogeneous infrastructure is judiciously used. In cloud computing a computationally powerful service provider provides access to clients. This research work discusses the concerns of secure computation in an environment where along with the parties; there is a TTP. This environment is considered in the proposed

protocol design. It reduces the computation at parties' side at the expense of the TTP.

The protocol (*JCAE*) will demonstrate SMC is not possible in case of dishonest majority. The protocol is based on ideal model of SMC where a TTP is assumed for collaborative computation.

5.2 SECURITY MECHANISM

SMC very often uses the concept of software agents, which could be the participating parties, anonymizers and TTP. This needs to develop a trust model between the agents, so that the entire participant can freely share their sensitive information for collaborative computation and better research outcomes. Here, the security mechanism used has various layers with predefined functionality.

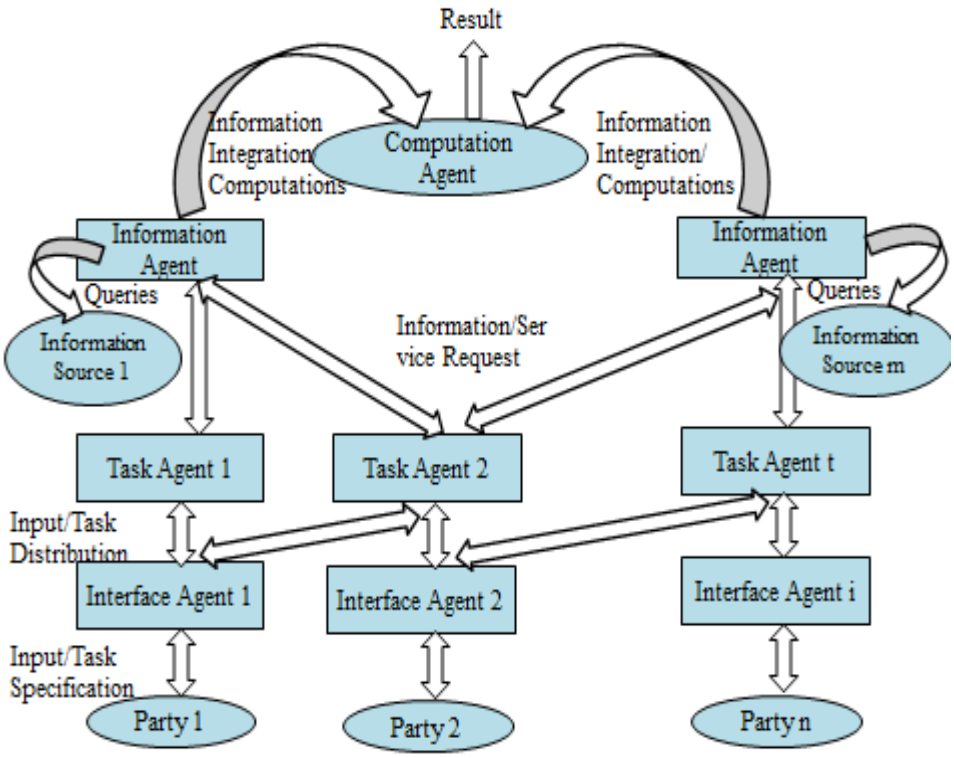

Figure 5.1: Secure Multi-Party Computation Framework

Figure 5.1 shows that the SMC is fundamentally divided into different layer and each layer has different agents to perform predefined functions. At the input layer parties provides their sensitive input for collaborative computations through some interface, for simulation ASP dotnet web service are used, which act as an interface to carry data from one layer to other layer. The top most layer is computation layer with a computation agent to perform final joint computations. In the protocols windows communication foundation (WCF) web services are used for computation purpose.

5.3 SECURE MULTI-PARTY COMPUTATIONS PROTOCOL USING ENCRYPTION TECHNIQUE

5.3.1 ARCHITECTURE OF *JCAE* PROTOCOL

The *JCAE* protocol architecture *Figure 5.2* is divided into multiple layers. Each layer has a predefined functionality. All the participant first register with the system as soon the party is registered a web service will be initiated, it pick a public-private key pair from the key pool and allot the public key to the party and forward the private key to TTP, number of packets and number of anonymizers is decided at the time of first party registration. Input can be provided from multiple terminals or can be taken from single terminal by different parties who are interested in collaborative computations. The bottom most layer is input or data layer through which input for computation is provided from various participants.

Once the registration process is completed the parties divide their private data into packets and encrypt it with the public key. After this parties' forwards packets to randomly selected anonymizer. Anonymization layer is to hide the identity of parties from TTP. Here, anonymizers' job is to collect and forward data packets without keeping

record of any input providers. Computation layer's functionality is to decrypt and perform collaborative computation after receiving all the packets.

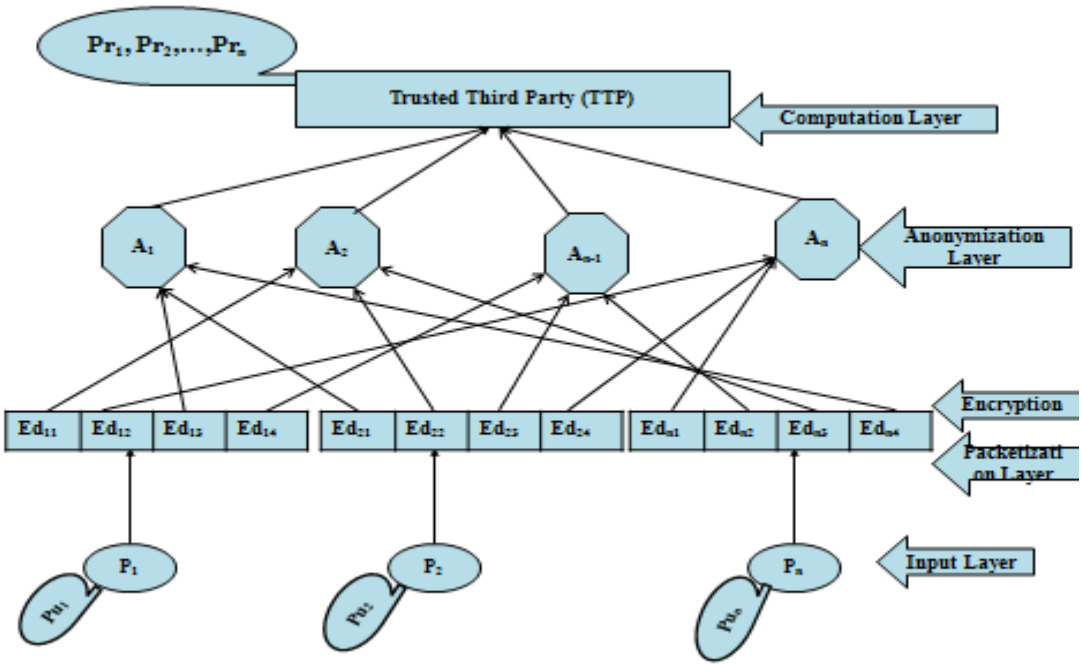

Figure 5.2: Architecture of *JCAE* Protocol

5.3.2 INFORMAL DESCRIPTION OF *JCAE*

The *JCAE* protocol is using asymmetric key for encryption, so parties first register with the system to get number of packets, anonymizers and a public key, divides the data into packets encrypts the data with public key send it to randomly chosen anonymizer to forward it to TTP. The TTP wait for computation until it receives inputs from all the anonymizers. As number of packets are decided in advance TTP first validate whether any packet is lost. After validation a key is selected from the available key pool to find the appropriate decryption key. Once decryption is done for all the parties TTP computes and broadcast the result *f*.

In this research readable data or unencrypted data is referred as plain data and represented as x_i/D_i. The procedure of hiding a message to protect its content is called encryption. This operation is represented as Enc(x_i, Pu_i). The encrypted message, E_i= Enc(x_i, Pu_i) is called cipher data, here x_i is plain data and Pu_i is encryption key (public key). The procedure of reversing cipher data back into plain data, D_i = Dec(B_i, k_j); is called as decryption, Here B_i is encrypted data and k_i decryption key (private key).

If the encryption key and decryption key are computationally deducible from each other, then it is called as symmetric encryption.

If the encryption key is distinct from the decryption key, then it is a public-key encryption or asymmetric encryption.

5.3.3 FORMAL DESCRIPTION

Inputs: (x_1, x_2... x_n) are inputs of *n* parties respectively.

Step 1: All the party register themselves before initiation of protocol, registration web service generates the asymmetric key and allot a public key Pu_i to registering party and forward private key Pr_i to TTP.

Step 2: All the parties divide data into fixed number of packets and encrypt the data packets (x_{i1}, x_{i2}... x_{itpk}) i.e. $P_{ij} = \sum_{i=1,j=1}^{n,\,tpk} E_i(x_{ij}, Pu_i)$. And send it to anonymizer.

Step 3: TTP verify total number of packets are same as the expected number of packets, if so TTP starts computation process. (Here, the identity of the party will be hidden as data is coming through anonymizers).

Step 4: For computation TTP first decrypt the packets with TTP's private keys.

Step 5: After decryption all the data TTP compute the result $f(D_1, D_2, \ldots, D_n)=S_D$ and broadcast the result via registration web service.

5.3.4 *JCAE* CHARACTERISTICS

5.3.4.1 CONFIDENTIALITY

JCAE achieved the confidentiality as per the expectations with improvement over the protocols presented in *sections 4.4* and *4.5* of *chapter 4*. All the parties in *JCAE* are using the unique public key to encrypt the data packets and send to TTP via randomly selected anonymizers. These data packets can be decrypted by the owner of private key. Therefore the unauthorised data access will not exist in the protocol. In this case, even if parties are semi-honest or the anonymizers are malicious, it will not break the protocol as the private key is known to TTP only. So here the collision between party & anonymizers is meaningless.

5.3.4.2 SECURITY

JCAE uses packetization, encryption and anonymization to attain the confidentiality, security and to hide individual identity. Any eavesdropper observing the packet transfer cannot predict the data as it is encrypted with an encryption key and private key to decrypt the packets is known to TTP only. This protocol is secure in case of '*n-1*' malicious parties as even if they get the data packets of intended party, it will be useless until they get the private key. Here, TTP doesn't know which private key belong to which party.

5.3.4.3 COMPLEXITY

The protocol (*JCAE*) assures confidentiality, security and anonymity so the complexity of computation increases, and throughput decreases. This decrease in throughput is due to increase in time for i) packetization and anonymization ii) encryption of data packets iii) most importantly to find decryption key. In worst case scenario $O(kn^2)$. This protocol increases complexity but abide to the objective of the research i.e. confidentiality and security.

5.3.5 PERFORMANCE ANALYSIS OF *JCAE*

Case 1: When the Data Anonymizers becomes Malicious

When the anonymizers dealing with parties data becomes malicious in that case, *JCAE* protocol will not affect as the private data is divided into packets and encrypted by the party with a key before sending to randomly selected anonymizers. The decryption key is known to TTP only. So even if the anonymizers become malicious they cannot break the privacy until the decryption key is known.

Case 2: Joint Malicious conduct by anonymizers and TTP

When the anonymizers and TTP join together for some malicious conduct then there is some probability of breaking individual confidentiality. It will be represented as: here 'm' is the total number of anonymizers; 't_{pk}' is the total number of packets per party, if k out of m anonymizers along with key anonymizer becomes malicious then the probability of breaking data privacy will be as shown in *Equation (5.1),*

$$\Pr(k,1) = \left(1 \Big/ {}_{m+1}C_{tpk+1} \right)^{(k)} \qquad (5.1)$$

Table 5.1: Joint Malicious conduct by Anonymizer and TTP

Number of Packets t_{pk}=5 Number of Malicious Anonymizers (k)	Maximum Number of Anonymizers m=8 Probability
2	0.056689342
3	0.013497462
4	0.003213682
5	0.000765162
6	0.000182181

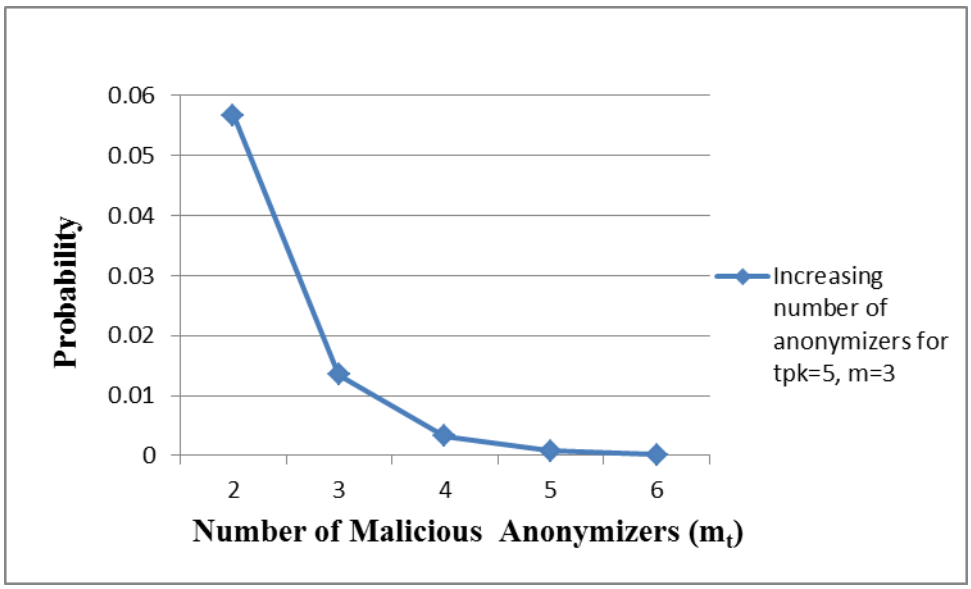

Figure 5.3: Probability of Joint Malicious Conduct by TTP and Anonymizers

The graph shown in *Figure 5.3* explains impact of joint malicious conduct by anonymizers and TTP. It is clear from the graph that probability of one anonymizer becoming malicious is considerable, as number of anonymizers increases the probability of becoming malicious is

insignificant (m>=5) so confidentiality increases with increase in number of anonymizers. Malicious behavior by one anonymizer cannot break the protocol.

Case 3: Joint Malicious conduct of the parties, anonymizers

When the parties collude with anonymizers for their benefit then there cannot be any loss of confidentiality for the targeted party until they get the private key from TTP. But if the TTP also join then the data of the target party may be lost if complete data of targeted party is with colluding anonymizers. In this case if r party out of n and l anonymizers out of m along with TTP becomes malicious then the probability of collision between all of them will be as shown in *Equation (5.2)*

$$\Pr(n,m,1) = \frac{1}{n^{l}} \times \frac{1}{m^{k+1}} \qquad (5.2)$$

Table 5.2: Collision between malicious parties and anonymizers

Number of Malicious party (n) / Number of malicious anonymizers(m)	Probability
1	0.0015625
2	1.95313E-05
3	2.44141E-07
4	3.05176E-09
5	3.8147E-11
6	4.76837E-13
7	5.96046E-15
8	7.45058E-17

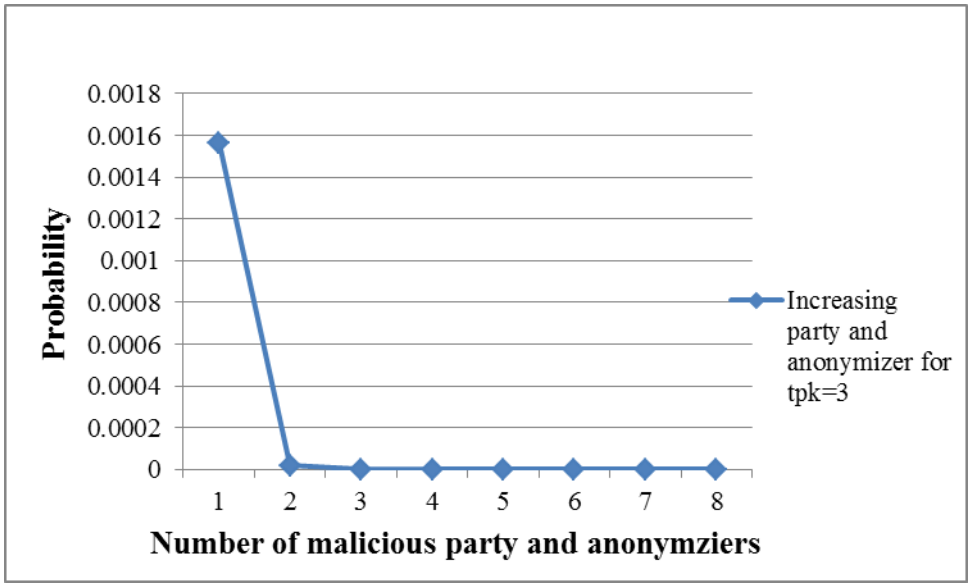

Figure 5.4: Probability of Joint Malicious conduct by parties and anonymizers

The graph shown in *Figure 5.4* explains impact of joint malicious conduct by parties and anonymizers. It is clear from the graph that probability of collision for joint malicious conduct of one party, one anonymizer with TTP is considerable. While as number of parties and anonymizers increases, the probability of collision becomes insignificant (no of parties n>=2, no of anonymizer m>=2). Hence confidentiality increases with increase in number of parties and anonymizers.

5.4 *JCAE* ALGORITHM

Joint Computation with Asymmetric Encryption (*JCAE*)

Assumptions:

1. Number of packets is same for all the parties.
2. Anonymizers are only to forward the data.
3. Parties give correct input.
4. TTP is trusted.

Inputs: (x_1, x_2, … , x_n) are parties input respectively.

(Ex_i, Pr_i, Pu_i), t_{pk} (Number of packets)

Output:

$f(D) = f(D_1, D_2, \ldots D_n)$;

Variable list:

n- Number of parties.

m- Number of anonymizers.

t_{pk}- Number of packets of each party.

T_{anony} -Total number of anonymizer for packet transfer.

$Count_{ttp}$ - Total number of packets at TTP.

Exp_{ttp} – Expected number of packets at TTP.

$Count_{key}$ – Total number of key received at TTP.

M_L_Anony - Maximum limit of anonymizer.

k[array]: Array for validation of key found (default value of array is 0)

B[array]: Array of Encrypted data;

$Count_{dkey}$: initialized to 0.// count of total number of decryption key found

Phase 1- (Registration)

a) All the parties first register with the system.

b) At the time of registration a key (Pu_i) is allotted to the party, and total packet (t_{pk}) & number of anonymizers (*m*) is decided.

Phase 2: (Packetization and Encryption)

for (*i*=1 to *n*) do

begin

a) Party '*i*' divides its private data into t_{pk} packets. // $x_i = D_{i1}, D_{i2}, \ldots D_{itpk}$;

for (*j*=1 to t_{pk})

begin

b) $E_{ij} = Enc(D_{ij}, Pu_i)$;

end; //end of *j* loop

end; //end of *i* loop

Phase 3 - (Anonymization)

for (i=1 to n) do

begin

for (j=1 to t_{pk}) do

begin

a) Randomly select an anonymizer 'A_{rn}';

If (Count(A_{rn}) < *M_L_Anony)* then

begin

b) Send E_{ij} to A_{rn};

c) Increase Count(A_{rn}) by 1;

else

d) Repeat step 2 (d);

end; // end of if

end; // end of *j* loop

end; // end of *i* loop

Phase 4 – (Data Collection at TTP)

for j=(1 to m) do

begin

for (i =1 to packets in anonymizer) do

begin

a) Redirect packets to TTP;

b) TTP append the packets in the pool;

c) $Count_{ttp}$= $Count_{ttp}$+1;

end; // end of *i* loop

end; // end of *j* loop

Phase 5 – (Data validation and Decryption)

a) $Exp_{ttp} = n \times t_{pk}$;

b) If $Count_{ttp} = Exp_{ttp}$ and $Count_{key} = n$ then

for (i = 1 to n) do

begin

for(j =1 to t_{pk}) do

begin

a) TTP select a private key Pr_i;

If (Pr_i is decryption key for E_{ij}) then

b) $D_{ij} = Dec(E_{ij}, \Pr_i)$;

c) $k[i]= i$;

else

d) Return 'Packet lost'

e) break;

endif;

end; //End of loop j;

end; //End of loop i;

Phase 6 – (Computation)

for (i=1 to n) do

begin

if (k [i] = n) then

begin

for (j=1 to t_{pk}) do

begin

a) $f_D = \sum_{i=1,j=1}^{n,tpk} D_{ij}$;

b) Broadcast $f(D)$;

end; // end of j loop

else

c) Return 'Packet lost/ key not found';

endif; //end of if statement

end; // end of i loop;

The proposed algorithm has different phases. Each phase has predefined functionality independent of others. The functions of phases are summarized as follows:

Phase 1 Registration:

- All the parties first register with the system.
- An encryption key Pu_i is allotted to each party.
- Number of packets (t_{pk}) and Number of anonymizers (m) is decided at the time of first party registration.

Phase 2 Packetization and Encryption:

- Each party divides its data into t_{pk} packets.
- All the packets are encrypted with the key Pu_i.

Phase 3 Anonymization:

- An Anonymizer is selected to forward the encrypted packets/cipher.
- First it is checked whether randomly selected anonymizer A_j has the capacity to accept the packet by comparing maximum limit of anonymizer (M_L_Anony) with the count (A_j) i.e. Total packets accepted before this step.
- If maximum limit of anonymizer is reached, then another anonymizer is selected.

Phase 4 Data Collection at TTP:

- After receiving packet all the anonymizers redirect the packets received to TTP.
- Counter variable $Count_{ttp}$ will be incremented by 1 for arrival of each packet.

Phase 5, 6 Data Validation, Decryption and Computation:

- First expected number of packets (Exp_{ttp}) at the TTP is compared with total packets received ($Count_{ttp}$) and total key with TTP ($Count_{ttp}$) is compared with expected total key (n) at TTP.
- If above conditions are satisfied, TTP takes a decryption key and check with packets to decrypt it.
- Another key is selected and the same procedure is repeated until the packets are found.
- Then it is checked whether the key is found for all the packets, if so computation is performed
- After decryption summation is performed on all the decrypted data packets to get the collaborative secure sum and broadcast the result.
- If condition is not satisfied then send the packet lost message and break the process.
- Otherwise key not found message is broadcasted.

5.5 COMPARATIVE STUDY

In this section the research work is compared with existing protocols, to show the improvement over them. *Table 5.3*

Table 5.3: Comparative Study of Secure Sum Protocols

Communication complexity	Minimum Number of Anonymizers for better	Minimum number of Packets	Minimum Number of party for protocol to work	Honest party	Computation at party level	Anonymity	Data leakage at party level/input layer	Confidentiality	Security /Eavesdropper attack	Case/ Protocol
O(n)	Anonymizers are	No	4 or more	Honest	Yes	No	Yes, Malicious party can collude to get the	Random number is used	No	Clifton's Secure
O(n)	Anonymizers are	fixed	3 or more	Honest	Yes	No	Yes, Could be if 2 party combines	If parties are honest and	No	k-secure sum
O(n)	Anonymizers are	Fixed	4 or more	Honest	Yes	No	Not as neighbors are changing	If parties are honest and	No	Ck-secure
$O(n^2)$	Anonymizers are	Same as number of	4 or more	Honest	Yes	No	No, as neighbors are changing in each	Privacy is persevered	No	Modified ck-secure
$O(n^2)$	Anonymizers are	Fixed same as	4 or more	Semi-honest	Yes	No	No, as one segment is distributed in	It will work in case of semi-	No	Dk-securesu
$O(n^2)$	tpk≤m	Fixed	2 or more	Semi-honest	No	Anonymization- to hide	No, As it is following ideal model of	Data Hiding Algorithm	Packetization	*JCRA*
$O(n^2)$	tpk≤m	Fixed	2 or more	Semi-honest	No	Anonymization- to hide	No, As it is following ideal model of	Data Hiding Algorithm	Packetization	*DRSS*
$O(n^2)$	tpk≤m	Fixed	2 or more	Semi-honest	No	Anonymization- to hide individual	No, As it is following ideal model of computation so there	Encryption technique (Hybrid	Packetization	*JCAE*

5.6 SUMMARY

The requirement of secure multi-party computation is tremendously increasing in the world due to the extensive use of internet for government, medical and personal purposes.

In this Chapter, the proposed protocol Joint Computation with Asymmetric Encryption (*JCAE*), presents the concept which can attain the twofold objective of attaining security and confidentiality of parties' private input. The protocol presented in this chapter has layered architecture where asymmetric encryption technique and anonymization are used. The proposed protocol is fairly efficient therefore it can be applied in real life scenarios.

The proposed protocol ensures data privacy when the number of parties and anonymizers are more (number of parties n>=2, number of anonymizer m >=2). Although in case of less party and one anonymizer the probability of joint malicious conduct is considerable but as the number of parties and anonymizers increases the probability of malicious conduct becomes insignificant. Here data security is also ensured since the input data is divided into packets before encryption and randomly distributed among different anonymizers.