CHAPTER 4

SECURE MULTI-PARTY COMPUTATIONS USING PSEUDO-RANDOMIZATION

The confidentiality is a big challenge for data generated from various sources such as social networking sites, online transaction, weather forecast to name a few. Various internet sources such as social networking sites, online shopping sites and cloud computing pica bytes of unstructured data is generated online with intrinsic values. The inflow of big data and the requirement to move this information throughout an organization has become a new target for hackers. This data is subject to privacy laws and should be protected. This big data can be utilized for collaborative computations. The proposed protocols are one step toward the security in case of above circumstances where data is coming from multiple participants and all are concerned about individual confidentiality and security

Today information enhancement has achieved a new velocity. The volume of information is blowing up from diverse sources. It leads to the requirement of security redefinition. As big data represents inviting opportunity for business security vendors identified that big data requires a different approach to security, SMC can be a solution for the security issues with big data (Kamara, 2011).

It has been presented in literature that, basic security conditions are applied at the network level. During this situation if the circumference of the network is damaged by the attacker then, they get complete unlimited access to the big data. Hence there are requirement to keep the controlling condition close to data. As the participants' basic concern is their data

security therefore the data packets must be highly secure against attacks (Mishra, 2008).

Retailer consumer's data is often analysed to decide production and maintain inventory system. In this scenario, individual confidentiality and security is the main concern. The presented Secure Sum protocol can be applied in the above mentioned circumstances.

Secure multi-party computation is required in collaborative computation; individuals are interested in collaborative computation for financial growth, and to identify consumer behaviour etc. Here individual participants are more concerned about their confidentiality and security. The proposed protocols are an initiative to fulfil the user's security concerns and to maintain confidentiality during computations.

This chapter presents pseudo-randomization technique and distributed randomization technique to maintain confidentiality in secure sum using anonymization.

4.1 INTRODUCTION

SMC works are mostly allocated to software agents, which can signify the participating parties, anonymizers and trusted third party. For example, in an organization multiple meetings are scheduled for same team for different agenda, in this case there could be announcement delays and other meetings are also scheduled simultaneously; it is a repetitive, long and monotonous job for individuals and may result in unproductive outcomes. So this meeting schedule can be automatized for better outcome keeping confidentiality of individuals using secure multi-party computations.

One technique to minimize confidentiality loss is to encode and hide (identity) the association of the parties' sensitive data. Usually this is

concerned as encryption and anonymization. In this chapter we will present models developed for secure sum problem using pseudo-randomization and anonymization.

Organizations are moving form in-premises infrastructure to cloud, where they need not maintain their own resources rather they dynamically pay and use resources from cloud infrastructure as per their need. Big data is an essential feature of cloud computing. Big Data is generated from various sources such as health care systems and services, manufacturing systems and services and online services. This data collected from various sources can be used for collaborative computations to get mutual benefit and help scientific research, such as to find out customer buying habit by super markets, diagnosis of disease in early stage etc. But there is a concern for individual's confidentiality and security in collaborative computations. And the level of trust all participants have on the computation authority. Hence there is a need of a mechanism which can assure secure computations and confidentiality in multi-party environment. It could be applicable to big data and cloud computing.

4.2 SECURE SUM PROBLEM

The secure sum of parties' personal inputs is a good example of SMC which has attracted the attention of researchers from organization and academics to develop SMC protocol with minimum data leakage and higher confidentiality and security.

The secure sum protocol was initiated by Clifton *et al.* (Clifton, 2002) in this authors used randomization technique for collaborative computation. In the proposed protocol participating parties were organized in a one-way ring. One party works as originator of the protocol through which computation begins by deciding a random number and adding it to

its private data. The sum is forwarded to next party for further computation and so on.

Confidentiality conservation for secure sum problem has been achieved by other methods also (Sheikh, 2010a; Sheikh, 2010b; Sheikh, 2010c) by mean of randomization and distribution. But this solution does not deal with problem of confidentiality leakage in certain scenario. This chapter presents methods to maintain confidentiality for secure sum using pseudo-randomization and anonymization.

4.3 FORMALIZATION OF SECURE SUM PROBLEM

The secure sum problem consists of set of parties who wish to perform collaborative sum over parties' private input without revealing the data and identity of participants.

The Secure Sum problem can be viewed in following steps:

1. A set of n parties $P_1, P_2 \ldots P_n$ each party holds private data x_i where $i \in (1, 2 \ldots n)$.
2. A data set $D = \{x_1, x_2 \ldots x_n\}$ is the set of all the parties' private data.
3. Secure sum of parties private data need to be computed without disclosing sensitive information.
4. Various constraints exist between the parties to share the data among the parties.
5. The intra-party constraints of party P_i on x_i are the personal information of party P_i and this is known only to party P_i.
6. The solution S_{ln} is a representation of the secure sum of variable set.

4.4 SECURE SUM PROTOCOL USING PSEUDO-RANDOMIZATION

The secure sum problem solution can be achieved using pseudo-randomization technique. In the protocols *JCRA* and *DRSS* the solutions are based on single and multiple pseudo-random numbers.

4.4.1 ARCHITECTURE OF JOINT COMPUTATION WITH RANDOMIZATION AND ANONYMIZATION (*JCRA*) PROTOCOL

Figure 4.1 shows the layered architecture of *JCRA* protocol. Here, each layer has predefined functionality.

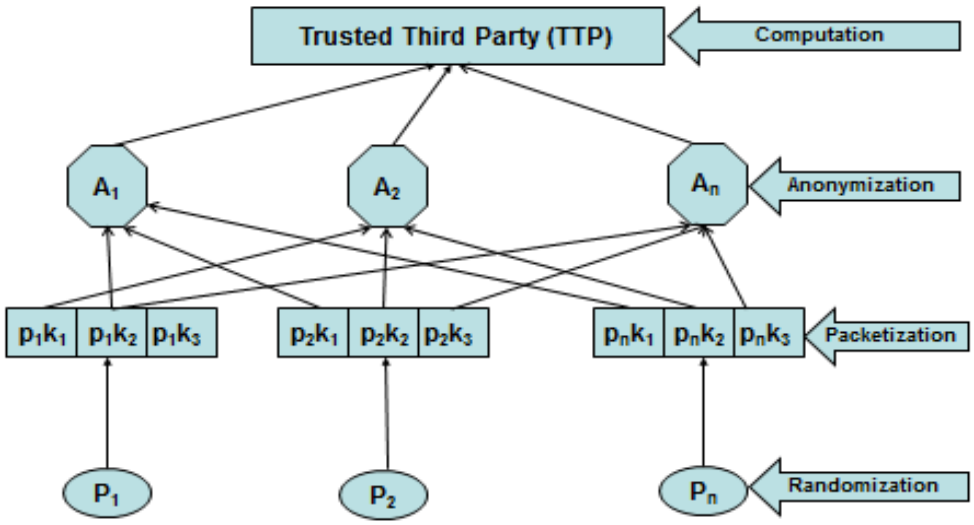

Figure 4.1: Architecture of *JCRA*

4.4.2 INFORMAL DESCRIPTION OF *JCRA*

This protocol is based on ideal model prototype of SMC. In this protocol parties interact only to generate pseudo-random number with the help of pseudo-randomization algorithm. It will be performed exactly once to generate a pseudo-random number (r) on which all the parties are mutually agree. Pseudo-Random number is used by randomization function to prevent parties' private data from unauthorised access. Pseudo-

Randomization function random (o, x_i, r) would be decided by one arbitrarily chosen party and shared among all other parties and TTP. Once it is received by all the parties, parties will use pseudo-randomization function to hide the actual input then break it in fixed number of packet and distribute among various anonymizers. Anonymizers then forward the packet to TTP. TTP recollects the data with the help of pseudo-randomization function to perform the collaborative computations.

4.4.3 FORMAL DESCRIPTION *JCRA*

Inputs: (P_1, P_2, ….., P_n) are parties with (x_1, x_2, …, x_n) inputs.

TTP has no input, all the parties and TTP knows pseudo-random number r and pseudo-randomization function random (o, x_i, r).

Outputs: All the parties and TTP learn $f(x_1 x_2, \dots, x_n)$.

Step 1: All the 'n' parties jointly executes pseudo-randomization algorithm to generate pseudo- random number r, as a result all the parties learn r.

Step 2: One Randomly selected party generates pseudo-randomization function random (o, x_i, r) and share it with 'n' parties and TTP.

Step 3: All the 'n' parties then computes

$R(x_1)$ = random (o, x_1, r)

$R(x_2)$ = random (o, x_2, r)

$R(x_n)$ = random (o, x_n, r)

Step 4: Each party divides the pseudo-randomized data into fixed number of packets and forward it to randomly selected anonymizer.

Step 5: Anonymizers then forward it to TTP.

Step 6: After receiving complete data of all the parties TTP recollects the data and evaluates collaborative computation function $f(D)$ and announce the result.

4.4.4 *JCRA* CHARACTERISTICS

4.4.4.1 CONFIDENTIALITY

In *JCRA* the secure sum computation is performed as per the rule. All the parties agree on a pseudo-random number to hide, packetize and pass the data. In the protocol even if *'n-1'* parties becomes malicious they cannot break the protocol until they collide with the anonymizers who are selected to forward the packets of the targeted party. Collision of any party with anonymizers containing targeted parties packets can lead to protocol break as all the parties are using same pseudo-random number '*r*'. It has been observed in the simulation results, the probability of collision between party and anonymizers becomes insignificant when we increase number of anonymizers or number of packets.

4.4.4.2 SECURITY

In the protocol (*JCRA*), to achieve security feature data is packetized and pseudo-randomized on the basis of time based randomization function. Any eavesdropper observing the packets cannot predict the data until he gets the details of all the packets and the pseudo-random number. Use of pseudo-random number overcomes the replay attack. This protocol is secure even if '*n-1'* parties collide until they combine with all the anonymizers containing the targeted parties' packets.

4.4.4.3 COMPLEXITY

The protocol (*JCRA*) assures confidentiality as well as anonymity so the complexity of protocol increases as compare to other ring based secure sum protocol. Protocol performs as per the rule in semi-honest model. If few anonymizers and parties divert from the protocol and collude to behave maliciously then the protocol break, as per the result probability of such behaviour is insignificant.

4.4.5 PERFORMANCE ANALYSIS OF *JCRA*

The performance analysis of *JCRA* protocol has been performed in various cases such as by increasing parties, packets and anonymizers. In the section below the probabilistic results are shown.

Probabilistic Analysis

Case 1: Increasing number of Parties- *Figure 4.2*, shows the packet transfer in *JCRA* by increasing number of parties for packets t_{pk}=3 and anonymizers m=8 in an average case. Here, averages case means to get the available anonymizer, parties have to check around half of the anonymizers.

$$P(n) = \left(n * t_{pk}\right) * \left(\frac{m}{2}\right) \tag{4.1}$$

Table 4.1: Number of packet transfers for varying parties

Number of Parties (*n*), t_{pk}=3, *m*=8	**Number of Packet Transfers**
2	24
3	36
4	48
5	60
6	72
7	84
8	96

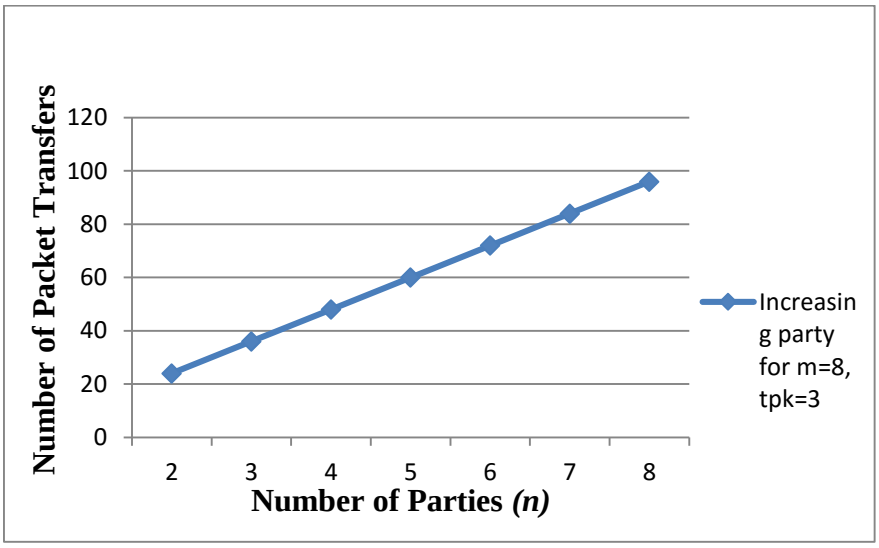

Figure 4.2: Packet transfer in *JCRA* by increasing parties

It is clear from the *figure 4.2* and *Equation (4.1)* that as the number of packet increases the communication between different layers increases.

Case 2: Increasing number of Packets- *Figure 4.3* shows the probabilistic representation of data security in *JCRA* for varying number of packets. Here, parties *n*=3 and anonymizers *m*=8 in an average case.

$$P(k,m) = (k/m)^n \qquad (4.2)$$

Table 4.2: Probability of Data Security by varying packets

Number of Packets t_{pk}, n=3, m=8	Probability of Data Security at packet layer
3	0.05273
4	0.12500
5	0.24410
6	0.42187
7	0.66992
8	1.00000

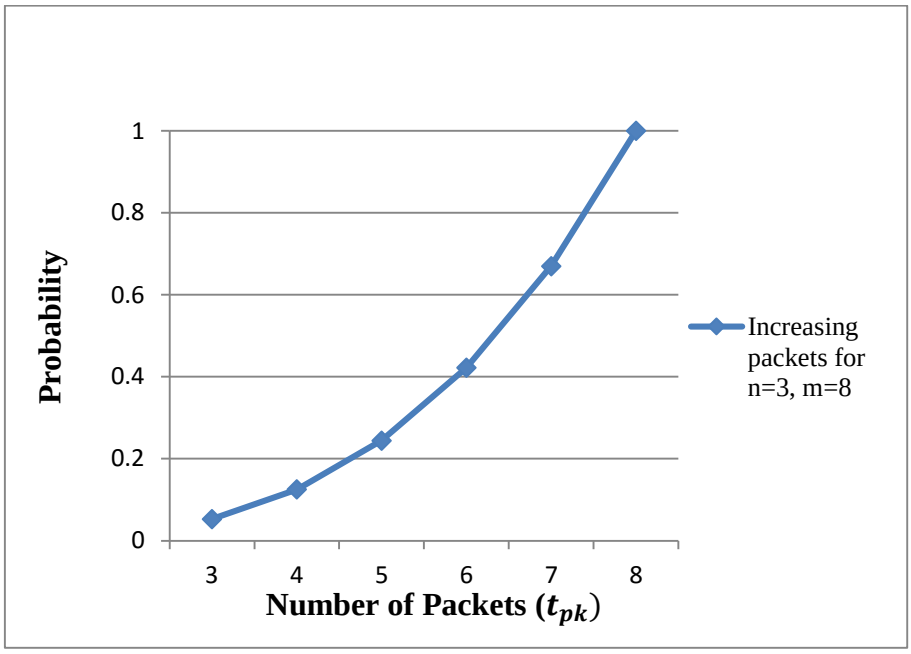

Figure 4.3: *JCRA* data security metric for varying packets

It is clear from the *figure 4.3* and *Equation (4.2)*, as the number of packets increases to total number of participating anonymizers' '*m'*, the data security increases at the cost of time.

Case 3: Increasing number of Anonymizers- *Figure 4.4*, shows the curve for varying the number of anonymizers. The probability of selection of

malicious anonymizer is $1/m^2$ and the probability of selection of t_{pk} malicious anonymizers will be as shown in *Equation (4.3)*.

$$P(n, m, tpk) = tpk * (1/m^2)^n \quad (4.3)$$

Table: 4.3: Probability of data security by varying anonymizers

Number of packets t_{pk}	Number of Anonymizers m	Probability of malicious conduct by anonymizers
3	3	0.03703
4	4	0.00659
5	5	0.00172
6	6	0.00058
7	7	0.00023
8	8	0.00011

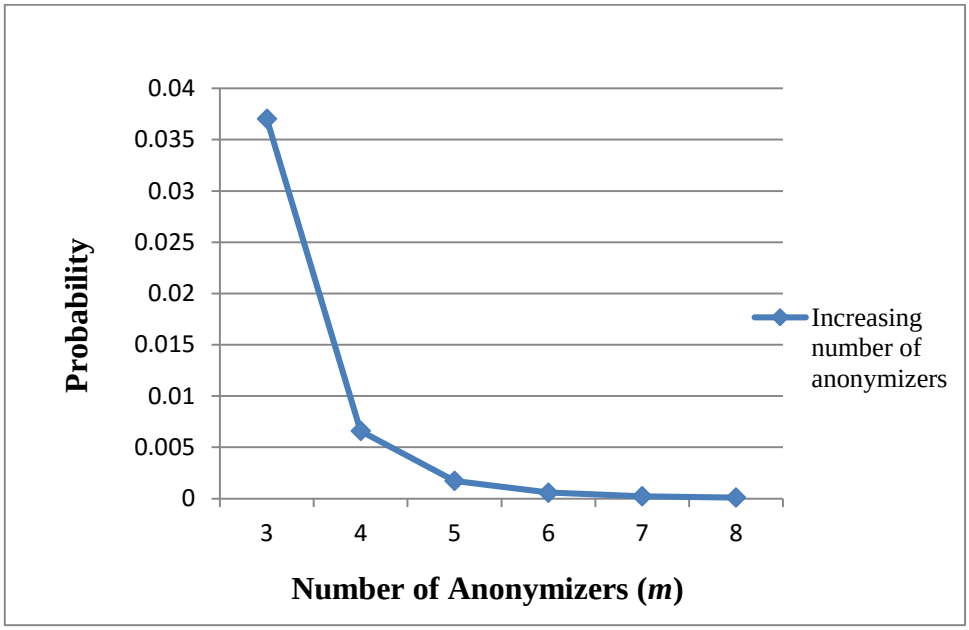

Figure 4.4: *JCRA* security metric for varying anonymizers

Here, it is clear from the *figure 4.4* as total number of participating anonymizers' increases the probability of getting malicious anonymizers

decreases and efficiency of the protocol increases. It leads to increase in data security.

Case 4: Malicious conduct by certain parties- All the parties will have following information: i) Pseudo-Random number *'r'* and pseudo-randomization function, ii) total number of packets of individual party 't_{pk}' iii) Parties own data *'D_i'*. This information is not sufficient to get any third parties data even if some parties collide.

Case 5: Malicious conduct by Anonymizers- In '*JCRA'* protocol anonymizer receives randomized packets *'P_iK_{tp}'* where *'t_{pk}'* is total number of packet of individual parities. In this case even if few anonymizers collude they will not get any information until they get pseudo-randomization function and pseudo-random number *'r'* as data is pseudo-randomized.

In case, if anonymizer gets pseudo-randomization function and pseudo-random number '*r*' then there might be some probability of breaking the protocol, representation of this will be as shown in *Table 4.4* and *Equation (4.4)*.

$$P(k, m) = \frac{1}{m^k} \quad (4.4)$$

Table 4.4: Probability of joint Malicious Conduct by Anonymizers in *JCRA*

Number of Malicious Anonymizers	Probability of joint malicious conduct
2	0.015625
3	0.001953
4	0.000244
5	3.05E-05
6	3.81E-06
7	4.77E-07
8	5.96E-08

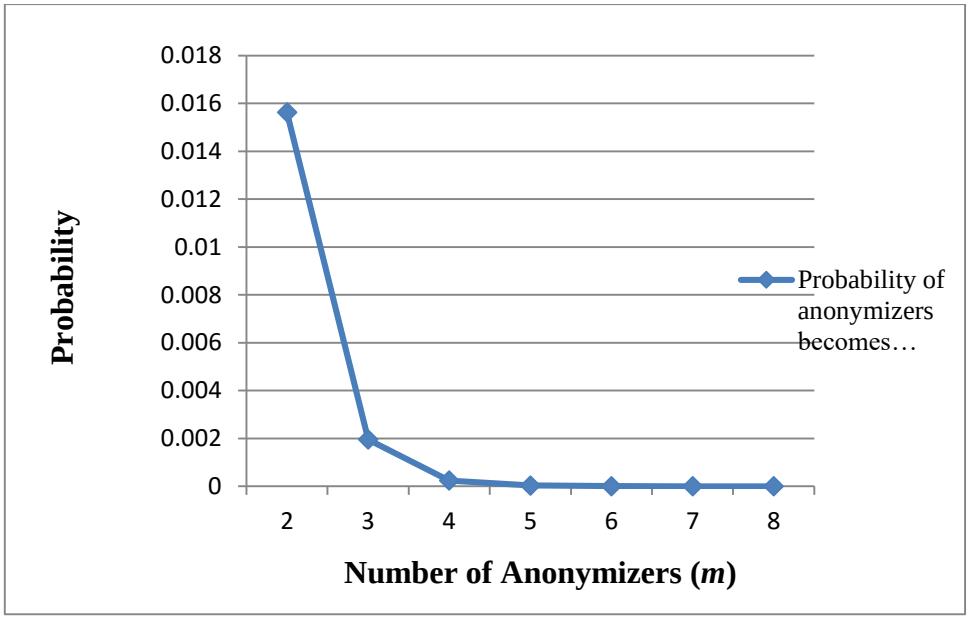

Figure 4.5: *JCRA* metric in case of malicious conduct by Anonymizers

It is clear from the *figure 4.5*, if number of anonymizers is four or more the probability of breaking the protocol is insignificance. So as the number of packets increases the probability of breaking the protocol becomes negligible.

Case 6: Joint Malicious conduct by anonymizers and parties: In the protocol presented in the thesis number of parties are *'n'* and number of anonymizers are *'m'*. Following information is known to parties:

i) Pseudo-Random number *'r'* and pseudo-randomization function, ii) total number of packets of individual party *'t_{pk}'* iii) Parties own data. Anonymizers receive randomized data packets from different parties, as the parties packets are distributed among *'m'* anonymizers; probability of getting any parties data is negligible until they collide. *Table 4.5 and Equation (4.5 & 4.6)*, shows the probabilistic results.

If '*k*' anonymizers collide with '1' party out of '*n'* then probability will becomes:

$$P(1,k) = \frac{1}{n} \times \frac{1}{m^k} \quad (4.5)$$

If '*k*' anonymizers collide with '*l*' parties then probability will becomes:

$$P(l,k) = \frac{1}{n^l} \times \frac{1}{m^k} \quad (4.6)$$

In proposed protocol for probabilistic analysis maximum number of anonymizers *m*=8 and maximum number of parties *n*=8.

Table 4.5: Probability of collision between malicious anonymizer and parties in *JCRA*

Number of malicious Anonymizers ↓	Number of malicious Parties → 1	2	3	4
2	0.001953125	0.000244141	3.05176E-05	3.8147E-06
3	0.000244141	3.05176E-05	3.8147E-06	4.7683E-07
4	3.05176E-05	3.8147E-06	4.76837E-07	5.9606E-08
5	3.8147E-06	4.76837E-07	5.96046E-08	7.4508E-09
6	4.76837E-07	5.96046E-08	7.45058E-09	9.3133E-10
7	5.96046E-08	7.45058E-09	9.31323E-10	1.1615E-10
8	7.45058E-09	9.31323E-10	1.16415E-10	1.4519E-11

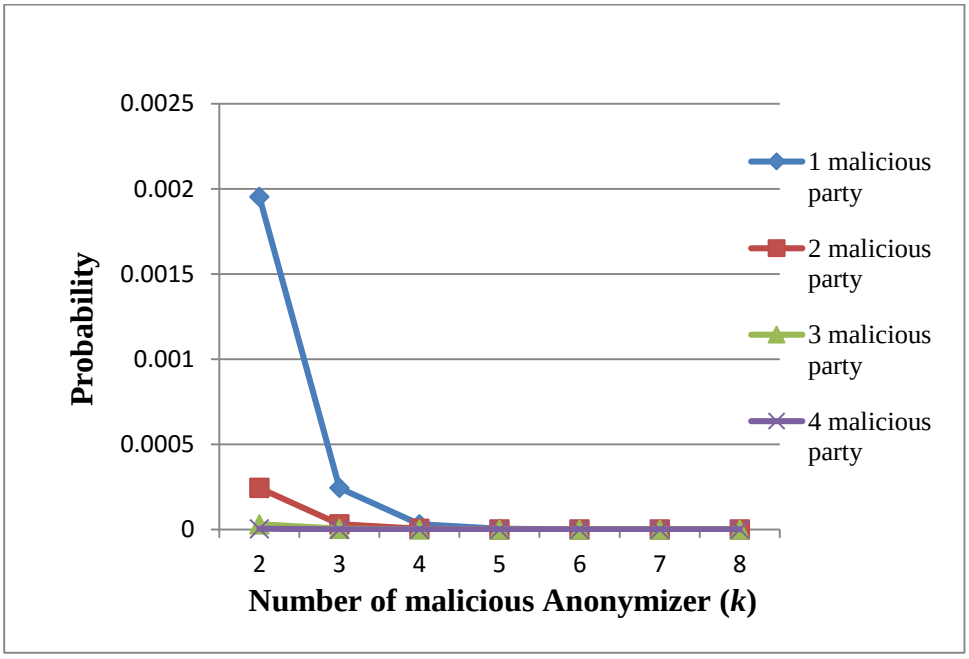

Figure 4.6: *JCRA* metric for collision between parties and anonymizers

Figure 4.6 shows the result of computations in graphical form. It shows that probability of any two anonymizers and any one party

becoming malicious together is more but as we increase number of anonymizers' probability of anonymizers becoming malicious together is insignificant. For the testing purpose minimum *'t_{pk}=3'* packets are assumed so at a time *'t_{pk}'* anonymizers should be malicious along with the party.

Case 7: Joint Malicious Conduct by TTP and Anonymizers: In the *JCRA* protocol single TTP is considered for joint computations and number of anonymizers is *'m'*. Following information is known to TTP: i) Pseudo-Random number *'r'* and pseudo-randomization function, ii) total number of packets of all the parties *'t_{pk}'*. In case if TTP behave maliciously then probability of breaking the protocol will be as shown in *Table 4.6*

$$P(1,k) = \frac{1}{m^{k+1}} \quad (4.7)$$

Table 4.6: Probability of collision between Anonymizers and TTP

Number of malicious Anonymizers	Probability of TTP and Anonymizers joint malicious conduct
2	0.001000000
3	0.000100000
4	0.000010000
5	0.000001000
6	0.000000100
7	0.000000010
8	0.000000001

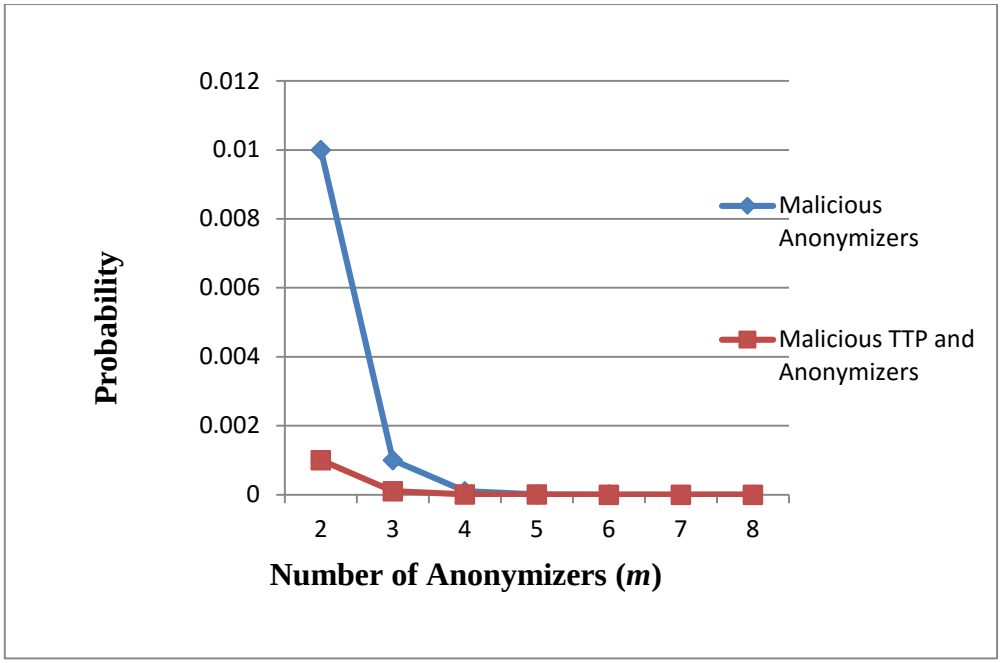

Figure 4.7: *JCRA* metric for collision between TTP and anonymizers

Figure 4.7 shows that as the number of anonymizers increases the probability of breaking of protocol decreases and subsequently becomes insignificant.

4.4.6 *JCRA* ALGORITHM

Joint Computation with Randomization and Anonymization (*JCRA*)

Assumptions:

1. Number of packets is same for all the parties (t_{pk}).
2. Anonymizers are only to forward data.
3. TTP computes correctly.
4. All the parties are giving valid input.

Inputs: (D_1, D_2, …, D_n) parties input for each party respectively, random(o,D_i, r), random number (r), Number of packets (t_{pk}).

Output: $f(D)$

Variable list: n - Number of parties.

t_{pk} - Number of packets.

A_r - Randomly selected anonymizer.

T_{ann} - Total number of anonymizer.

S_D - Combine input of all the parties.

C_{tpk} - Total packet count at TTP.

E_{tpk} - Expected number of packet at TTP.

Max_limit_anonymizer- Maximum number of packet an anonymizer can receive.

//**Phase 1: Pseudo-Randomization**

1. First party registering the system will generate the pseudo-random number '*r*', and total number of anonymizers and packets per party are decided at the time of first party registration.
2. P_i generates random(O, D_i, r) and share with registration web service to share it with other parties ;
3. for (i = 1 to n) do
4. begin
5. $R(D_i)$ = random(O,D_i, r)// pseudo-randomize the individual data;
6. end;

// Phase 2: Packetization

1. Divide $R(x_i)$ in t_{pk} parts. // (p_ik_1, p_ik_2… p_ik_{tpk});
2. for (j=1 to t_{pk}) do
3. begin
4. randomly select one anonymizer A_r;
5. if (count(A_r)<*Max_limit_anonymizer*) then
6. begin
7. send p_ik_j to A_r;
8. increase the count(A_r) by 1;
9. end;
10. else
11. chose another anonymizer and repeat step 3(c);
12. end;

13. end;

//Phase 3: Data Collection at TTP

1. for (j = 1 to T_{ann}) do
2. begin
3. for (i = 1 to packet in anonymizer) do
4. begin
5. redirect packet to TTP;
6. TTP will append packet to S_D;
7. $S_D = \sum_{i=1}^{n} Di + r$
8. C_{tpk} increased by 1; // increase total packet count by 1.
9. end;
10. end;

//Phase 4: Data Verification and Computation

begin

a) $E_{tp} = n \times tp$;

If (C_{tpk} = E_{tpk}) then

b) Compute $f(D)$ by derandomizing S_D using r;

c) Broadcast the result $f(D)$;

else

d) return 'packets lost';

end;

4.5 SECURE SUM PROTOCOL USING DISTRIBUTED PSEUDO-RANDOMIZATION (*DRSS*)

In this protocol to increase the confidentiality multiple pseudo-random numbers are used. So in-order to get actual value of a party's input along with *'t_{pk}'* cipher data packets all the pseudo-random number packet are required.

4.5.1 ARCHITECTURE OF *DRSS* PROTOCOL

Figure 4.8 shows the layered architecture of *DRSS*. Each layer shown in the architecture has predefined functioning.

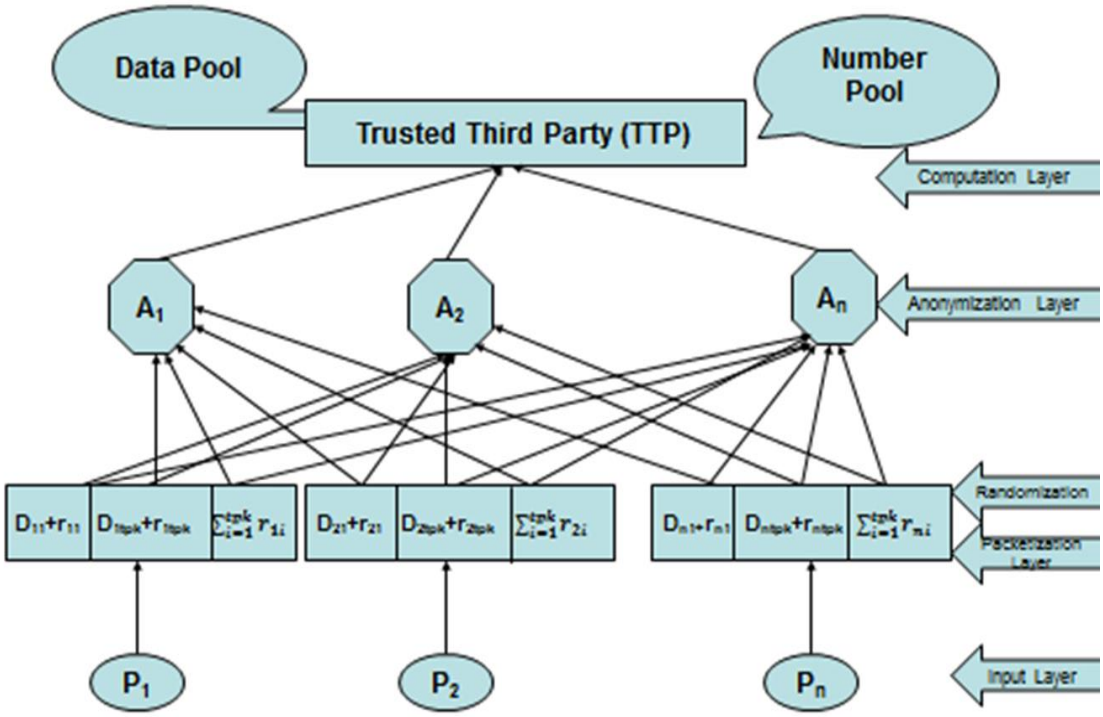

Figure 4.8: Architecture of *DRSS*

4.5.2 INFORMAL DESCRIPTION OF *DRSS*

In this protocol pseudo-randomization and packetization are used together to solve secure sum problem. This protocol will work suitably in case when the majority of participants are "honest but curious". Anonymization layer is used to hide the identity of participants. The party chooses different anonymizer to send pseudo-randomized data and pseudo-random numbers (r_{ij}, r_{ij+1}… r_{itpk}). TTP on receiving data keep it in data pool and pseudo-random number in the pseudo-random number pool. Once complete data and all pseudo-random numbers are received, TTP perform validation of the data packet, and pseudo-random number by comparing the total

number of expected packets and received packets if validation is successful TTP proceed for computation.

4.5.3 FORMAL DESCRIPTION OF *DRSS*

Input: (D_1, D_2… D_n) are data of each party respectively.

Step 1: Parties divide data into 'k' packets and generate 'k' pseudo-random numbers locally, and makes 'k' pairs as (D_{ji}, r_{ji}) (here j represents party whose data is pseudo-randomized) and r_{ji} is the pseudo-random number added to packet D_{ji}.

Step 2: All the packets (D_{ji}, r_i) are sent to anonymizers such that if A_i is getting D_{ji} then total of all the random number added to parties packets $\sum_{i=1,j=1}^{n,tpk} r_{ji}$ must be sent to another anonymizer A_j.

Step 3: Anonymizers after receiving all the packets, forward data packets to data pool and pseudo-random number to separate pool.

Step 4: After receiving all the packets TTP first verify whether the expected number of packets is same as the received number of packets.

Step 5: If the verification is successful, then TTP proceed for secure collaborative computation of over the data packets received such that no information is leaked other than what can be interpreted from the output.

4.5.4 *DRSS* CHARACTERISTICS

4.5.4.1 CONFIDENTIALITY

The protocol (*DRSS*) works on multiple pseudo-random number to ensure higher degree of confidentiality. As all the packet are padded with a pseudo-random number generated by the party so the probability of getting targeted parties data is insignificant. Because each parties pseudo-random number is known to party only so other party cannot break the protocol until they collude with *'t_{pk}+1'* anonymizers. Here, the usage of pseudo-random number avoids replay attack.

4.5.4.2 SECURITY

In the protocol (*DRSS*), to achieve security feature data is packetized and different pseudo-random number is added to each packet. These pseudo-random numbers are generated by time based randomization function. Any eavesdropper observing the traffic cannot predict the data until he gets the details of all the packets and all the pseudo-random numbers. Use of time based pseudo-random numbers overcomes the replay attack. This protocol is secure even if *'n-1'* parties collide until they combine with all the anonymizers containing the targeted parties' data packets and pseudo-random number.

4.5.4.3 COMPLEXITY

As different pseudo-random numbers are added by every party in each packet it increases computation complexity as well as communication complexity. But enhance confidentiality and security.

4.5.5 PERFORMANCE ANALYSIS OF *DRSS*

Probabilistic Analysis

Case 1: Increasing Number of Party: *Figure 4. 9* show the number of communication by increasing the number of parties in an average case. Here, average case means, we need to check with $m/2$ anonymizers to get the appropriate anonymizers as per the *DRSS* protocol assumptions (*c. f. 4.5.6*)

$$P(n, m, t_{pk}) = (((n \times t_{pk}) + 1) \times (\frac{m}{2})^2 \quad (4.8)$$

Table 4.7: Number of packet transfer for varying parties

Number of Party (*n*)	Number of Packet transfer
2	112
3	160
4	208
5	256
6	304
7	352
8	400

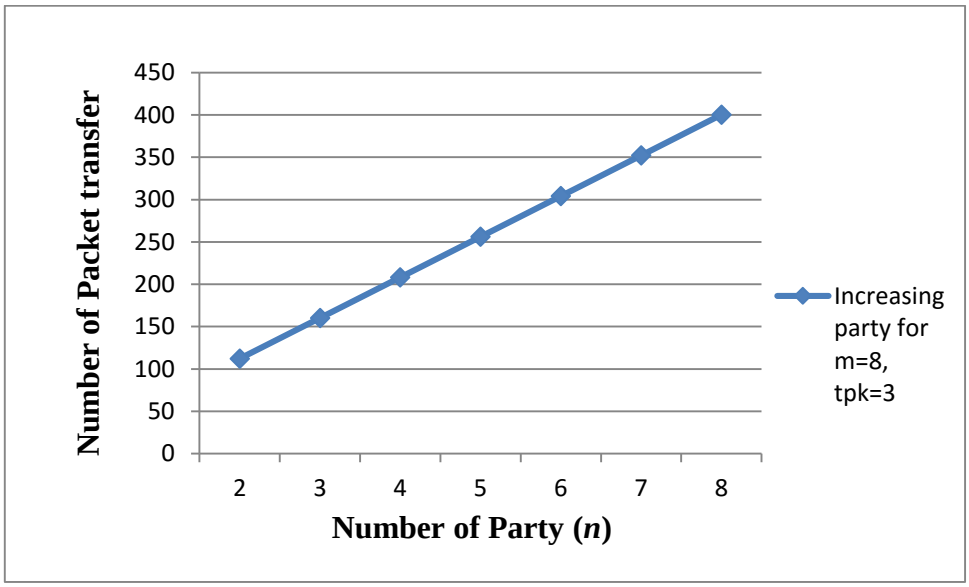

Figure 4.9: Packet transfer in *DRSS* by increasing parties

It is clear from the *figure 4.9* and *Equation (4.8)* that as the number of packet increases the communication between different layers increases linearly. As communication increases so the throughput decreases.

Case 2: Increasing number of Packets: *Figure 4.10* shows the probabilistic representation of data security in *DRSS* for varying number of packets. Here, parties n=3 and anonymizers m=8 in an average case

$$P(n, m, t_{pk}) = \left(\frac{k+1}{m}\right)^n \quad (4.9)$$

Table 4.8: Probability of Data Security by varying packets

Number of packets (t_{pk})	Probability of data security at packet layer
3	0.06250
4	0.12207
5	0.21094
6	0.33496
7	0.50000
8	0.71191

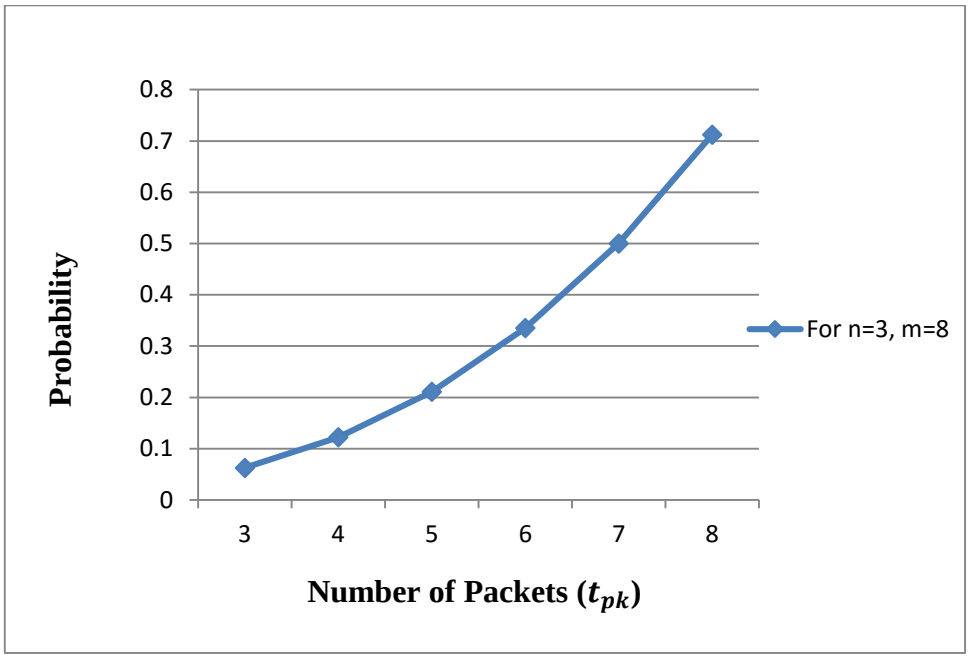

Figure 4.10: *DRSS* data security metric for varying packets

Case 3: Increasing number of Anonymizers: *Figure 4.11* shows the curve for varying the number of anonymizers. The probability of selection of malicious anonymizer is $1/m^2$ and the probability of selection of (t_{pk} +1) malicious anonymizers will as shown in *Equation* (4.10)

$$P(n, m, t_{pk}) = (t_{pk} + 1) \times (\frac{1}{m^2})^n \quad (4.10)$$

Table 4.9: Probability of data security by varying anonymizers

Number of packets (t_{pk})	Number of Anonymizers (m)	Probability of malicious conduct by anonymizers
3	3	0.004115
4	4	0.000976
5	5	0.000320
6	6	0.000128
7	7	5.9499E-05
8	8	3.05176E-05

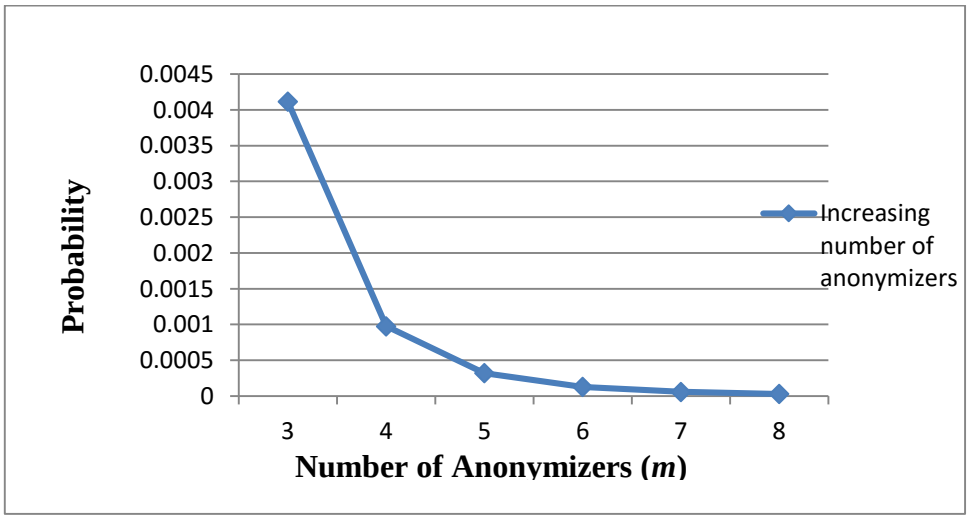

Figure 4.11: *DRSS* security metric for varying anonymizers

Here, it is clear from the *figure 4.11*, as we increase the number of anonymizers the probability of getting malicious anonymizers decreases and efficiency of the protocol increases. It leads to increase in data security.

Case 4: In *section 4.4.6*, it is shown when only one pseudo-random number *'r*' is added to the data, In that case even if '1' out of '*n*' party becomes malicious and share '*r*' with anonymizer then there is some probability of confidentiality loss. In current scenario, the above problem of confidentiality loss can be avoided as, let '*k*' is the number of packets and after adding '*r'* it will become '*k+1'*,

Equation (4.11) shows one party's data distribution to '*p*' malicious anonymizers out of '*m'* is represented as

$$\Pr(1, p, m) = \left(\frac{p}{m}\right)^{(k+1)} \tag{4.11}$$

'*n*' parties data distribution to '*p'* malicious anonymizers is represented in *Equation (4.12), i. e.* If '*p'* malicious anonymizers out of '*m*' anonymizers combine together, then the probability of breaking the protocol will be as follows:

$$\Pr(p, m) = \frac{1}{n} \times \left(\frac{p}{m}\right)^{(k+1)} \tag{4.12}$$

Table 4.10: Probabilistic result of *DRSS* by increasing number of malicious anonymizers

Number of Malicious Anonymizers	Probability
2	1.90735E-06
3	4.88833E-05
4	0.000488281
5	0.002910383
6	0.012514114
7	0.042951114
8	0.125000000

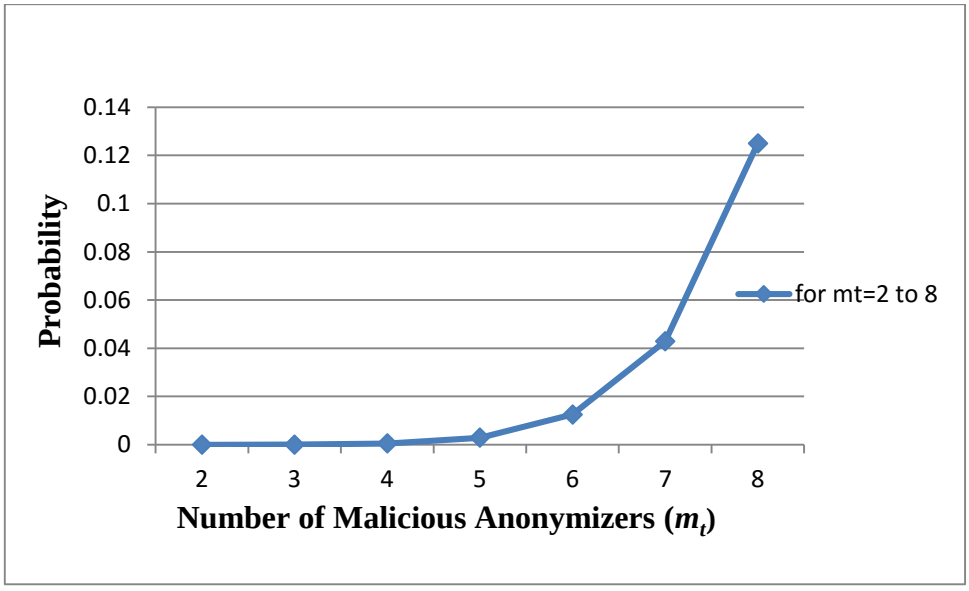

Figure 4.12: Probability of breaking Protocol in case of Malicious Anonymizers

Figure 4.12, shows that even if '*m'* (here *m* is the total number of anonymizers) anonymizers combine probability of breaking protocol is insignificant. (For analysis purpose '8' anonymizers are considered).

Case 5: When different pseudo-random numbers r_1, $r_2 \ldots r_k$ are added to each packet as $D_{11}+ r_{11}$, $D_{12}+ r_{12} \ldots D_{1k}+ r_{1k}$ then the probability of breaking

the protocol will be represented by *Equation (4.13)*, if '*l*' out of '*m*' anonymizers, combine together to reveal targeted parties data as well as pseudo-random numbers then the probability of breaking in case of different number of packets is shown in *Figure 4.13*.

$$\Pr(l,m) = (\frac{l}{m})^{(k+1)} \qquad (4.13)$$

Table 4.11: Test result by increasing pseudo-random number and malicious anonymizers

Pseudo-random Number (r) ↓	Number of malicious Anonymizers (m_t) →	1	2	3	4
	2	0.000100	0.00160000	0.00810000	0.025600000
	3	0.000001	0.00006400	0.00072900	0.004096000
	4	1E-08	0.00000256	6.56E-05	0.000655

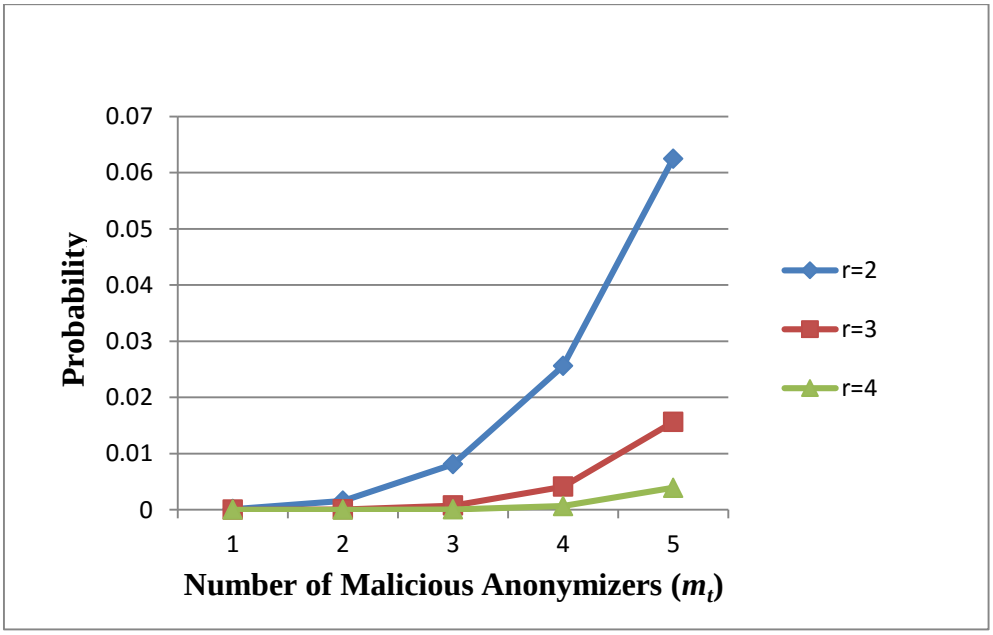

Figure 4.13: Probability of breaking protocol by varying pseudo-random number and malicious anonymizers

Figure 4.13, shows the probability of breaking the protocol, by varying pseudo-random numbers. To get a parties complete data, data packets and pseudo-random number packet should be combined, and this packets are distributed to '*k+1'* different anonymizers. So the probability of data leakage in the protocol becomes insignificant as we increase number of packets. To get all the packet of intended party *'t_{pk}+1'* anonymizers should also be malicious and that is very rare.

Case 6: When *'n-1'* out of '*n*' party join together to get the private input of n^{th} party, in the proposed protocol, they cannot break the protocol and get the private input of another party. If these '*n-1'* parties try to interpret the input of n^{th} party from the computation result, they need to have pseudo-random numbers of n^{th} party that is known to party only.

4.5.6 *DRSS* ALGORITHM

Algorithm: Distributed Randomized Secure Sum (*DRSS*)

Assumption:

1. Participating parties are "semi-honest".
2. Same number of packets is used by all the parties.
3. Data and pseudo-random number are provided to different anonymizers
4. TTP is trusted.
5. Parties provide correct input.

Input: (D_1, D_2, …, D_n) are input of each party respectively.

Variable list:

n: number of parties.

m: number of anonymizers.

k: number of packets for each party.

r_i : pseudo-random number //initialized to 0.

S_D: Data pool

S_R: Pseudo-Random number pool.

$Count_{dttp}$: Total number of data packets at TTP (initialized to zero).

$Count_{rttp}$: Total number of packets (pseudo-random number) at TTP (Initialized to zero).

E_{dttp}: Expected number of data packets at TTP.

E_{rttp}: Expected random number packets at TTP.

Max_Limit_A: Maximum limit of anonymizer. //$Max\_Limit\_A \geq n * t_{pk}$.

Phase 1: (Packetization)

for (j =1 to n) do

begin

a) Each party divides data 'D_j' into 'k' packets;

b) Each party individually generates 'k' pseudo-random number;

for (i =1 to k) do

begin

c) Add each D_{ji} to r_{ji} ;

// here $\sum_{j=1,i=1}^{n,k} D_{ji} = \sum_{i=1}^{k} (D_{ji} + r_{ji})$;

d) Forward (D_{ji}) to randomly selected anonymizer; //each data packet to different anonymizer.

e) $r_j = r_j + r_{ji}$;

end; // end of loop i.

f) Forward r_j to randomly selected anonymizer; // here paseudo-random number and data packets should not be given to same anonymizer.

end; //end of loop j.

Phase 2: (Anonymization)

for (i =1 to n) do

begin

for (j =1 to k) do

begin

a) Randomly select one anonymizer Am_d ;

b) if (count(Am_d) < *Max_Limit_A*) then

begin

c) send D_{ij} to Am_d;

d) increase count(Am_d) by 1;

else

f) Repeat step 2(a);

endif; //end of if statement.

end; //end of loop j

g) Randomly select an anonymizer Amr;

if (count(Am_r) < *Max_Limit_A*) then

begin

i) send r_j to Am_r;

j) Increase count(Am_r) by 1;

else

l) Repeat step 2(g)

endif;// end of if statement.

end; // end of loop $_i$.

Phase 3: (Receive Data and Pseudo-Random number at TTP)

a) for ($_i$ = 1 to $_m$) do

begin

while (count(Am_i) >= 1) do

begin

c) Send data packets to S_D;

d) $Count_{dttp} = Count_{dttp} + 1$;

e) Send pseudo-random number packets to S_R;

f) $Count_{rttp} = Count_{rttp} + 1$;

end;

end;

Phase 4: (Data Validation and Computation)

a) $E_{dttp} = n \times k$;

$E_{rttp} = n$;

If ($Count_{dttp} = E_{dttp}$ and $Count_{rttp} = E_{rttp}$) then

Begin

c) $f(z) = S_D - S_R$;

d) Broadcast$f(z)$;

end;

else

f) Return “Packet Lost”;

g) endif; // end of if

4.6 SUMMARY

The protocols proposed in this chapter provide efficient empirical solutions for SMC issues in case of larger number of parties. However there are theoretical protocols which could be used to solve the SMC issues but implementation of those protocols is not possible due to their complexity. The protocols presented in this chapter are using extra layer to solve the SMC problem using anonymity. This Chapter presented new architectures which enables SMC by hiding the identity of the parties participating in collaborative computations. In this each layer has predefined functioning that enables parties to break huge data into packets to enhance confidentiality.

When this protocol was analyzed with parties and equal number of anonymizers, it concluded that there may be a security breach. It means

that as the number of malicious anonymizer increases then the security loss probability increase linearly.

The chances of data leakage of a party will be very high in case of protocols where fixed anonymizers are used to provide all its data. This type of protocol will be of low cost and easy to implement. In this case even a party can implement its own anonymizer. This protocol is suitable for the area where party does not want to disclose their identities and the anonymizer will be in the full control of parties. The key disadvantage in such type of system is that, the party has to maintain the anonymization process. In the research work instead of fixed anonymizer for each party, independent anonymizers are considered to improve overall protocol functioning. This enables the freedom of random selection of anonymizer. One main advantage of the protocols presented is, if any anonymizer fails then other anonymizers are available for the party. Other advantages of these protocols are, if a party finds that an anonymizer is malicious then the party can ignore it. As multiple anonymizers are used and selection is done randomly it increases overall security of the collaborative computations.