Reflection: A Lightweight Protocol for Private Matching

Mahmood Ahmad *, Zeeshan Pervez †, YongIK Yoon ‡, Byeong Ho Kang,§, Sungyoung Lee *
*Department of Computer Engineering, Kyung Hee University,South Korea
†School of Computing, University of the West of Scotland, Paisley, PA1 2BE, United Kingdom
‡Sookmyung Womens University, South Korea
§School of Computing and Information Systems, University of Tasmania, Australia
Email: rayemahmood *, zeeshan.pervez†, Yiyoon‡, Byeong.Kang§,sylee*@ *oslab.khu.ac.kr,†uws.ac.uk, ‡sm.ac.kr,§utas.edu.au

***Abstract*—Applications of private matching (PM) are not limited to any specific domain of interest. Social media, e-health and commerce applications can afford to share certain common information without disclosing any extra details. PM ensures to reveal only common information between the communicating parties. In this paper we have presented a light weight protocol named Reflection for PM. Depending upon different requirements of involving parties, Reflection can be used in presence of third party or as a point to point communication. Using services of third party (cloud) where communication has to route through it, it still preserves the privacy aspect of values being matched. The protocol works on a random number and produces the result of matching on behalf of original values, hence it converges in only two pass communication. It also protects any additional slipway of information during or after the PM. It safeguards the user communication against eavesdropper and even from offline attacks. In its simpler version, the protocol can work without using any encryption. Value comparison mechanism in presence of encryption, makes the overall process of PM more secure against hostile users.**

***Keywords*-Private Matching; Security and Privacy;Cloud computing**

I. INTRODUCTION

In present era of digital world, private matching (PM) has a large domain of applications where it is used for sharing the common information. Applications of PM are not limited to any specific scenario but are quite essential in Social media [1],[2] business community [3] and e-health applications [4]. Ensuring privacy aspect of shared information is the core idea for any PM protocol. In general, PM protocol is a set of operations through which two parties learns their common values without disclosing any additional information. The solution for (PM) has been introduced by Freedman et al [5]. According to its definition, if two parties, Alice and Bob having set of values $A = \{a_1, a_2...a_n\}$ and $B = \{b_1, b_2...b_n\}$ respectively, then at the end of protocol they should only learn which values they have in common and nothing else as given in equation 1.

$$A \cap B =: \{a_u | \exists_v, a_u = b_v\} \leftarrow PM(A, B) \qquad (1)$$

For private matching there exist more than one solutions depending upon the requirements of involving parties. Considering Alice and Bob as involving parties and Eve as a third entity, these assumptions are usually based on following scenarios.

- Only Alice is interested to know either she has any value in common with Bob or not.
- Both, Alice and Bob are interested to know about their common value(s)
- If Alice learns the outcome of PM at first place then she is expected to share it with Bob. To ensure honesty on her share, services of third party may be required.
- If Eve is acting as third entity between Alice and Bob then she can also learn about values being matched. Using services of third party can disclose information on matching values, that is not desired by Alice or Bob.
- At times, PM is required only to know about magnitude of similarity rather contents similarity.

Through private matching the involving parties participate with full confidence without worrying about the leakage of un-necessary information. This leakage of information is the awareness about those values which are not common with either party or the set cardinality issue. According to [6], private matching has been classified into three scenarios. In *first scenario* the involving parties Alice and Bob, both should learn the final outcome of PM result which is called symmetric PM. In *second scenario* which is non symmetric, only one party learns if there exist any commonality of values or not. The *third scenario* is knowing about total number of common elements instead of exact values match. All these requirements have been met and addressed using different PM protocols.

Other than variations of PM that have been evolved so far, basic motivation behind PM is to assist two parties to learn only what is required. No additional information should be revealed during or after protocol execution to either parties. Considering an example where Department of Homeland Security (DHS) is looking for suitable candidates from various law enforcement agencies for its new task. To hide details on this task, required skill sets are not advertised publicaly. Similarly, candidates applying for

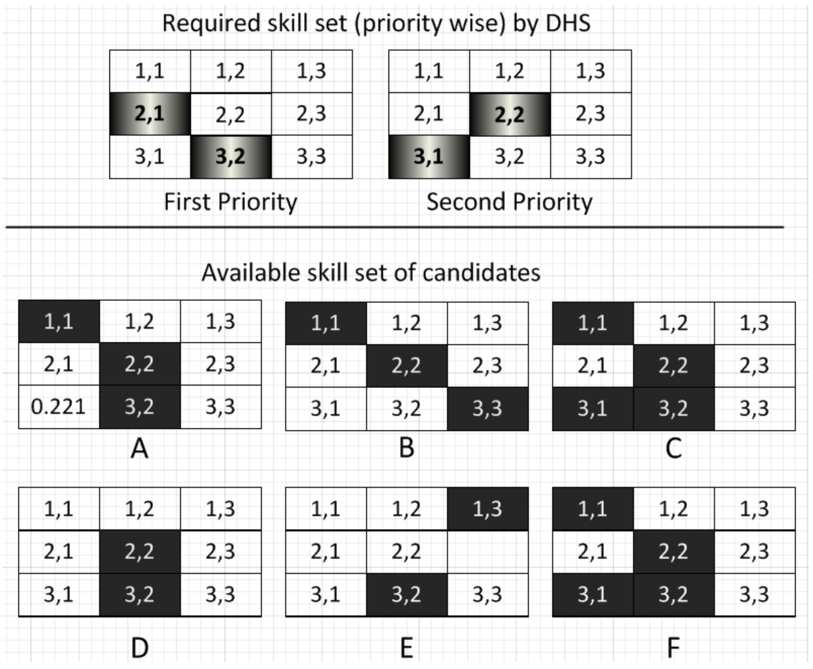
Required skill set (priority wise) by DHS
First Priority
Second Priority
Available skill set of candidates
A
B
C
D
E
F

Figure 1. Selection of suitable candidates from desired and available skill set

this task do not want to reveal those skill sets which are not required by DHS. For this situation PM can preserve privacy aspect from both sides. The selection process will evaluate each candidate on a pair of value, which is represented as $(Skill, ExpertyLevel)$. For simplicity, this pair is represented with numeric values and candidates are represented with alphabet characters from A-to-F. According to task demands, candidates having skill set $\{(2,1),(3,2)\}$ will be considered first. Candidates having skill set $\{(2,2),(3,1)\}$ will be considered if no candidate is found from first priority. Required skill set and skill set of each individual is represented with grey and black boxes respectively in Figure 1.

The selection process can be done in two ways. First, evaluate all applications in a sequence and finalize the selection. This approach has to process all applications no matter they will be finally selected or not. The second solution could be to look for required skill one after the other. The later approach converges quickly and also protects privacy concerns of job applicants. Keeping in view the nature of problem and others similar to it, we have developed a light weight protocol for private matching and name it *Reflection*. It preserves privacy on both sides of communications (involving parties) and converges more quickly. Using *Reflection*, the first PM round will eliminate all candidates, as no one possesses this skill i.e. $(2,1)$. To evaluate candidates on second priority with skill set, $(2,2)$ is matched first. This evaluation will disqualify only candidate E. In further evaluation for $(3,1)$, only C and F get selected such because they meet the required selection criteria i.e. $\mathcal{C}_{\{(2,2),(3,1)\}}$, $\mathcal{F}_{\{(2,2),(3,1)\}}$. With *Reflection*, we have made following contributions in the area of private matching.

- The applications of PM where encryption is an expensive operation, *Reflection* can work without encryption and can still preserve privacy issues of involving parties.
- It is most suitable where decision for matching N^{th} value is dependent on result from $(N-1)^{th}$ match. This approach will narrow down the search space by squeezing the unnecessary matching values thus assuring enhanced privacy.
- At the end of protocol execution, values which might be same but are not required can be saved from being exposed.
- During PM, involving parties may or may not include services of third party. In this case third party will remain unaware from values being matched.
- If $\mathcal{O}$ is the output of PM for two values $\mathcal{V}_1$ and $\mathcal{V}_2$ then another match on same values will not necessarly end up in $\mathcal{O}$. This feature makes *Reflection* resistible against offline attacks.

The rest of the paper is organized as follows.
Section II is about Related work.Section III covers the main idea and its methodology. Discussion on threat model is given in Section IV. Other than P2P communication, the protocal has been modified where thrid party is involved too. Varition of Reflection protocol in context of third party is given in SectionV. Future work is in section VI. Section

VII concludes the paper.

II. Related Work

In very initial work, Freedman et al. presented his idea on private matching[5]. With gradual evolution on PM mechanism various algorithms and techniques have been proposed so far. These variations are mainly due to different requirements by parties involving in PM or it is result of fine tuning to achieve least computational cost. Other than values which are found common, the residual data during or after the PM can assist a curious user to find additional information which is not desirable to be known otherwise. Keeping this in view, we will present the related work which is about PM in general and avoiding slipway of additional information in particular. In recent years the protocol of Private Set Intersection (PSI) [5],[7],[8],[9] and Private Set Union (PSU) [4],[10],[11] are considered as main driving factors in the area of PM. In [12], these two approaches have been highlighted for loosing adequate privacy on server end. The Private Set Intersection Cardinality (PSI-CA) or Private Set Union Cardinality (PSU-CA) [12] is for a scenario where clients are only allowed to learn the magnitude of common values rather the exact values. This approach encourages to find out total number of matching values first and that too without knowing them. After executing PSI-CA or PSU-CA the involving parties can further decide either to go for value matching protocol or not.

With PM protocol, values that are found common can lead for record linkage problem. The linkage works with attempts to match records stored at distinct parties (e.g., hospitals), but which represent the same entity (e.g.,patient). Records found at one place can be matched with record on other place. To handel this issue two approaches have been proposed which is the sensitization [6],[13] and secure multiparty communication (SMC)[14]. Total reliance on first approach is not effective in terms of accuracy. Results acquired from santized matching can end up in false positives. On the other hand SMC overcomes this issue but with computational overhead. The computational cost of SMC operations performed in [5],[15],[7] end up in $O(m * n)$ cryptographic operations. If $m = n = 100$ then integration of this task will require 10^8 cryptographic operations. The idea on santized query has been upgraded to differential privacy in [16]. With this technique, instead of sending the perturbed records after sensitization, user query is evaluated statistically. For detailed description on differential privacy reader may refer to [16],[17]

In [18] , Agrawal et al. proposed a model for private matching under the provision of encryption function $\mathcal{E}$ and $\mathcal{E}'$ such that $x \in X, \mathcal{E}(\mathcal{E}'(x)) = \mathcal{E}'(\mathcal{E}(x))$. Matching values using this approach will reveal common elements to only one party at first place. Now it is the discretion of that party who has knowledge of common elements whether to share it with other party or not. Even if sharing is done, there is another question that either all elements are shared honestly or not. Besides, using double encryption Agrawal, et al suggested to use simple hashing mechanism to find out the common elements between two parties. Matching common elements with hashing technique will definitely require a common hash function at first place to be decided between the involving parties. Besides common elements found using hash technique, $H(A) \cap H(B)$, a curious party can employ a brute force attempt using same hash function to find out those elements which are not common over a finite domain of elements. Hashing techniques are not appropriate in PM as they are more towards ensuring integrity of file contents.

PM protocols are not limited with client server communication. Utilizing services of third entity which could be a cloud or semi trusted third party (TTP) is another option to consider. Using services of third entity can overcome the issue where both parties need to know common values at the same time. The technique of Oblivious Polynomial Evaluation (OPE) using the homomorphic encryption [19] can assist to achieve this model as given in [5]. It utilizes the roots and coefficients of polynomial. Encrypted coefficients are evaluated by cloud server or any TTP to find if values hold equality or not. The output which is revealed at server or third party gives no clue for actual value being matched, no matter equality holds or not. Using the same principles of polynomial we have used it in such a way that instead of evaluating polynomial $P(x)$ on valid roots, it will be evaluated on random value. This technique works even without employing encryption and gives no clue about the value being matched. The proposed technique can work in both scenarios i.e, with or without using encryption. The *Reflection* protocol has achieved a notable performance in terms of communication overhead and local execution time. In terms of communication, it takes place only twice. For execution, the complexity grows linearly. No matter final result ends up in exact match or not, actual values can not be traced back from the communication.

III. Main Idea

The motivation behind *Reflection* protocol is to match values with minimum communications and least computations. In its basic version, we have evaluated it without using any encryption mechanism. In existing PM protocols, encryption or hashing techniques are applied on actual values to be matched whereas in proposed protocol we generate random values on behalf of actual values to find the exact match. This indirect approach is tolerable in terms of execution time and from a security perspective, the random values cannot be traced back to their original values. This artifact survived to find matching between two values without using encryption. In extended version of *Reflection*, encryption can be used on these random values. Using encryption will provide an additional layer of security against curious or malicious users. In case, the decryption mechanism is a successful attempt by a hostile user, still the output will end up in a random value and will give no clue on original values.

A. Notations and Assumptions

Table I
NOTATIONS USED IN THE DESCRIPTIVE DETAIL OF *Reflection* PROTOCOL

Notation	Description
$V = \{v_1, v_2, .., v_n\}$	Set of values to be matched
v_i	A particular value instance v_i, where $v_i \in V$
$\mathcal{P}(.)$	Polynomial created by using r and $(r+v_i)$, where r is a random number
(Ω, Ψ)	Pair of values exchanged between users during the PM process. First parameter Ω is coffeicient identifier which is either low or $high$. Value of second parameter Ψ depends upon the value of Ω, such that $\Psi = \begin{cases} (r) & \text{if } \{\Omega_{low}\} \\ (r+v_i) & \text{if } \{\Omega_{high}\} \end{cases}$
Δ	Final value calculated by each user using *Reflection*. This value will help to know either values are same or not
$\mathcal{M}$	Message generated by each user for symmetric encryption
$\mathcal{E}_s$, $\mathcal{D}_s$	Symmetric encryption and decryption algorithms.

The notations used in *Reflection* protocol are given in Table I. Values to be matched are taken from a finite set V, and a polynomial for value evaluation by each user is represented by $P(.)$. Coefficient identifier is represented by Ω, which is shared by each party during the PM. Final value which will discover either the orginal values being matched are same or not is represented by Δ. The only assumption of *Reflection* protocol is, if one party sends Ω_{lower} then the other party will send Ω_{high}. Although the protocol will still work without this assumption, but in that case it will end up in two different values for Δ. For these two different values, two comparison would be required.

B. Methodology

In proposed model for private matching we build basis for its solution using the polynomial equations. Under certain mechanism, two different polynomials are created by each user who are participating in private matching. Considering Alice and Bob as involving parties, having values $v1_{Alice}$ [1] and $v2_{Bob}$ respectively, and are interested to privately match their values. The processes begins by creating a random number, both user will generate their own random numbers i.e. r_{Alice}, r_{Bob} and create a polynomial $P_{Alice}(.)$ and $P_{Bob}(.)$. The polynomial is created by using r and v_i, which are also valid roots for this $P(.)$. Next, both user will share (Ω, Ψ) with each other. After that, Alice and Bob will evaluate their respective $P(.)$ on these parameters. If Alice has received first parameter as Ω_{low} then $P_{Alice}(.)$ is evaluated on r_{Bon} and $(v1_{Alice} + r_{Bob})$ or, on $(r_{Bob} + v2_{Bob})$ and $((r_{Bob} + v2_{Bob}) - v1_{Alice})$ in case of Ω_{high}. Similarly Bob will compute result of his $P_{Bob}(.)$. So far, Alice and Bob have communicated only once and have come up with result of their polynomials, Δ_{Alice} and Δ_{Bob}. Both parties can now share their results, and if($v1_{Alice} == v2_{Bob}$) then Δ_{Alice} and Δ_{Bob} will be same or otherwise. In both cases, value result of each Δ is not equal to $v1_{Alice}$ or $v2_{Bob}$, which protects the privacy of actual numbers. The second value which is communicated in *Reflection* is Δ.

If algorithm is repeated again for the same value i.e. $v1_{Alice}$ and $v2_{Bob}$, it will always end up in different value i.e Δ. Variation in value of Δ with each comparison is independent either values being matched are same or not. The steps involved in our design follows Algorithm 1.

```
Algorithm 1: REFLECTION: Private matching protocol
  Input: Random number $\mathcal{R}$,Value to be Matched $\mathcal{V}$,
         shared information $\Psi$, Higher and Lower bits
         for Cofficient $\Omega_{high}, \Omega_{low}$
  Output: Final evaluated Result $\Delta$
1  $\mathcal{R} \leftarrow GenerateRandomNumber()$
2  $\mathcal{V} \leftarrow ValuetobeMatched$
3  //Create polynomial $P(x)$
4  $P(x) = (X - R)(X - (R + \mathcal{V}))$
5  //Select coefficient to share
6  if ($lowercofficient$) then
7      $\Psi = r$
8      $send(\Psi, \Omega_{Low})$
9  else
10     $\Psi = r + \mathcal{V}$
11     $send(\Psi, \Omega_{high})$
12 //Receive value and coefficient marker from other party
13 receive($\Psi$,$\Omega$)
14 //evaluate $P(x)$ on $\Psi$
15 eval$_1$=$P(x) \leftarrow \Psi$
16 if ($\Omega_{low}$) then
17     eval$_2$=$P(x) \leftarrow (\Psi + \mathcal{V})$
18 else
19     eval$_2$=$P(x) \leftarrow (\Psi - \mathcal{V})$
20 $\Delta$=eval$_1$+eval$_2$
21 return $\Delta$
```

IV. THREAT MODEL

In this section we will present two possible threat models which may occur during or after the PM

[1]Any notation with a subscript means, it is generated or used by that user

A. Protection Against an Eavesdropper

The first category of threat model is during the PM process. Malory is a curious user, who is interested to know about values that are undergoing in a PM process between Alice and Bob. Malory intercepts the communication and successfully acquires the network traffic packet containing r_{Alice}, Δ_{Alice}, r_{Bob}, and Δ_{Bob}. This information will not help Malory to know exact values of $v1_{Alice}$ or $v2_{Bob}$ or any extra information. This assumption holds only if $(v1_{Alice} \neq v2_{Bob})$. In case, $(v1_{Alice} == v2_{Bob})$, values of Δ_{Alice} and Δ_{Bob} will be same. Although Malory still cannot discover the original values, however; equality of Δ will reveal that Alice and Bob have similar values.

To avoid this additional leakage of information, we have modified the second (last) communication of *Reflection* protocol. Instead of sharing Δ as a result , encrypted (symmetric encryption) results will be shared instead. For this purpose value of Δ is used as a symmetric key by each user. If both user have same value of Δ then they will be able to recover the original message or otherwise. For this reason, we have used Δ as a symmetric key. Construction of message that will be symmetrically encrypted by each user is given in Algorithm 2.

Algorithm 2: REFLECTION: Message Construction

```
  Input: $\Psi_y$
  Output: Final Message $\mathcal{M}_x$
1 //Create Message $\mathcal{M}$
2 $\mathcal{M}_x = \Delta_x * \Psi_y$
3 return $\mathcal{M}_x$
```

Input for this algorithm which is Ψ, is value received from other user. If Alice is constructing her message, then she will use Ψ_{Bob}. Using value of Ψ received from other user will ensure that Alice and Bob create a different message such that $\mathcal{M}_{Alice} \neq \mathcal{M}_{Bob}$. These two different messages will produce two different encrypted outputs even with similar value of Δ as a symmetric key, as given in equation 2.

$$\mathcal{E}_s(\mathcal{M}_{Alice}, \Delta_{Alice}) \neq \mathcal{E}_s(\mathcal{M}_{Bob}, \Delta_{Bob}) \tag{2}$$

This dissimilarity will be hard for Eve to discover that Alice and Bob have similar values or not. When this encrypted output is received, it is decrypted using $\mathcal{D}_s$ to recover the message. For instance, Bob will perform following step given in equation 3, to find if his value is same with Alice or not

$$\mathcal{D}_s = ((\mathcal{E}_s(\mathcal{M}_{Alice}, \Delta_{Alice})), \Delta_{Bob}) \div \Psi_{Bob} \tag{3}$$

If output of equation 3 is equal to Δ_{Bob}, it means Bob and Alice have similar values. Similarly Alice will perform the same process to discover the final result.

B. Offline Attack

Offline attack occurs after the PM process has achieved its conclusion and the final result is a mismatch. To reach this conclusion, some value has to be exchanged between the involving parties. If hashing or encryption is applied during the PM process, a curious party can perform the exhaustive approach to discover this mismatch. If $\mathcal{H}$ is a hashing functions then $\mathcal{H}(v_i)$ will always end up in same hash output. Similarly, if $\mathcal{E}$ is an encryption function with constant key k, then $\mathcal{E}(v_i, k)$ will always end up in similar encrypted output. If hashing or encryption is applied during the private matching process, the mismatch value is susceptible to offline attacks. With offline attack, we mean that the malicious party who is interested to guess that particular value, does not have to communicate with the other party again and again. This mismatched value can be tested with exhaustive approach by using different hashing algorithms or encryption mechanism on a finite set of values. The value which will produce same hashed or encrypted output reveals the original value. In case of an encryption function that uses multiple keys such that $\mathcal{E}(v, \mathcal{K}_1) = o_1$ and $\mathcal{E}(v, \mathcal{K}_2) = o_2$ where o_1 and o_2 are two different outputs, then it is more resisting for offline attacks. Similary in *Reflection*, although we have not opted encryption with different keys, but effect of randomization makes *Reflection* resistable against offline attacks due to different value output of Δ.

V. Reflection: In presence of third party

Other than using *Reflection* in point to point communication, services of third party can be used easily without disclosing any information. The purpose of using third party services is to facilitate both users to know about the final outcome at the same time. Just like in point to point communication, the involving parties will share their random number and encrypted final value through third party.

VI. Future Work

In this paper we have presented a light weight protocol for private matching between two parties. It works for point to point as well as in presence of third party. In its present design, the protocol works for exact match and cannot identify which value is bigger than the other, in case of inexact match. In our future work we will extend our idea in finding which value is greater than the other without revealing the actual value.

VII. Conclusion

Private matching has attracted lot more attention in recent years due to large number of applications and their privacy concern while exchanging information with each other. Applications of private matching is desired from point to point communication and in presence of third entity as well. In point to point communication users can adopt Reflection protocol for private matching as it has been tested efficient in terms of computational resources and security. We have also tested same protocol for situations where two users can utilize services of semi honest trusted third party. Sharing final encrypted output with dynamically created symmetric key makes *Reflection* resistible for curious users.

ACKNOWLEDGMENT

This research was funded by the MSIP(Ministry of Science, ICT & Future Planning), Korea in the ICT R&D Program 2013

REFERENCES

[1] M. Li, N. Cao, S. Yu, and W. Lou, "Findu: Privacy-preserving personal profile matching in mobile social networks," in *INFOCOM, 2011 Proceedings IEEE*. IEEE, 2011, pp. 2435–2443.

[2] R. Zhang, Y. Zhang, J. Sun, and G. Yan, "Fine-grained private matching for proximity-based mobile social networking," in *INFOCOM, 2012 Proceedings IEEE*. IEEE, 2012, pp. 1969–1977.

[3] Q. Dai and R. J. Kauffman, "Business models for internet-based e-procurement systems and b2b electronic markets: an exploratory assessment," in *System Sciences, 2001. Proceedings of the 34th Annual Hawaii International Conference on*. IEEE, 2001, pp. 10–pp.

[4] K. Frikken, "Privacy-preserving set union," in *Applied Cryptography and Network Security*. Springer, 2007, pp. 237–252.

[5] M. J. Freedman, K. Nissim, and B. Pinkas, "Efficient private matching and set intersection," in *Advances in Cryptology-EUROCRYPT 2004*. Springer, 2004, pp. 1–19.

[6] Y. Li, J. Tygar, and J. M. Hellerstein, "Private matching," in *Computer Security in the 21st Century*. Springer, 2005, pp. 25–50.

[7] L. Kissner and D. Song, "Privacy-preserving set operations," in *Advances in Cryptology–CRYPTO 2005*. Springer, 2005, pp. 241–257.

[8] S. Jarecki and X. Liu, "Efficient oblivious pseudorandom function with applications to adaptive ot and secure computation of set intersection," in *Theory of Cryptography*. Springer, 2009, pp. 577–594.

[9] C. Hazay and Y. Lindell, "Efficient protocols for set intersection and pattern matching with security against malicious and covert adversaries," in *Theory of Cryptography*. Springer, 2008, pp. 155–175.

[10] C. Hazay and K. Nissim, "Efficient set operations in the presence of malicious adversaries," *Journal of cryptology*, vol. 25, no. 3, pp. 383–433, 2012.

[11] J. Hong, J. W. Kim, J. Kim, K. Park, and J. H. Cheon, "Constant-round privacy preserving multiset union." *IACR Cryptology ePrint Archive*, vol. 2011, p. 138, 2011.

[12] E. De Cristofaro, P. Gasti, and G. Tsudik, "Fast and private computation of cardinality of set intersection and union," in *Cryptology and Network Security*. Springer, 2012, pp. 218–231.

[13] L. Sweeney, "k-anonymity: A model for protecting privacy," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 10, no. 05, pp. 557–570, 2002.

[14] O. Goldreich, *Foundations of Cryptography: Volume 2, Basic Applications*. Cambridge university press, 2009, vol. 2.

[15] J. Vaidya and C. Clifton, "Secure set intersection cardinality with application to association rule mining," *Journal of Computer Security*, vol. 13, no. 4, pp. 593–622, 2005.

[16] C. Dwork, "Differential privacy," in *Automata, languages and programming*. Springer, 2006, pp. 1–12.

[17] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory of Cryptography*. Springer, 2006, pp. 265–284.

[18] R. Agrawal, A. Evfimievski, and R. Srikant, "Information sharing across private databases," in *Proceedings of the 2003 ACM SIGMOD international conference on Management of data*. ACM, 2003, pp. 86–97.

[19] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Advances in cryptologyEUROCRYPT99*. Springer, 1999, pp. 223–238.