Evaluation of Encryption Algorithms for Privacy Preserving Association Rules Mining

Ashraf B. El-Sisi and Hamdy M. Mousa
(Corresponding author: Ashraf B. El-Sisi)

Computer Science Department, Faculty of Computers and Information,
Menoufiya University Gamal Abd-ElNaser St., Shebin Elkom 32511, Egypt
(Email: ashrafelsisi@hotmail.com, hamdimmm@hotmail.com)

(Received Mar. 06, 2012; revised and accepted Jun 27, 2012)

Abstract

Encryption algorithms used in privacy preserving protocols can be affected on overall performance. In this paper we study several encryption algorithms with two methods of privacy preserving association rule mining on distributed horizontal database (PPARM4, and PPARM3). The first method PPARM4 computes association rules that hold globally while limiting the information shared about each site in order to increase the efficiency. The second method PPARM3 is a modification for PPARM4 based on a semi-honest model with negligible collision probability. Common encryption algorithms for the two methods of privacy preserving association rule mining on distributed horizontal database selected based on performance metric. So a performance comparison among five of the most common encryption algorithms: RSA, DES, 3DES, AES and Blowfish with the two privacy methods are presented. The comparison has been conducted by running several encryption settings with the two methods of privacy preserving association rule mining on distributed horizontal database. Simulation has been conducted using Java. Results show that, PPARM3 gives better performance with all encryption algorithms implemented. Also PPARM3 with encryption algorithm DES gives best result with different database sizes. Based on the results we can tune the suitable encryption algorithm from our implementations to the required overall performance.

Keywords: Association rules mining, distributed data mining, encryption, security, privacy

1 Introduction

Data Mining (DM) techniques have been widely used in many areas especially for strategic decision-making [1, 2, 17, 19, 20, 21, 28, 33]. Apart from its usual benefits, it also has a few disadvantages associated with it. Experts say that data mining in the wrong hands will end up in destruction. The main threat of data mining is to privacy and security of data residing in large data stores [3, 4, 6, 7, 9, 16, 27]. Some of the information considered as private and secret can be brought out with advanced data mining tools. It is a real concern of people working in the field of database technology. Different research efforts are under way to address this problem of preserving security and privacy. The privacy term is overloaded, and can, in general, assume a wide range of different meanings. For example, in the context of the Health Insurance Portability and Accountability Act (HIPAA) Privacy Rule, privacy means the individual's ability to control who has the access to personal health care information. From the organizations point of view, privacy involves the definition of policies stating which information is collected, how it is used, and how customers are informed and involved in this process. We can considering privacy as "The right of an entity to be secure from unauthorized disclosure of sensible information that are contained in an electronic repository or that can be derived as aggregate and complex information from data stored in an electronic repository". There are many methods for privacy preserving distributed association rule mining across private databases. So these methods try to compute the answer to the mining without revealing any additional information about user privacy. An application that needs privacy preserving distributed association rule mining across private databases, like medical research. Sensitive information contained in a database can be extracted with the help of non-sensitive information. This is called the inference problem. Different concepts have been proposed to handle the inference problem. The process of modifying the transactional database to hide some sensitive information is called sanitization. By sanitizing the original transactional database, the sensitive information can be hidden. In the sanitization process, selective transactions are retrieved and modified before handing over the database to a third party. Modification of transaction involves removing an item from a transaction or adding an element to the transaction. In some cases, transactions will be either added to or removed from the database as suggested in [8]. The modified database is called sanitized database or released database. The efficiency of a privacy-preserving algorithm is measured based on: (1) the time taken to hide the data, (2) the number of new rules introduced because of the hiding process, and (3) the number of legitimate rules lost or unable to be extracted from the released database. Encryption algorithm used in privacy preserving can be affected on overall performance, so in this paper we addresses the problem of evaluate several encryption algorithms with two protocols of privacy preserving association rule mining (PPARM4, and PPARM3) on distributed horizontal database. The numbers (4 and 3) in abbreviations (PPARM4, and PPARM3) respectively means the number of computation steps to get the results of protocol. The first protocol (PPARM4) computes association rules that hold globally while limiting the information shared about each site in order to increase the efficiency [15]. The second protocol (PPARM3) is a modification for the first based on a semi-honest model with negligible collision probability [14]. The organization of this paper as following, Section 2 gives an overview about the problem and the related work in the area of privacy preserving association rule mining on distributed homogenous databases. In Section 3 some details of the two protocols of the algorithms of computing the distributed association rule mining (PPARM4, and PPARM3) to preserve the privacy of users. Sections 4 and 5 describe implementation and results of several encryption algorithms with the two methods of privacy preserving association rule mining on distributed horizontal database respectively. Finally, some conclusions are put forward in Section 6.

2 Distributed Association Rule Mining Relatd Work

In this section, we will discuss the problem of computing association rules within a horizontally partitioned database. We assume homogeneous databases. All sites have the same schema, but each site has information on different entities. The goal is to produce association rules that hold globally, while limiting the information shared about each site to preserve the privacy of data in each site.

2.1 Association Rule Mining

Association rule mining finds interesting associations and/or correlation relationships among large sets of data items. Association rules show attributes value conditions that occur frequently together in a given dataset.

In [5] the association rules mining problem can formally be defined as follows: Let I = {i1, i2, . . . , in} be a set of items. Let DB be a set of transactions, where each transaction T is an itemset such that T $\subseteq$ I. Given an itemset A $\subseteq$ I, a transaction T contains A if and only if A $\subseteq$ T. An association rule is an implication of the form A $\rightarrow$ Y where A $\subseteq$ I, B $\subseteq$ I and A $\cap$ B =$\emptyset$. The rule A $\rightarrow$ B has support S in the transaction database DB if S% of transactions in DB contains A $\cup$ B. The association rule holds in the transaction database DB with confidence C if C% of transactions in DB that contain A also contains B. An itemset X with k items is called a k-itemset.

2.2 Distributed Association Rule Mining Problem

The problem of mining association rules is to find all rules whose support and confidence are higher than certain user specified minimum support and confidence. Clearly, computing association rules without disclosing individual transactions is straightforward. We can compute the global support and confidence of an association rule AB $\rightarrow$ C knowing only the local supports of AB and ABC, and the size of each database:

$$Support_{AB \Rightarrow C} = \frac{\sum_{i=1}^{numberofsites} Support\_count_{ABC}(i)}{\sum_{i=1}^{numberofsites} database\_Size(i)} \tag{1}$$

$$Support_{AB} = \frac{\sum_{i=1}^{numberofsites} Support\_count_{AB}(i)}{\sum_{i=1}^{numberofsites} database\_Size(i)} \tag{2}$$

$$Confidence_{AB \Rightarrow C} = \frac{Support_{AB \Rightarrow C}}{Support_{AB}} \tag{3}$$

Note that this requires no sharing of any individual transactions. And protects individual data privacy, but it does require that each site disclose what rules it supports, and how much it supports each potential global rule. What if this information is sensitive? Clearly, such an approach will be secure under secure muti-party computation (SMC) definitions by some modification, a way to convert the above simple distributed method to a secure method in SMC model is to use secure summation and comparison methods to check whether threshold are satisfied for every potential itemset [14]. For example, for every possible candidate 1-itemset, we can use the secure summation and comparison protocol to check whether the threshold is satisfied. Figure 1 gives an example of testing if itemset ABC is globally supported, it shows determining if itemset support exceeds 5 percent threshold. Each site first computes its local support for ABC, or specifically the number of itemsets by which its support exceeds the minimum support threshold (which may be negative). The parties then use the secure summation algorithm (the first site adds a random (R) to its local excess support, then passes it to the next site to add its excess support, etc. and finally when pass to first site subtract the generated random from the result). The only change is the final step, the last site performs a secure comparison with the first site to see if the sum ≥ R. In the example, R -10 is passed to the second site, which adds its excess support (5) and passes it to site 3. Site 3 adds its excess support; the resulting value (22) is tested using secure comparison to see if it exceeds the Random value (21). It is, so itemsets ABC is supported globally.

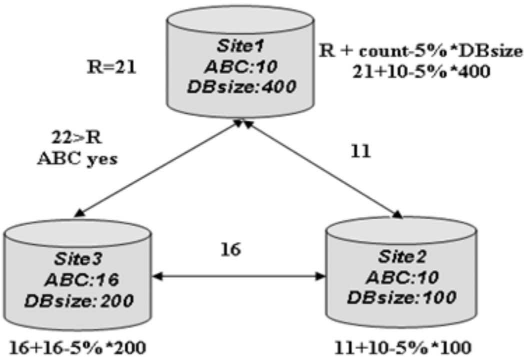

Figure 1: Computing global support securely

Due to huge number of potential candidate itemsets, we need to have a more efficient method. This can be done by observing this lemma, (If a rule has support > k% globally, it must have support > k% on at least one of the individual sites). A distributed algorithm for this would work as follows, request that all rules are sent by each site with support at least k, for each rule returned, request that all sites send the count for their transactions that support the rule, and the total count of all transactions at the site. From this, we can compute the global support of each rule, and be certain that all rules with support at least k have been found. This has been shown to be an effective pruning technique. In order to use the above lemma, we need to compute the union of locally large sets. Then use the secure summation and comparison only on the candidate itemsets contained in the union. Revealing candidate itemsets means that the algorithm is no longer fully secure: itemsets that are large at one site, but not globally large, would not be disclosed by a fully secure algorithm. However, by computing the union securely, we prevent disclosure of which site, or even how many sites, support a particular itemset. This release of innocuous information (included in the final result) enables a completely secure algorithm that approaches the efficiency of insecure distributed association rule mining algorithms. The function now being computed reveals more information than the original association rule mining function. However, the key is that we have provable limits on what is disclosed.

3 (PPARM4, and PPARM3) Protocols Details

In secure multi-party computation approaches, given two parties with inputs x and y respectively, the goal of secure multi-party computation is to compute a function f(x, y) such that the two parties learn only f(x, y), and nothing else. In [5] there are various approaches to this problem. In [18] an efficient protocol for Yao's millionaires' problem showed that any multi-party computation can be solved by building a combinatorial circuit, and simulating that circuit. A variant of Yao's protocol is presented in [24] where the oblivious transfers is used to make secure decision tree learning using ID3 with efficient cryptographic protocol and their also two solution of our problem under the secure multi party computation for association rule mining [29], [26]. In [26] an explanation of a much more efficient method for this problem is described. To obtain an efficient solution without revealing what each site supports, they instead exchange locally large itemsets in a way that obscures the source of each itemset. They assume a secure commutative encryption algorithm with negligible collision probability. Intuitively, under commutative encryption, the order of encryption does not matter. If a plaintext message is encrypted by two different keys in a different order, it will be mapped to the same cipher text. Formally, commutatively ensures that Ek1(Ek2(x)) = Ek2(Ek1(x)). The main idea is that each site encrypts the locally supported itemsets, along with enough "fake" itemsets to hide the actual number supported. Each site then encrypts the itemsets from other sites. An example illustrate the protocol in [26] is given in Figure 2. Using commutative encryption, each party encrypts its own frequent itemsets (e.g., Site 1 encrypts itemset ABC). The encrypted itemsets are then passed to other parties, until all parties have encrypted all itemsets. These are passed to a common party to eliminate duplicates, and to begin decryption. (In Figure 2, the full set of itemsets is shown to the left of Site 1, after Site 1 decrypts). Then this set is passed to each party, and each party decrypts each itemset. The final result is the common itemsets (ABC and ABD in Figure 2), an approach to proof that protocol preserves privacy can be found in [18]. This approach to prove that algorithm reveals only the union of locally large itemsets and a clearly bounded set of innocuous information.

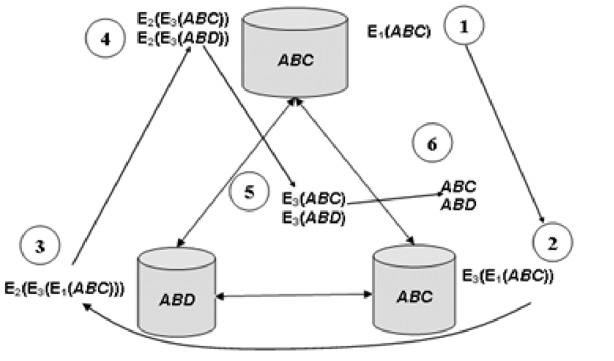

Figure 2: Steps needed for computing the algorithm in [26]

3.1 PPARM4 Protocol

Other method for privacy preserving association rule mining on distributed homogenous databases (PPARM4) in [15], showed that the protocol in [26] employs commutative encryption algorithm so it adds large overhead to the mining process then another protocol improves this by applying a public-key cryptosystem algorithm on horizontally partitioned data among three or more parties. In this protocol, the parties can share the union of their data without the need for an outside trusted party. Each party works locally finding all local frequent itemsets of all sizes. Then use public key cryptography to find the union of a frequent local itemset. We find that this method reduce the number of steps from 6 to 4 to calculate the global candidate item sets as shown in Figure 3 where K1 is private key and k2 public key . In [15] the results showed that this improvement reduces the time of mining process compared to method in [26].

For description this protocol, let P = {P0, . . . , Pn} be a set of N parties where |N| ≥ 3. Each party Pi has a database DBi. With assuming that parties running the protocol are semi-honest. The goal is to share the union of DBi as one shuffled database $DB_{Comp} = \bigcup_{i=0}^{n} DB_i$ and hide the link between records in DB_{Comp} and their owners. This employs a public-key cryptosystem algorithm on horizontally partitioned data among three or more parties. In this protocol, the parties can share the union of their data without the need for an outside trusted party. The information that is hidden is what data records where in the possession of which party. Protocol is described for one party as the protocol driver as shown in Table 1. The first party called Alice.

Table 1: Description of PPARM4 protocol

1. *Alice generates a public encryption key kPA.* Alice makes kPA known to all parties (for illustration another two parties called Bob and Carol can be used).

2. *Each party (including Alice) encrypts its database DBi with Alice's public key.* This means the encryption is applied to each row (record or transaction) of the database. Parties will need to know the common length of rows in the database. We denote the result of this encryption as kPA(DBi) (refer to Figure 4). Note that, by the properties of public cryptosystems, only Alice can decrypt these databases.

3. *Alice passes her encrypted transactions kPA(DB1) to Bob*. Bob cannot learn Alice's transactions since he does not know the decryption key (Figure 5).

4. *Bob mixes his transactions with Alice's transactions.*That is, he produces a random shuffle of kPA(DB1) and kPA(DB2) before passing all these shuffled transactions to Carol (see Figure 6).

5. *Carol adds and shuffles her transactions kPA(DB3) to the transactions received from Bob.*

6. *The protocol continues in this way, each subsequent party receiving a database with the encrypted and shuffled transaction of all previous parties in the enumeration of the parties. The i-th party mixes randomly his encrypted transactions kPA(DBi) with the rest and passes the entries shuffled transaction to the (i + 1)-th party.*

7. *The last party passes the transactions back to Alice* (see Figure 7).

8. *Alice decrypts the complete set of transaction with her secret decrypt key.*

She can identify her own transactions. However, Alice is unable to link transactions with their owners because transactions are shuffled.

9. *Alice publishes the transactions to all parties.* If the number of parties is N, then N − 1 of the parties need to collude to associate data to their original owners (data suppliers).

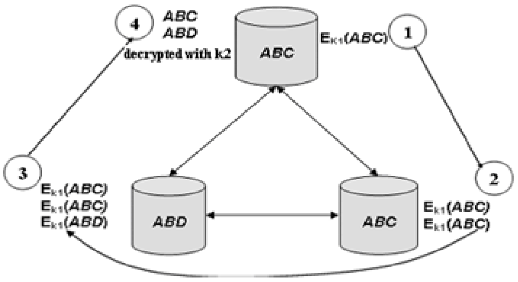

Figure 3: Steps needed for computing PPARM4 algorithm

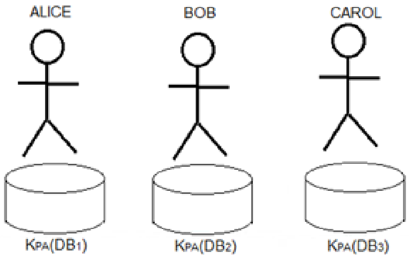

Figure 4: Each party encrypts with its key.

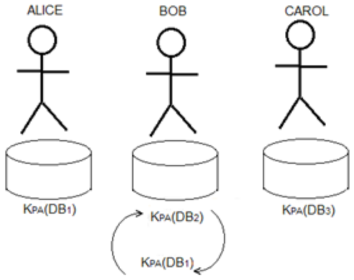

Figure 5: Alice passes data to Bob.

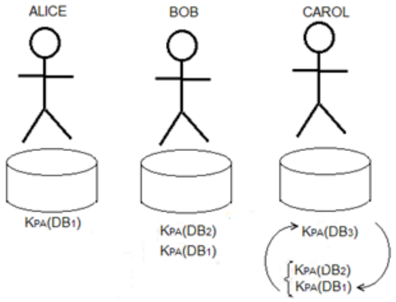

Figure 6: Bob passes data to Carol.

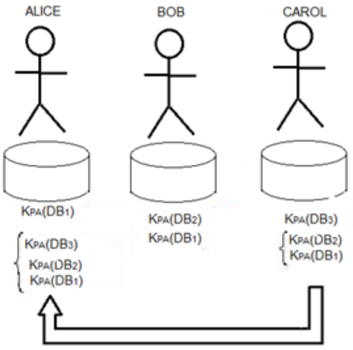

Figure 7: Carol decrypts and publishes to all parties.

3.2 PPARM3 Protocol

In [14] fast privacy preserving association rule mining on distributed homogenous databases (PPARM3), reduces the number of steps from four steps (PPARM4) to only three steps for any numbers of clients to calculate the global candidate itemsets. An example of PPARM3 protocol is shown in Figure 8 where K1 is private key and k2 public key. The details of PPARM3 protocol as in Figure 9 and Table 2. Table 3 shows a comparison of the three algorithms in [14, 15, 26].

3.2 Encryption Algorithms

Many encryption algorithms are widely available and used in information security [11, 12, 22, 23, 25]. They can be categorized into Symmetric (private) and Asymmetric (public) keys encryption. In Symmetric keys encryption or secret key encryption, only one key is used to encrypt and decrypt data. The key should be distributed before transmission between entities. Keys play an important role. If weak key is used in algorithm then everyone may decrypt the data. Strength of Symmetric key encryption depends on the size of key used. For the same algorithm, encryption using longer key is harder to break than the one done using smaller key [13]. There are many examples of strong and weak keys of cryptography algorithms like RC2, DES, 3DES, RC6, Blowfish, and AES. RC2 uses one 64-bit key. DES uses one 64-bits key. Triple DES (3DES) uses three 64- bits keys while AES uses various (128,192,256) bits keys. Blowfish uses various (32-448); default 128bits while RC6 is used various (128,192,256) bits keys [10, 30, 31, 32]. Asymmetric key encryption or public key encryption is used to solve the problem of key distribution. In Asymmetric keys, two keys are used; private and public keys. Public key is used for encryption and private key is used for decryption (E.g. RSA and Digital Signatures). Because users tend to use two keys: public key, which is known to the public and private key which is known only to the user. There is no need for distributing them prior to transmission. Asymmetric encryption techniques are slower than Symmetric techniques, because they require more computational processing power.

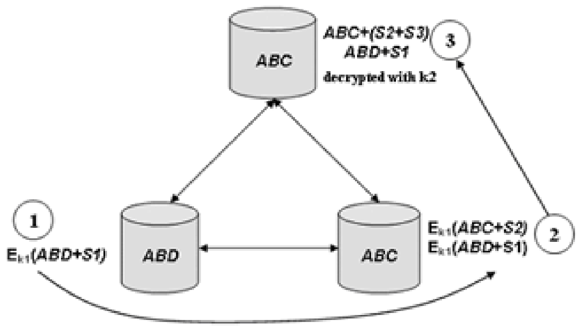

Figure 8: Steps needed for computation of PPARM3 algorithm

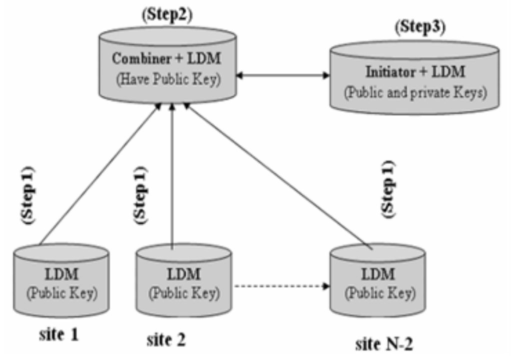

Figure 9: General structure of PPARM3 algorithm

Table 2: Description of PPARM3 protocol

Protocol: Finding large itemsets of size k and global association rules.

Require: N >3 sites one site is algorithm initiator and another is data mining combiner and other called clients (local data mining) sites numbered (1..N − 2) and we assume negligible collision probability.

Step 1: *All local data mining (LDM) compute the mining results using fast distributed mining of association rules (FDM) [26] as locally large k-item sets (LLi(k)) and local support for each item set in LLi(k) then Encrypt frequent item sets and support (LLei(k)) then send it to the data mining combiner.*

Step 2: *The combiner merge all received frequent items and supports with the data mining combiner frequent items and support in encrypted form then send LLe(k) to algorithm initiator to compute the global association rules .*

Step3: *The algorithm initiator receives the frequent items with support encrypted. The initiator first decrypt it, then merges it with his local data mining result to obtain global mining results L(k), then compute global association rules and distribute it to all protocol parties.*

4 Implementations of PPARM4, and PPARM3 with Different Encryption Methods

We implement PPARM4 and PPARM3 algorithms using java. Because the distributed association rules mining need the mining run in more than one site, we can use the RMI (remote method invocation) to connect the sites with each other. Our application is two parts one name server and other is client so we have two site works as server. First is the protocol initiator and second is the data mining combiner and we need client for every participant in the protocol. The initiator is responsible of the threshold of the mining algorithm so it need to define the support and confidence and also generate the public key (k2)and private key (k1) used in encryption and decryption in protocol and finally compute the final results. The data mining combiner responsible of combining the results of clients sites and mix the results to make better privacy of user data and every client is responsible of making the local data mining and encrypt the results of mining and send to data mining combiner. In our implementation our test in data that represent based on 0/1 matrix. And using Public-Key Cryptography, where each user places in a public file an encryption procedure. That is, the public file is a directory giving the encryption procedure of each user. The user keeps secret the details of his corresponding decryption procedure. There are several examples of commutative encryption; perhaps the most famous being RSA (if keys are not shared) and Pohlig-Hellman encryption. Firstly we use RSA that is useful to fulfill our requirements. After that we repeat the same work with different encryption algorithms (DES, 3DES, AES and Blowfish] to evaluate the best encryption algorithm improving the overall performance privacy preserving protocol.

5 Results of Implementations and Discussion

By running PPARM4 and PPARM3 algorithms 75 times. Running testing based on 5 homogenous data bases with different size from 2500 bytes to 2500000 bytes by 15 time for every data base. The 15 values are much closed to each other. The values listed in Table 4 and Table 5 are the average values of time for PPARM4 and PPARM3 respectively in case RSA, DES, 3DES, AES and Blowfish encryption algorithms. Testing is performed using P4 (2.8 GHZ) with Java (SDK 1.6). Figure 10 and Figure 11 shown the relation between different database sizes and time consuming of protocols (PPARM4 and PPARM3) for each encryption algorithm (RSA, DES, 3DES, ASE, and Blowfish(BF)) respectively. The results show PPARM3 is faster than PPARM4 in case all encryption algorithms implemented by the same ratio. Figure 12 and Figure 13 shows a comparison between time of protocol (PPARM4 and PPARM3) respectively and different encryption algorithms (RSA, DES, 3DES, AES and Blowfish(BF)) with variable database size. From these results we can say that the best performance of time for privacy protocols PPARM4 and PPARM3 is in case using encryption algorithm DES. So we can tune the required performance of privacy protocol by control in changing the encryption algorithm.

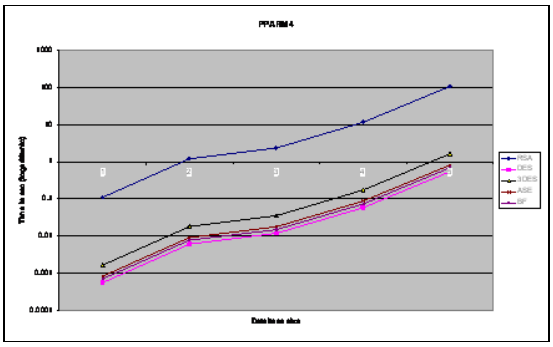

Figure 10: Computation time in ms for PPARM4 with different encryption algorithms

Table 3: Comparison between algorithms in [17, 18, 23].

Comparison factors	Algorithm [23]	Algorithm [17]	Algorithm [18]
Steps for computation	6 steps	4 steps	3 steps
Rounds to compute results	2 rounds	2 rounds	1 round
Cryptography used	Communicative	Public key	Public key
Mining algorithm	Apriori	Apriori	Apriori-Tid

Table 4: Computation time in ms for PPARM4 with different encryption algorithms

Database size (B)	RSA	DES	3DES	AES	Blowfish
2500	0.110583	0.000553	0.001659	0.000829	0.000704
25000	1.211518	0.006058	0.018173	0.009086	0.00771
50000	2.358483	0.011792	0.035377	0.017689	0.015009
250000	11.63843	0.058192	0.174577	0.087288	0.074063
2500000	109.4534	0.547267	1.641801	0.8209	0.696522

Table 5: Computation time in ms for PPARM3 with

Database size	RSA	DES	3DES	AES	Blowfish
2500	0.081684	0.000408	0.001225	0.000613	0.00052
25000	0.404043	0.00202	0.006061	0.00303	0.002571
50000	0.722407	0.003612	0.010836	0.005418	0.004597
250000	3.181355	0.015907	0.04772	0.02386	0.020245
2500000	53.23277	0.266164	0.798492	0.399246	0.338754

different encryption algorithms

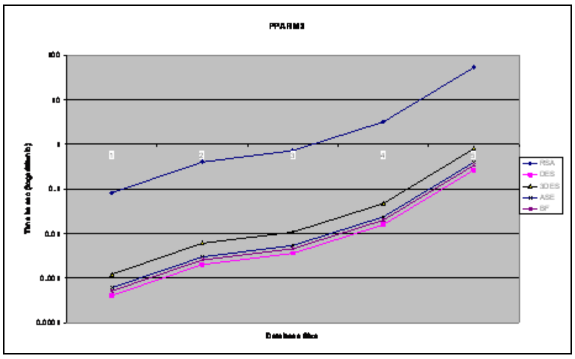

Figure 11: Computation time in ms for PPARM3 with different encryption algorithms

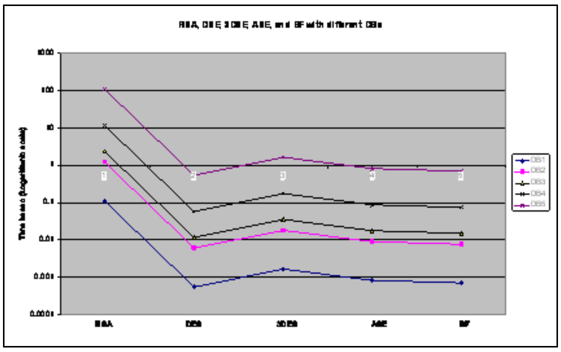

Figure 12: Computation time in ms for PPARM4 with different database sizes

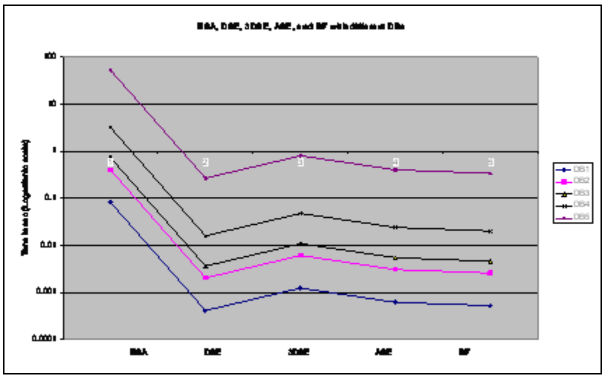

Figure 13: Computation time in ms for PPARM3 with different database sizes

6 Conclusions

In this paper we presented study several encryption algorithms (RSA, DES, 3DES, AES and Blowfish) with different protocols of privacy preserving association rule mining on distributed horizontal database (PPARM4, and PPARM3). Testing is performed using P4 (2.8 GHZ) with Java (SDK 1.6). The results show PPARM3 is faster than PPARM4 in case all encryption algorithms implemented by the same ratio. The results prove that, the best performance of time for privacy protocols PPARM4 and PPARM3 is in case using encryption algorithm DES. So we can tune the required performance of privacy protocol by control in changing the encryption algorithm. In future work we can consider finer tuning by implement many different setting for encryption algorithm using with privacy protocol. For example longer key is harder to break than the one done using smaller key.3DES uses three 64-bits keys while AES uses various (128,192,256) bits keys. Blowfish uses various (32-448); default 128bits.

Acknowledgments

The authors gratefully acknowledge the anonymous reviewers for their valuable comments.

References

[1] R. Agarwal, T. Imielinski, and A. Swami, “Mining association rules between sets of items in large databases,” in *Proceedings of the ACM International Conferences on Management of Data*, pp. 207-216, 1993

[2] R. Agarwal and R Srikant, “Fast algorithm for mining association rules,” in *Proceedings of the 20th International Conference on Very Large Data Bases*, pp. 487-499, 1994.

[3] R. Agrawal and R. Srikant, “Privacy preserving data mining,” in *Proceedings of the ACM SIGMOD Conference*, 2000.

[4] D. Agrawal and C. C. Aggarwal, “On the design and quantification of privacy preserving data mining algorithms,” in *Proceedings of the ACM PODS Conference*, 2001.

[5] D. Agrawal and C. C. Aggarwal, “On the design and quantification of privacy preserving data mining algorithms,” in *Proceedings of the Twentieth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*,, pp. 247–255 Santa Barbara, California, USA: ACM, May 21-23 2001.

[6] M. Z. Ashrafi, D. Taniar, and K. Smith, “PDAM: Privacy-preserving distributed association-rule-mining algorithm,” *International Journal of Intelligent Information Technologies*, vol. 1, pp. 49-69, 2005.

[7] M. Atallah, E. Bertino, A. Elmagarmid, M. Ibrahim, and V. Verykios, “Disclosure limitation of sensitive rules,” in *Proceedings of Knowledge and Data Exchange Workshop*, 1999.

[8] C. Clifton and D. Marks, “Security and privacy implications of data mining,” in *Proceedings of the ACM Workshop Data Mining and Knowledge Discovery*, 1996.

[9] C. Clifton, , “Protecting against data mining through samples,” in *Proceedings of the 13th IFIP WG11.3 Conference on Database Security*, 1999.

[10] D. Coppersmith, “The data encryption standard and its strength against attacks,” *IBM Journal of Research and Development*, pp. 243-250, May 1994.

[11] M. R. Doomun and K. M. S. Soyjaudah, “Analytical comparison of cryptographic techniques for resource-constrained wireless security,” *International Journal of Network Security*, vol. 9, no. 1, pp. 82–94, July 2009.

[12] N. El-Fishawy and O. M. A. Zaid, “Quality of encryption measurement of bitmap images with rc6, mrc6, and rijndael block cipher algorithms,” *International Journal of Network Security*, vol. 5, no. 3, pp. 241–251, nov. 2007.

[13] D. S. A. Elminaam, H. M. A. Kader and M. M. Hadhoud, “Performance Evaluation of Symmetric Encryption Algorithms,” *International Journal of Computer Science and Network Security*, vol. 8, no. 12, pp. 280-286, 2008.

[14] A. El-Sisi, “Efficient privacy preserving association rules mining algorithm on distributed homogenous data base,” *International Arab Journal of Information Technology*, vol. 7, no. 2, 2010.

[15] V. Estivill-Castro and A. Hajyasien “Fast private association rule mining by a protocol securely sharing distributed data,” in *Proceedings of the 2007 IEEE Intelligence and Security Informatics*, pp. 342-330, New Brunswick, New Jersey, USA, May 23-24, 2007.

[16] M. Fatima and N. Safia, “Privacy preserving k-means clustering: a survey research,” *International Arab Journal of Information Technology*, vol. 9, no. 2, 2012.

[17] U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, *Advances in Knowledge Discovery and Data Mining*, AAAI Press/The MIT Press. 1996.

[18] O. Goldreich, “Secure multi-party computation,” (working draft). http://www.wisdom.weizmann.ac.il/oded/pp.html

[19] J. W. Han, J. Pei, and Y. W. Yin, “Mining frequent patterns without candidate generation,” *in Proceedings of the ACM International Conference on Management of Data*, pp. 1-12, 2000.

[20] J. W. Han and M. Kamber, *Data Mining: Concepts and Techniques*, Morgan Kaufmann Publishers. 2001

[21] C. Hidber, “Online association rules mining,” in *Proceedings of the ACM SIGMOD Conference Management of Data*, 1999.

[22] M. S. Hwang and C. Y. Liu, “Authenticated encryption schemes: current status and key issues,” *International Journal of Network Security*, vol. 1, no. 2, pp. 61–73, Sep. 2005.

[23] M. H. Ibrahim, “A method for obtaining deniable public-key encryption,” *International Journal of Network Security*, vol. 8, no. 1, pp. 1–9, Jan. 2009.

[24] I. Ioannidis and A. Grama, “An efficient protocol for yao’s millionaires’ problem,” in *Hawaii International Conference on System Sciences (HICSS-36),* Waikoloa Village, Hawaii, Jan. 6-9, 2003.

[25] I. A. Ismail, M. Amin, and H. Diab, “A digital image encryption algorithm based a composition of two chaotic logistic maps,” *International Journal of Network Security*, vol. 11, no. 1, pp. 1-10, 2010.

[26] M. Kantarcioglu and C. Clifton. “Privacy-preserving distributed mining of association rules on horizontally partitioned data,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 9, pp. 1026–1037, Sep. 2004.

[27] G. Lee, C. Chang, and A. L. P. Chen, “Hiding sensitive patterns in association rules mining,” in *Proceedings of the 28th Annual International Conference on Computer software and Applications Conference*, pp. 424-429, 2004.

[28] D. Lin and Z. M. Kedam, “Pincer-search: An efficient algorithm for discovering the maximum frequent set,” *IEEE Transactions on Knowledge and Data Engineering*, 14. 2002.

[29] Y. Lindell and B. Pinkas, “Privacy preserving data mining,” *Journal of Cryptography*, pp.177-206, 2002.

[30] R. L. Rivest, A. Shamir, and L. Adleman, “A method for obtaining digital signatures and public-key cryptosystems,” *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, 1978.

[31] B. Schneier, *The Blowfish Encryption Algorithm*, Retrieved Oct. 25, 2008.

[32] W. Stallings, *Cryptography and Network Security*, 4th Ed. Prentice Hall , PP. 58-309 . 2005.

http://www.schneier.com/blowfish.html

[33] G. I. Webb, “Efficient search for association rules,” in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 99-107, 2000.

.

Ashraf B. El-Sisi received the B.S. and M.S. in Electronic Engineering and Computer Science Engineering from Menofia University, Faculty of Electronic in 1989 and 1995, respectively and received his PhD in Computer Engineering & Control from Zagazig University, Faculty of Engineering in 2001. His research interest includes privacy preserving data mining, AI approaches in software testing, Intelligent Agent, Implementation of security and graphics algorithms based on FPGA, Testing biometric security algorithms and devices.

Hamdi M. Mousa received the B.S. and M.S. in Electronic Engineering and Automatic control and measurements from Menofia University, Faculty of Electronic Engineering in 1991 and 2002, respectively and received his PhD in Automatic control and measurements Engineering (Artificial intelligent) from Menofia University, Faculty of Electronic in 2007. His research interest includes intelligent systems, Natural Language Processing, privacy, embedded systems, and GSP applications.