Fingerprinting Non-Numeric Datasets Using Row Association and Pattern Generation

Mashal Ahmad[1], Arsalan Shahid[2], Muhammad Yasir Qadri[3], Khalid Hussain[4], Nadia N. Qadri[1]
[1]COMSATS Institute of Information and Technology (CIIT), Wah Cantt, Pakistan
[2]School of Computer Science, University College Dublin, Belfield, Dublin 4, Ireland
[3]Centres of Excellence in Science & Applied Technologies (CESAT), Pakistan
[4]Muslim Youth University, G-10/4, Islamabad, Pakistan

{mashal.ahmad,drnadia}@ciitwah.edu.pk, arsalan.shahid@ucdconnect.ie, yasirqadri@acm.org

***Abstract*— Being an era of fast internet-based application environment, large volumes of relational data are being outsourced for business purposes. Therefore, ownership and digital rights protection has become one of the greatest challenges and among the most critical issues. This paper presents a novel fingerprinting technique to protect ownership rights of non-numeric digital data on basis of pattern generation and row association schemes. Firstly, fingerprint sequence is formulated by using secret key and buyer's Unique ID. With the chunks of these sequences and by applying the Fibonacci series, we select some rows. The selected rows are candidates of fingerprinting. The primary key of selected row is protected using RSA encryption; after which a pattern is designed by randomly choosing the values of different attributes of datasets. The encryption of primary key leads to develop an association between original and fake pattern; creating an ease in fingerprint detection. Fingerprint detection algorithm first finds the fake rows and then extracts the fingerprint sequence from the fake attributes, hence identifying the traitor. Some most important features of the proposed approach is to overcome major weaknesses such as error tolerance, integrity and accuracy in previously proposed fingerprinting techniques. The results show that technique is efficient and robust against several malicious attacks.**

***Keywords*—**Fingerprinting; Ownership protection; Non-numeric datasets; Pattern generation and row association schemes

I. INTRODUCTION

Digital assets usage as a valuable information has increased tremendously over some past decades [1]. These assets are publically available to anyone around the globe [2]. Due to convenience in copying and sharing of electronic assets across the internet, the merchants of such useful information are mainly worried about the copyright protection [3]. This has urged researchers and engineers to contribute in this field; and thus, many algorithms have been proposed, employed and some of them have been commercialized in market. Nevertheless, in these schemes, apart from level of security, computational complexity of the algorithm has been addressed over the decades. In this perspective, an algorithm with high level of security with implementation simplicity has always been a demand of industry [4]. Some techniques have been proposed in order to hide useful information, i.e., fingerprinting and watermarking [5]. Fingerprinting is a method of hiding the digital marks in the data with intention to detect the recipients [6], whereas watermarking aims to identify the sources of data [7]. Both of these approaches protect the digital data from being copied illegitimately which is a serious threat to database applications.

In this paper, we have devised a robust technique for fingerprinting non-numeric datasets using row association and pattern recognition. The main idea is to use fingerprint sequence for the selection of rows and then generate their fake rows that will act as fingerprints. Our fingerprinting model uses RSA encryption technique [8] which provides association between fingerprinted rows and original rows. The detection is based on the identification of fingerprinted rows. It is to be noted that in relational database watermarking and finger printing, much of the work is done on numeric attributes. The techniques applied on numerical data are not applicable on non-numerical data as non-numerical attributes have well-defined semantics and cannot tolerate small modifications like bits flipping. The aim of this research is to provide a technique which result in significant reduction of the issues related to sanctuary and fortification of digital assets from being used illegitimately.

The rest of this paper has been divided in to six sections. In the following section related work is discussed which includes the review of relational database watermarking and fingerprinting techniques and identify the deficiencies in them. Section 3 presents the approach overview. Fingerprinting mechanism has been given in section 4. Results have been shown in section 5 and the final section presents the conclusion.

II. RELATED WORK

This section presents the related research findings in the area of digital assets protection from being used illegitimately through standard state-of-the-art techniques i.e. fingerprinting and watermarking.

The watermarking schemes are categorized into two types, i.e., resilient and benign schemes. The resilient patterns [9]–[11] mainly concern copyright protection, whereas the benign methods [12]–[14] are aimed for tamper detection and integrity proof of database relations. Agrawal et al. [11] presented a watermarking routine of hiding the watermark bits in least significant bits (LSB) of selected entities of particular subsets of tuples by combining the secret parameter and the tuple's primary key. This technique does not provide a mechanism for multi-bit watermarks; instead only a secret key is used. The LSB-based procedure employed is not robust to the alteration attacks since fluctuating the one position of LSB results in forfeiting

watermark without losing the credibility of data. Moreover, unrestrained manipulation of LSB in any row can cause unwanted results. Zhang et al. [15] proposed a watermarking routine using image based watermarking (IBW) in relational databases. In order to represent the copyright information an identification image is inserted into the relational data. Ruanaidh et al. in [16] proposed that efficient way of detecting the watermark is that it must be vivid, secure and resilient to attacks. However, the method of constructing watermarking is very complicated in the above mentioned researches and detection of watermarks is not very effective.

Guo et al. [17] demonstrated a double folded insertion method for fingerprinting of relational databases. It identifies the malicious attacks to a relation of database. The watermark is created on the basis of parameters of database relation. Firstly, the merchant's secret key is used to control the insertion of fingerprint bits in targeted group of tuples and further this step detects the recipient of database. In the second step, the fingerprint as the secret key embeds the pattern for validating the extracted fingerprint providing the numerical assurance. But this method results in affecting integrity and usability of data and hence become impractical for safety critical databases.

Kamel in [18] presented a scheme to protect the integrity of database relations. Their scheme divides the database relations in groups and each group is marked independently. Li et al. in [19] formulated a fingerprinting method that is independent of actual primary key and generates virtual primary key using most significant bits of different attributes of each row. Selection of the attributes is based on a hidden key that is only known to the owner of the database. But watermarking approach uses the idea of reallocations of rows and is susceptible to sorting attacks. Liu et al. [20] presented block method that validates specific bit locations of data containing the specific values. The bit locations are determined by the keys known only to the owner of the data. The results verify that this scheme is resilient to various types of attacks.

In relational database watermarking and fingerprinting the work done by researchers is mostly on numeric attributes. Moreover, it is notable that the techniques applied on numeric data are not applicable on non-numeric data as non-numeric attributes cannot even tolerate small modifications.

III. APPROACH OVERVIEW

In order to understand this problem more precisely, consider a scenario in which merchants or owners (Alice) sell digital data to buyers or purchasers (Carol, Mallory and Eric). Some of the treacherous purchasers, i.e., Mallory may redistribute the data to other people without asking the merchants. These disloyal purchasers are also known as traitors. In such a context, enforcing data ownership is extremely important. Therefore, to prevent such piracy of data the owner must device some schemes. One such scheme is fingerprinting. Fingerprinting can address the above mentioned problems in an effective manner and also helps trace piracy in outsourced data. Figure 1 highlights the complete scenario in detail.

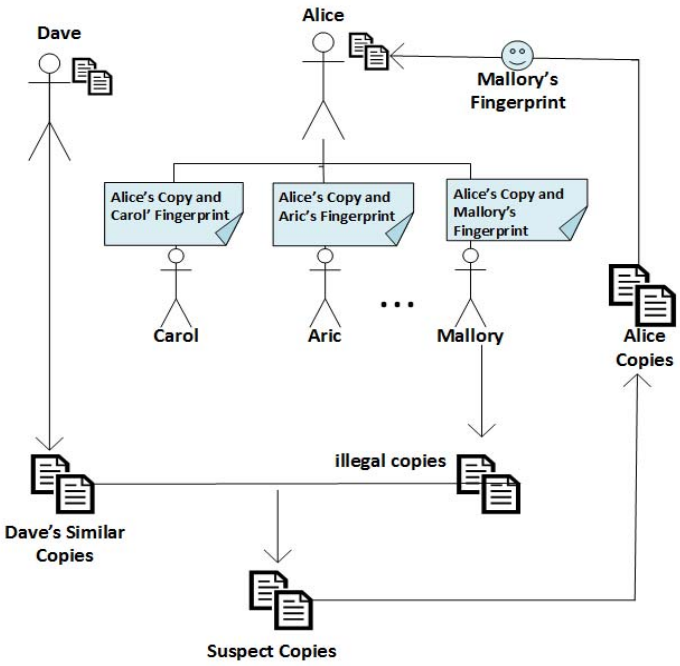

Fig. 1. Illustration of Scenario

A. *Poposed Methdology*

Our proposed fingerprinting model is based on the fake row generation and row association between original and fake row. The tuple section is based on finger prints. For each row a fake row is created that act as a finger print. Table 1 shows the list of parameters (notations) which are used in our model.

Table 1: List of notations

Symbols	Explanation
D_o	Original Dataset
D_f	Fingerprinted database
D_b	Buyer's id database
η	Number of rows
ω	Actual number of rows marked
Ks	Secret parameter
$Ksfp$	Selected Chunks for fingerprints
R_{sel}	Selected Row
R_f	Generated fake row
Fps'	Regenerated Fingerprints sequence
Fps	Fingerprints sequence
Pk_{sel}	Primary key of selected row
P_{kf}	Fake primary key of selected row
$\hat{G}$	Secret Parameter for Pattern Generation
$G1$	Pseudorandom Generated Sequence for Row Selection
$G2$	Pseudorandom Generated Sequence for chunk formation
T	Threshold
N	Number of buyers

B. *Design Model*

Figure 2 shows the block diagram of fingerprinting insertion model and fingerprinting extraction model is displayed which elaborate the main modules of the fingerprinting system.

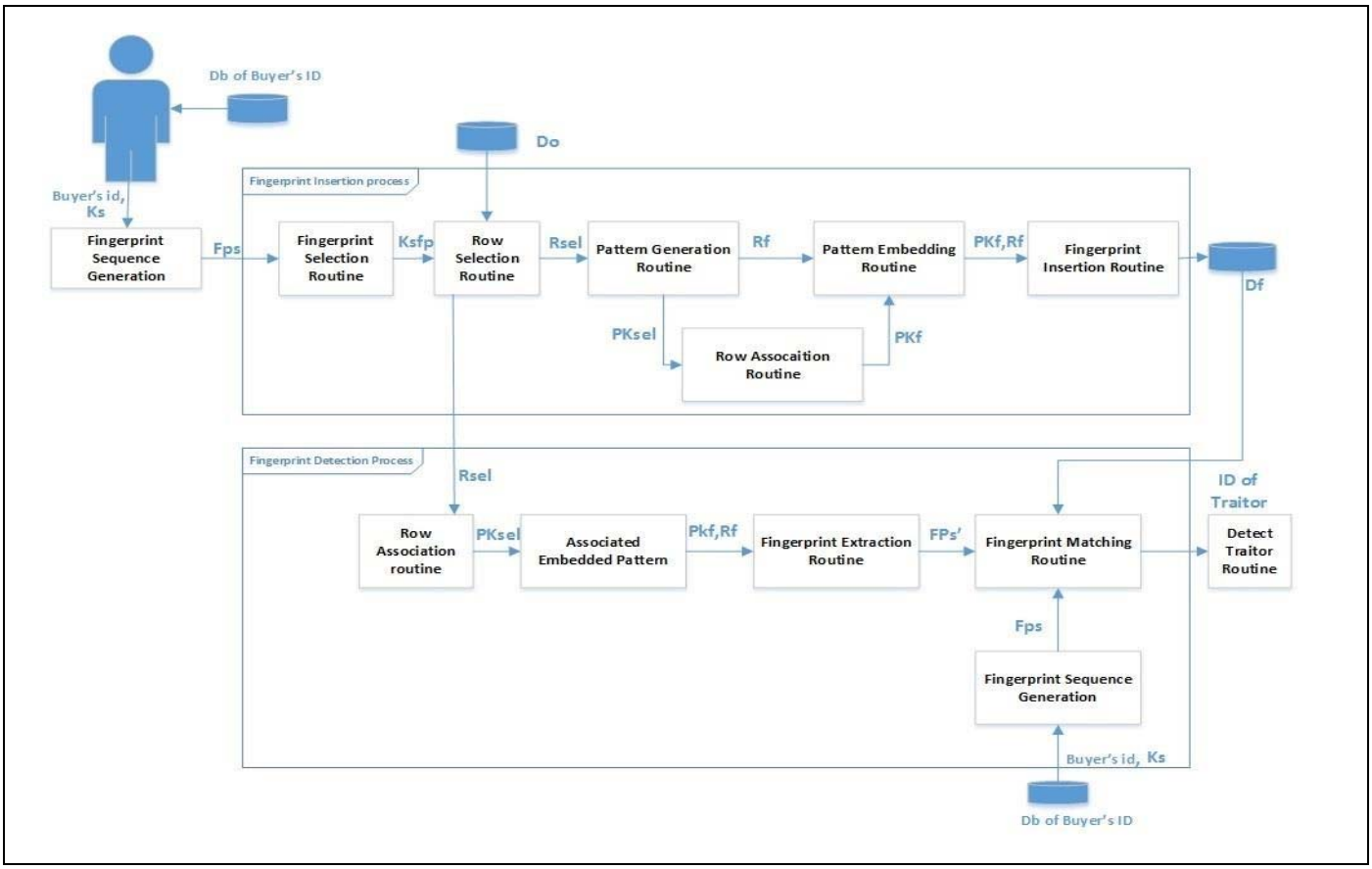

Fig. 2. Fingerprint insertion and detection model

A data set (Do) is transformed into a fingerprinting version (Df) by applying a fingerprinting encoding function. Encoding function takes secret key (Ks) and the buyer id as inputs. Ks is only known to the owner. The fingerprint insertion routines can be summarized in the following steps:

Step 1(I): Fingerprint Sequence Generation Routine: By using secret key (Ks) and ID associated with particular user fingerprint sequence is formulated by concatenating Ks and ID. Both Ks and ID can be of any length as it is actually seed to pseudo random generator.

Step 2(I): Fingerprint Selection Routine: This sequence of fingerprint is seeded to random number generator that gives the different chunks of fingerprints (Kfp), i.e., either the sequence is in units, tens or hundreds forms.

Step 3(I): Row selection Routine: Chunks formed in step 2(I) are than checked if they are in range of total number of rows of datasets. On basis of these chunks rows are selected which are candidates of fingerprinting.

Step 4(I): Pattern Generation Routine: The primary key of candidate row is encrypted using RSA algorithm and fake rows are formulated against encrypted primary key by using a secret parameter ω, i.e., the percentage of total rows.

Step 5(I): Row Association Routine: An association is developed between the fake (Rf) and the selected row, ($Rsel$) as fingerprint.

Step 6(I): Pattern Embedding Routine: Finally the fake row (Rf) and selected row ($Rsel$) embedded as a fingerprint.

Step 7(I): Fingerprint Insertion Routine: Fingerprints are inserted in the fake rows obtained in step 3(I).

The fingerprinted version (Df) is distributed to the intended beneficiary. Then it can suffer from unintentional distortions or attacks aimed at destroying the fingerprints. Note that even intentional errors are introduced without any knowledge of *K*s or *D*o, since these are not publicly available. Fingerprint decoding is the process of extracting the embedded fingerprints using the fingerprinted data set (Df), the secret key and the mapping function. The decoding algorithm is blind as the original data set D is not required for the successful decoding of the embedded fingerprint. Fingerprint decoding is explained in the following steps:

Step 1(D): Fingerprint Row Selection Routine: Using the fingerprint row selection routine used in
Step 2(I), the marked rows are selected.

Step 2(D): Row Association Routine: The association between the pattern generated row and its fingerprinted row is evaluated using the association routine in Step 5(I) original row and secret key.

Step 3(D): Fingerprint Extraction Routine: Once the existence of fake row (Rf) is confirmed, regenerated fake pattern is matched with that of the identified fake row, (Rf) to verify the detection of the fingerprint.

Step 4(D): Fingerprint Matching Routine: The fingerprints extracted in step 3D are matched with the fingerprints saved in customer information database.

IV. THE FINGERPRINTING INSERTION MECHANSIM

This section presents the mechanism of our proposed fingerprinting technique.

A. Fingerprint Sequence Generation Routine

Fingerprint creation algorithm that creates the fingerprint to be inserted into the database and which is unique for each buyer is shown in Algorithm 1. This routine takes the secret key (Ks) and the buyer's identification number id as inputs. The secret key and unique ID of buyer is concatenated and converted into same format. These sequences are solely unique for each buyer and generates unique fingerprints for each buyer and kept save in database by owner for keeping record of recipient.

Algorithm 1: Fingerprint sequence generation

Input: Buyer id
Secret Key: K_s
Output: Chunk of finger print (K_{sfp})
For each buyer $n \in N$ do
1. Input the buyer id and secret key
2. Fingerprint_buyer ← convert to same format (secret key || id_n)
3. FPs(selected) ← chunk (fingerprint _buyer)
4. Save id in DB

B. Fingerprint Row Selection

This section describes the selection algorithm that selects the tuple for marking from data set based on secret key Ks (See Algorithm 2). Pseudorandom sequence generates random numbers that is passed through the procedure of chunk formation and conversion of chunks into Fibonacci sequence. In this way the Row Selection () function works. At Line 1-2 Ǫ1 and Ǫ2 are seeded with Ks and $Ksfp$, therefore, it is very difficult for the attacker to guess and generate exact sequence of numbers. The sequence that is generated through Ǫ1 is passed through the procedure of chunk formation to extract row numbers and the sequence generated through Ǫ2 decides the choice of chunk formation method. The Ǫ2. Random Next (1, 4) function randomly generates numbers between 1 and 4 these numbers indicate the choices of chunk formation method. The chunks formed are converted to Fibonacci sequence at line 18-19.

Algorithm 2: Row selection

Input: $Ksfp$,Ks, ω
Output: Rows Selected
1. Seed Ǫ1with Ks, $Ksfp$
2. Seed Ǫ2with Ks, $Ksfp$
3. While (true)
4. Seq ← Ǫ1.Random Next ( )
5. Choice ←.Random Next(1,4)
6. Select Case Start
7. Case1: Units(Seq)
8. Case 2: Tens(Seq)
9. Case 3: Hundreds(Seq)
10. Case 4: Thousands(Seq)
11. End Select
12. If Number in Seq <= η Then
13. Number Array=Sequence generated
14. Remove Duplication
15. End If
16. If Sizeof (NumberArray) > ω then
17. Exit
18. NumberArray=Fibonacci(NumberArray)
19. Sort (NumberArray)
20. Return NumberArray as Rows Selected

C. Pattern Generation Routine

For each selected row $Rsel$, fake row Rf is generated using pattern generation algorithm (See Algorithm 3). Pattern of fake row is design by scanning all the attributes of table and randomly choosing values of different attributes of table except the primary key attribute. In Pattern Generation each selected row's Primary key is selected in line 2. In line 3 encryption of primary key of selected row $PKsel$ using RSA [8] algorithm is done which produces fake primary key PKf and develops an association between selected row and fake row. From line 4-5 the fake primary key is stored in an array and any duplication if found is removed. From Line 6-10 pattern is generated for all attributes of selected rows except primary key. Random number generator produces a sequence of numbers. These numbers corresponds to the values from the selected rows which are selected by the random number generator to be filled in as fake rows.

Algorithm 3: Pattern Generation

Input: RowSelected, Dcol
Output: Rf
1. foreach $Rsel \in$ RowSelected
2. fetch $PKsel$ from $Rsel$
3. PKf = RSA algorithm ($PKsel$)
4. Store Pkf in fakearray[i]
5. Remove Duplication
6. for i < Dcol //for each attribute in row
7. fakearray[i] ← D'col
8. RandomGenerator(fakearray[i])
9. Store as Rf
10. Next i
11. End

D. Row Association Routine

By applying RSA encryption on primary key of selected row (PK_{sel}), a fake primary key of selected row (PK_f) is attained and finally the fake row (R_f) is inserted with PK_f into the dataset as fingerprint. Row association routine is shown in Algorithm 4.

Algorithm 4: Row Association

Input: $PKsel$
Output: PKf
1. PKf ← Encrypt($PKsel$)
2. Return PKf

E. Fingerprints Insertion with fake pattern

Algorithm 5 shows embedding of the fingerprints with fake patterns. The original dataset, Do undergoes the procedure of row selection routine based on a fingerprints (Rf). In line 1, the RowSelection() subroutine returns the selected rows. The detail of RowSelection() subroutine is presented in Figure 4. In line 2-5, it is shown that for each selected row ($Rsel$), a fake row (Rf) is generated by PatternGen() subroutine. It is to be noted that the design of pattern depends on the contents of data set. At Line 4-5 the dataset $D0$ is updated and the next row is selected

for fingerprinting. At Line 7 when all the candidate rows are fingerprinted the original dataset Do is returned as fingerprinted Df.

Algorithm 5: pattern embedding and Fingerprint insertion.

Input: $\boldsymbol{D_o}$,PK_f, R_f
Output: Fingerprinted data set D_f
1. Rows selected = RowSelection()
2. For each R_{sel} ϵ Row Selected
3. $\boldsymbol{R_f}$ ← PatternGen ( )
4. $\boldsymbol{D_o}$ ← Embedding($\boldsymbol{PK_{sel}}$,$\boldsymbol{R_f}$)
5. Update D_0
6. Next $\boldsymbol{R_{sel}}$
7. Return $\boldsymbol{D_o}$ as $\boldsymbol{D_f}$

V. THE FINGERPRINTING DETECTION MECHANSIM

When the owner gets a pirated copy of his database he has to determine the traitor. He will first run the detection algorithm with the same secret key and other parameters and then the detect traitor subroutine to find the traitor.

A. Fingerprint Detection

For purpose of traitor detection first we regenerate finger print with each user id to whom we sell our database then after concatenating the user id with secret key (Ks) row is selected. Once we determined selected row ($Rsel$) its associated fake row (Rf) is identified with mapping function. Fake pattern is regenerated (Fps') and is matched with row identified if matching result is true then finger print is extracted else if matches are few then traitor is detected and particular user will be identified as a traitor. Algorithm 6 describe fingerprinting detection.

Algorithm 6: Fingerprint Detection

Input: Df, α, Rf, Ks
Output: Detected fingerprint sequence Fps
1. Rows Selected ← Row Selection (Ks,$Ksfp$)
2. For each $Rsel$ ∈ Rows Selected do
3. Fps' ← Pattern Gen()
4. Total_count ←Total_coutn +1
5. PKf ← Mapping ($PKsel$)
6. Flag ← Matching (PKf,Rf)
7. If Flag ←true Then
8. Match_count ← Match_count +1
9. End If
10. Next $Rsel$
11. T ← threshold (Total_count, α)
12. If (Match_count < T) Or (Match_count > Total_count) Then
13. FPs detected
14. Traitor detection( )
15. End If

B. Fingerprint Extraction and Matching Routine

The regenerated fake pattern (F_{ps}') is then matched with that of the identified fake row (R_f). The "Match_Count" variable is incremented on true. Algorithm 7 shows matching and extraction routine.

Algorithm 7: Fingerprint extraction

Input: PKf, Rf
Output: Boolean value
1. Boolean Flag ← False
2. Move to first of Df Records
3. Repeat Until end of Df Records
4. If PKf' ← Pkf Then
5. If Rf at Pkf ← Fps' Then
6. Flag ← true
7. Extraction and record (Rf)
8. End If
9. End If
10. Move to Next of Df Records
11. End Repeat
12. If Flag ← true Then
13. Return true
14. Else
15. Return false
16. End If

C. Fingerprint Traitor Detection Routine

After fingerprint is detected, traitor subroutine is used to detect the traitor. Traitor is identified if the extracted fingerprint pattern F_{ps}' matches one of the buyer's fingerprints (N), which is computed in the same manner as that of fingerprint generation sequence routine. Algorithm 8 highlights detection of traitor.

Algorithm 8: Traitor detection

Input: extracted fingerprint Fps', buyer id, Ks
Output: traitor id
1. buyer ← traitor detection(Fps',ks)
2. if buyer != null then buyer is traitor else none suspected
3. detect_traitor (fingerprint Fps', secret key Ks) return buyer id
4. for each buyer n in Db do
5. select buyer id say idn
6. fingerprint_buyer ← convert to same format (Ks || idn)
7. fingerprint Fps ← chunk(fingerprint _buyer)
8. if fingerprint Fps' ← Fps return buyer id of traitor

VI. RESULTS

In this section results of an experimental study are reported that analyze the resilience of proposed fingerprinting scheme to subset attacks. Three types of subset attacks have been considered, i.e., subset selection attack, subset addition attack and mix-match attack.

A. Subset Selection Attack

This type of attack is encountered when a malicious user takes a portion or subset of the dataset containing the fingerprints and uses this subset to make another smaller relation by destroying the fingerprint. The malicious attacker can now perform any kind of operation on the subset like addition, deletion and modification. This attack can only be successful when the malicious user is able to find out the association between the patterns generated fake fingerprinted rows Rf and their associated selected rows $Rsel$. There are different cases for this type of attack one of them is that suppose attacker selects a subset containing the rows selected $RselA$ and its associated fake pattern generated fingerprinted rows RfA, initially marked by its real owner. The attack can only be successful if attacker selects small sample size like 40% or 50% where only 15% or 16% fingerprints can be extracted. Figure 3 given below shows the results of subset selection attack. It can be shown that as the sample size gets larger and larger the more percentage of fingerprints can be extracted and when 100% rows are selected

by attacker for attack, owner can easily extract her whole fingerprint that is the attack becomes unsuccessful.

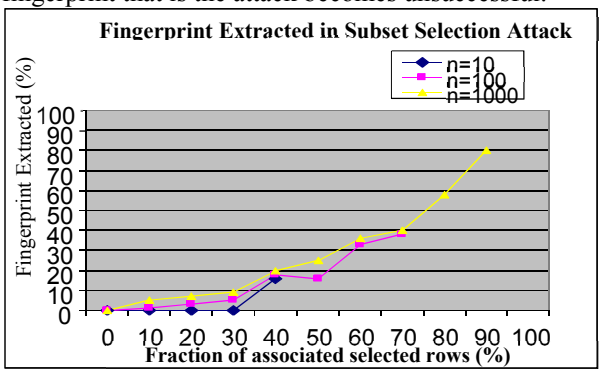

Fig. 3. Subset selection attack

B. Subset Addition Attack

In this type of attack the attacker tries to insert his own fingerprint in already fingerprinted relation and claims false ownership. In this type of attack the proof of ownership lies in the fact that Alice's fingerprints must exist after Mallory marks or fingerprints the relation. Figure 4 shows the results of subset addition attack. It is shown that probability of selected rows decreases as a result the efforts of attacker regarding ownership claim becomes useless.

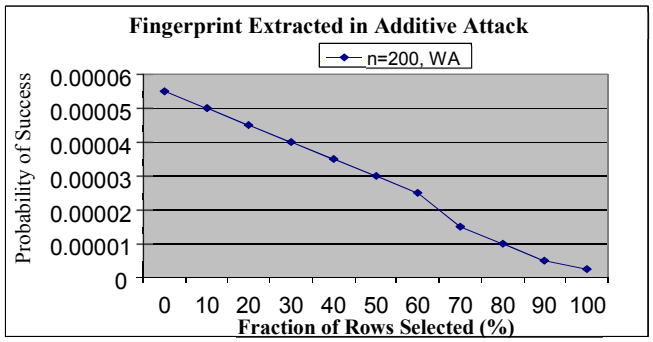

Fig. 4. Subset addition attack

C. Mix-and-Match Attack

This type of attack involves the attacker to take different portions of data from different sources containing similar relations one of which can be owner's fingerprinted dataset D_f and combines them to make a relation of his own. The results of mix-and-match attack are shown in figure 5 given below. It is shown that as attacker includes more rows from owner's relation, the chance of successful detection of fake rows and their embedded fingerprints increases.

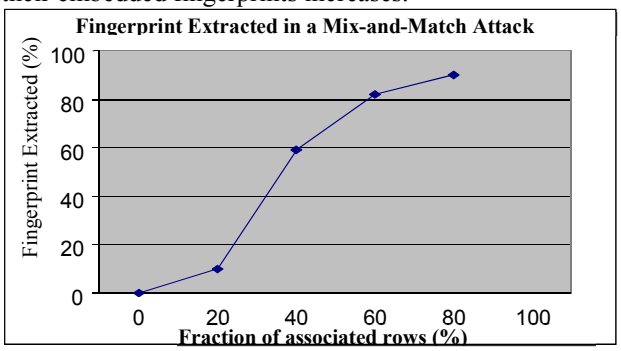

Fig. 5. Mix-and-match attack

We have compared our technique with different fingerprinting methodologies and is presented as follows in table 2.

Table 2. Comparison of proposed technique with existing techniques

Features	Our proposed technique	Technique [11]	Technique[17]	Technique [19]	Technique [20]
Accuracy	As no operations are performed on the original rows so high accuracy is ensured	Provides medium accuracy	Medium accuracy	Small errors were found while fingerprinting data.	High accuracy as it marked the multimedia data.
Error Tolerance	Pattern generation technique is resilient to error tolerance because it uses fake data that does not require checking of attributes bearing sensitivity. So provides 100% error tolerance	Provides medium error tolerance	Medium error tolerance	To apply fingerprint on numeric data only those attributes are selected that are error tolerant	Check areas of image where mark can embed, so it needs error tolerant areas
Usability Constraint Checking	Checking usability is not required because every change is made in the pattern generated row, so in case if undesirable results are produced, it will not affect the usability of original data.	It requires checking usability of the data as errors are introduced in the original data so in order to retain the quality of the data this constraint is checked.	Usability constraint of marked rows is accumulated	It requires checking usability of the data as errors are introduced in the original data	Usability of data is compromised to some extent.
Integrity	Integrity is 100% maintained because there no distortion at all in the original data due to fake pattern generated fingerprinted rows.	Integrity is not guaranteed because there is distortion in original data due to errors introduced during watermarking	Integrity is compromised	Integrity is not ensured due to errors introduced in the original dataset	Integrity is maintained.

VII. CONCLUSIONS

This research work presented a flexible fingerprinting technique for relational data that embeds the fingerprint in the fake attributes of pattern generated rows. The problem of error tolerance and accuracy related to the relational datasets is formulated as pattern generation technique maximizes the accuracy of results on query processing from marked relations. Association between selected rows and their associated fingerprinted fake generated rows has been provided by public key encryption algorithm RSA. The analysis and results described through graphs showed the robustness of the proposed technique against various malicious attacks. The technique is evaluated with the existing techniques and is proved to be robust.

REFERENCES

[1] E. Yakel, "Digital assets for the next millennium," OCLC Syst. Serv. Int. Digit. Libr. Perspect., vol. 20, no. 3, pp. 102–105, 2004.

[2] F. A. Petitcolas, R. J. Anderson, and M. G. Kuhn, "Information hiding-a survey," Proc. IEEE, vol. 87, no. 7, pp. 1062–1078, 1999.

[3] B. N. Chaudhary and A. K. Sharma, "A Modern Watermarking Approach for Non- Numeric Relational Database," International Journal of Advanced Research in Computer Science and Software Engineering, vol. 3, no. 12, pp. 1102–1108, Dec. 2013.

[4] D. Maltoni, D. Maio, A. K. Jain, and S. Prabhakar, Handbook of fingerprint recognition. Springer Science & Business Media, 2009.

[5] A. M. Alattar, K. L. Levy, R. R. Stager, G. B. Rhoads, and E. E. Ellingson, Digital watermarking and fingerprinting including synchronization, layering, version control, and compressed embedding. Google Patents, 2006.

[6] P. C. Mandal, "Modern Steganographic technique: A survey," Int. J. Comput. Sci. Eng. Technol. IJCSET, vol. 3, no. 9, pp. 444–448, 2012.

[7] P. Singh and R. S. Chadha, "A survey of digital watermarking techniques, applications and attacks," Int. J. Eng. Innov. Technol. IJEIT, vol. 2, no. 9, pp. 165–175, 2013.

[8] M. Agrawal and P. Mishra, "A comparative survey on symmetric key encryption techniques," Int. J. Comput. Sci. Eng. IJCSE, vol. 4, no. 05, pp. 877–882, 2012.

[9] G. Gupta and J. Pieprzyk, "Database relation watermarking resilient against secondary watermarking attacks," in Information Systems Security, Springer, 2009, pp. 222–236.

[10] M. Huang, J. Cao, Z. Peng, and Y. Fang, "A new watermark mechanism for relational data," in Computer and Information Technology, International Conference on, 2004, pp. 946–950.

[11] R. Agrawal, P. J. Haas, and J. Kiernan, "Watermarking relational data: framework, algorithms and analysis," VLDB J., vol. 12, no. 2, pp. 157–169, 2003.

[12] H. Rashidi and others, "A novel watermarking scheme for detecting and recovering distortions in database tables," ArXiv Prepr. ArXiv10090827, 2010.

[13] S. Iqbal, A. Rauf, S. Mahfooz, S. Khusro, and S. H. Shah, "Self-constructing fragile watermark algorithm for. relational database integrity proof," World Appl. Sci. J., vol. 19, no. 9, pp. 1273–1277, 2012.

[14] S. Bhattacharya and A. Cortesi, "A Distortion Free Watermark Framework for Relational Databases.," in ICSOFT (2), 2009, pp. 229–234.

[15] Z.-H. Zhang, X.-M. Jin, J.-M. Wang, and D.-Y. Li, "Watermarking relational database using image," in Machine Learning and Cybernetics, 2004. Proceedings of 2004 International Conference on, 2004, vol. 3, pp. 1739–1744.

[16] J. Ó. Ruanaidh, W. J. Dowling, and F. M. Boland, "Watermarking digital images for copyright protection," IEE Proc.-Vis. Image Signal Process., vol. 143, no. 4, pp. 250–256, 1996.

[17] F. Guo, J. Wang, and D. Li, "Fingerprinting relational databases," in Proceedings of the 2006 ACM symposium on Applied computing, 2006, pp. 487–492.

[18] I. Kamel, "A schema for protecting the integrity of databases," Comput. Secur., vol. 28, no. 7, pp. 698–709, 2009.

[19] Y. Li, V. Swarup, and S. Jajodia, "Constructing a virtual primary key for fingerprinting relational data," in Proceedings of the 3rd ACM workshop on Digital rights management, 2003, pp. 133–141.

[20] S. Liu, S. Wang, R. H. Deng, and W. Shao, "A block oriented fingerprinting scheme in relational database," in Information Security and Cryptology–ICISC 2004, Springer, 2005, pp. 455–466.

[21] D. Bleichenbacher, "Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS# 1," in Advances in Cryptology—CRYPTO'98, 1998, pp. 1–12.

[22] E. F. Brickell, "A survey of hardware implementations of RSA," in Advances in Cryptology—CRYPTO'89 Proceedings, 1990, pp. 368–370.