Experimental Study of Improved Watermarking Scheme on Relational Databases

N.G. Pardeshi
Assistant Professor
University of Pune
Department of Computer Engineering
SRESCOE Kopargaon-423603

J.N. Kale
Assistant Professor
University of Pune
Department of Computer Engineering
SRESCOE Kopargaon-423603

M.S. Ankoshe
Assistant Professor
University of Pune
Department of Computer Engineering
SRESCOE Kopargaon-423603

ABSTRACT
The main objective of watermarking is to protect a digital content from unauthorized redistribution and copying by enabling ownership provability over the content. The goal of digital watermarking is to insert a robust and imperceptible watermark into the digital content such that the mark does not destroy the value of the content, and the mark is hard to be removed by attackers without destroying the utility of content. The measurement of the value of the content is closely related to the data type and its intended use.

We have proposed a new marking scheme called as improved watermarking. Watermarking scheme first generates the bit string of fixed length. The mark bit is computed with the help of watermark bit and mask bit, which is used to mark the certain bit location of the attribute value.

General Terms
Relational Database Security, Improved Watermarking on Relational Database

Keywords
Digital watermarking, relational database systems, copyright protection, ownership verification, imperceptibility, robustness, Watermark embedding, Watermark extraction.

1. INTRODUCTION
The watermarking software introduces small errors into the object being watermarked. These intentional errors are called *marks* and all the marks together compose the *watermark*. The marks must not have a significant impact on the usefulness of the data and they should be placed in such a way that a malicious user cannot destroy them without making the data less useful. Thus, watermarking does not prevent copying, but it prevents illegal copying by providing a means for establishing the original ownership of a redistributed copy [1] [2] [3]. Most of these techniques were initially developed for still images [4] and then extended to video [5] and audio sources [6] [7].

The increasing use of databases in applications is creating a similar need for watermarking databases. For example, in the semiconductor industry, parametric data on semiconductor parts is provided primarily by three companies: Aspect, IHS, and IC Master. They all pay a large number of people to manually extract part specifications from datasheets. They then license these databases at high prices to design engineers. Companies like Acxiom have compiled large collections of consumer and business data. In the life sciences industry, the primary assets of companies such as Celera are the databases of biological information. The Internet is exerting huge pressure on these data providers to create services that allow users to search and access databases remotely. While this trend is a fortunate thing to end users, it is exposing the data providers to the threat of data theft. They are therefore demanding capabilities for identifying pirated copies of their data. Watermarking on text data [8] and software [9] differs from database watermarking.

We suggest that rights management of relational data through watermarking should become an important topic for database research. Database relations that can be watermarked have attributes which are such that changes in a few values do not affect the applications. But are there real-world datasets that can tolerate a small amount of error without degrading their usability? Consider the ACARS meteorological data used in building weather prediction models [10]. The wind vector and temperature accuracies in this data are estimated to be within 1.8 m/s and 0.5^0 C respectively [10]. The errors introduced by watermarking can easily be constrained to lie within the measurement tolerance in this data. Later in the paper, we report experimental results using a forest cover dataset. It contains measurements for variables such as elevation, aspect, slope, distance to hydrology and roadways, soil type, etc. Small changes in some of the measurements do not affect the usability of this data.

2. WATERMARKING MODEL
In general, any watermarking system consists of three modules such as watermark embedding, attacker channel, and watermark extraction. Each module along with its input and output is shown in the figure 1.

2.1 Watermark Embedding
Embedding module is responsible for insertion of watermark bit string into the relational database. Generally inputs for embedding module are i) watermark bit string to be embedded, ii) original database iii) secret key. By using the insertion algorithm, certain bit locations in the database are altered. Output of encoder module is watermarked database.

2.2 Attacker Channel
As watermarked database is publically available for use, it can be accessed by number of database users. If certain users are malicious, they can try to destroy the original watermark and insert their own. Thus watermarked database can undergo with different types of attacks such as subset selection, subset deletion, and subset alteration.

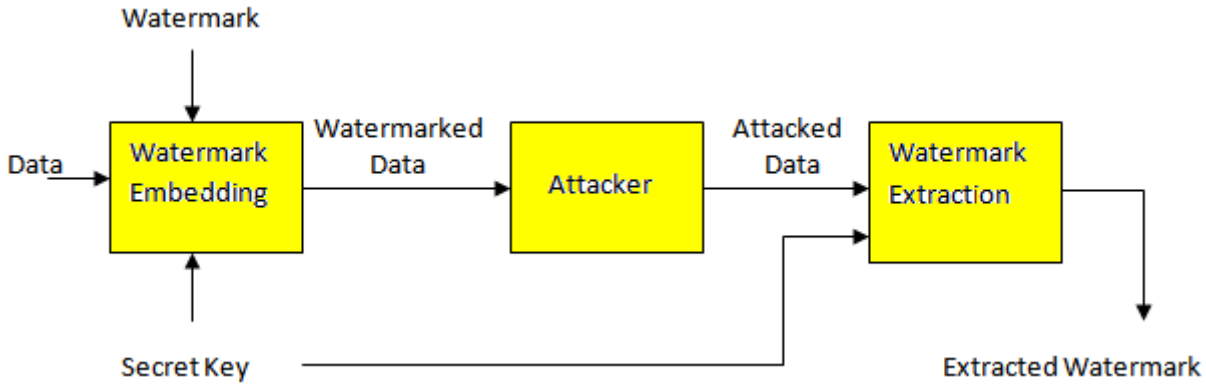

Figure 1: Watermarking Model

2.3 Watermark Extraction

Input for extraction module is watermarked database. Extraction module is responsible for detecting the watermark bit string in the marked database. Detection algorithm verifies the bit locations which are marked by the insertion algorithm and validates the identity of the mark. Output of decoder module is extracted watermark bit string from the database.

3. WATERMARKING SYSTEM

Watermarking scheme consists of two algorithms, watermark insertion and watermark detection. Our improved watermarking scheme is developed by extending the watermarking scheme, which was proposed by Agrawal et al. for watermarking relational databases [11] [12].

Consider a database relation R with primary key P and v numerical attributes $A_0, \ldots, A_{v-1}$. Assume that it is acceptable to change one of ρ least significant bits in a small number of numeric values. The relation has η tuples and a fraction $1/\gamma$ of them will be used for watermarking.

The owner of R has a secret key K. A cryptographic pseudorandom sequence generator [13] [14] S is used to select tuples, attributes, bits, and decide how to change the bits in watermarking. Such a pseudorandom sequence generator produces a sequence of numbers from an initial seed. Without the knowledge of the seed, it is infeasible to compute the next number in the sequence. Different seeds lead to different sequences. For each tuple r, S is seeded with the primary key $r.P$ concatenated with the secret key. Let $S_i(K, r.P)$ denote the i^{th} number in the sequence generated by S. The values of $S_i(K, r.P)$ are uniformly distributed for different primary key values.

3.1 Watermark Insertion

The algorithm as shown in figure 2 inserts the watermark of buyer n into relation R. For each tuple r of R, the algorithm seeds the sequence generator S with the concatenation of the secret key K and the primary key $r.P$ of the tuple. If the first sequence number $S_1(K, r.P) \bmod \gamma = 0$, the tuple is selected. Therefore, on average, one out of γ tuples are selected. For each selected tuple, the algorithm selects exactly one attribute; in particular, it selects attribute i if $S_2(K, r.P) \bmod v = i$. Similarly, it selects least significant bit j from the selected attribute if $S_3(K, r.P) \bmod \rho = j$. On the other hand, the algorithm computes a mask bit x according to $S_4(K, r.P)$; if $S_4(K, r.P)$ is even, then $x = 0$ else $x = 1$. It also selects a watermark bit f_l if $S_5(K, r.P) \bmod L = l$. Finally, the algorithm XOR's the watermark bit with the mask bit and assigns the selected bit with the XOR'ing result. The purpose of using mask bits is to hide the distribution of watermark bits. Note that all selections and computations in watermark insertion are based on $S_i(K, r.P)$ which are uniformly distributed. On average, η/γ bits (or tuples) are used to embed a watermark, and each watermark bit f_l is embedded $\eta/(\gamma L)$ times. Also, note that the secret key is involved in every step of the process. Without knowing the secret key and without comparing multiple watermark copies, buyers are prevented from knowing where the watermark is embedded.

Input:

1. The relation R with scheme $(P, A_0, \ldots, A_{v-1})$
2. The private key K is known only to the owner of the database.
3. The parameters γ, v, and ρ are also private to the owner.
4. Watermark of buyer n: $F(K,n) = H(K|n)$
   // $F(K,n) = (f_0, \ldots, f_{L-1})$

Output:

1. Watermarked relation R

Algorithm:

1) **for each** tuple $r \in R$ **do**
2) **if** ($S_1(K, r.P) \bmod \gamma$ equals 0) **then** // mark this tuple
3) attribute_index $i = S_2(K, r.P) \bmod v$
   // mark attribute A_i
4) bit_index $j = S_3(K, r.P) \bmod \rho$
   // mark least significant bit j
5) mask_bit $x = 0$ if $S_4(K, r.P)$ is even; $x = 1$ otherwise
6) watermark_index $l = S_5(K, r.P) \bmod L$
7) watermark_bit $f = f_l$
   //select watermark bit l
8) mark_bit m = $x \oplus f$
9) set least significant bit j of $r.A_i$ to m.
10) **return** R

Figure 2: Watermark Insertion Algorithm

3.2 Watermark Detection

In watermark detection, a merchant of database relation R would like to determine whether another relation R' was pirated from R and, if so, identify the attacker who distributed R' without authorization. If R' is pirated, the algorithm

assumes that the primary key attribute values as well as the order among marked attributes have not changed (or else can be recovered). Note that *R'* may consist of only a subset of original tuples; it may include some additional tuples that were not in *R*, and some bit values in *R'* could have been changed by the attacker before detection.

The algorithm as in figure 3 initiates a watermark template $F = (f_0, \ldots, f_{L-1})$ as (?,. . ., ?), where "?" indicates that a mark bit is in an unreadable state. It then locates the marked bits exactly as the insertion algorithm does. From each marked bit, the algorithm extracts a watermark bit f_l by XOR'ing the bit value with a computed mask bit. If the marked bit has been changed by the attacker, the extracted f_l may not match its original value. The algorithm uses two counting variables *count*[*l*][0] and *count*[*l*][1] to indicate the number of times that f_l is extracted to be 0 and 1, respectively. After all marked bits are checked, the algorithm assigns 0 (or 1, respectively) to f_l if the ratio *count*[*i*][0]/(*count*[*i*][0]+*count*[*i*][1]) (or *count*[*i*][1]/(*count*[*i*][0]+*count*[*i*][1]) , respectively) is greater than τ , where $\tau \in (0.5, 1)$ is a real parameter that is related to the assurance of the detection process.

A attacker is detected in a subroutine *detect* if the recovered watermark template $F = (f_0, \ldots, f_{L-1})$ matches one of the *N* buyers' watermarks, which is computed in the same manner as the insertion algorithm.

Input:

1. The watermarked relation *R'* with scheme ($P, A_0, \ldots, A_{v-1}$)
2. *K*, γ, *v*, ρ are the same as in watermark insertion

Output:

1. Detected watermark bit string
2. Buyer number

Algorithm:

1)// initiate watermark template and counts

2) watermark template $F = (f_0, \ldots, f_{L-1}) = (?,. . ., ?)$

3) //'?'// represent an unknown value

4) **for each** *i* = 0 to *L*-1 **do** *count*[*i*][0] = *count*[*i*][1] = 0

5) // *count*[*i*][0], *count*[*i*][1] are votes for f_i to be 0 and 1 respectively

6) // scan all tuples and obtain counts for each watermark bit

7) **for each tuple** $r \in R'$ **do**

8) **if** ($S_1(K, r.P)$ mod γ equals 0) **then**
// this tuple was marked

9) attribute_index $i = S_2(K, r.P)$ mod *v*
// attribute A_i was marked

10) bit_index $j = S_3(K, r.P)$ mod ρ
// bit *j* was marked

11) **if** least significant bit *j* of $r.A_i$ does not exist , **then**

12) skip to the next tuple

13) mark_bit m = least significant bit *j* of $r.A_i$

14) mask_bit $x = 0$ if $S_4(K, r.P)$ is even; $x = 1$ otherwise

15) watermark_bit $f = m \oplus x$

16) watermark_index $i = S_5(K, r.P)$ mod *L*

17) *count*[*i*][*f*] = *count*[*i*][*f*] + 1
// update the votes

18) // recover watermark

19) **for each** *i* = 0 to *L*-1 **do**

20) **if** *count*[*i*][0] + *count*[*i*][1] = 0 **then return** *none suspected*

21) $f_i = 0$ if *count*[*i*][0]/(*count*[*i*][0]+*count*[*i*][1]) > τ ;

22) $f_i = 1$ if *count*[*i*][1]/(*count*[*i*][0]+*count*[*i*][1]) > τ ;

23) **return** *none suspected otherwise*

24) $F = (f_0, \ldots, f_{L-1})$

25) // determine a attacker

26) Buyer *n* = *detect (F, K, L, N)* // detect a attacker based on template *F*

27) **if** $n \geq 0$ **then return** *buyer n is the attacker*

28) **else return** *none suspected*

29) // subroutine: detect a attacker

30) *detect* (template *F*, secret key *K*, watermark length *L*, number of buyers *N*)

31) **for each** buyer *n* = 0 to *N*-1 **do**

32) *F' = H(K|n)*

33) **if** *F* matches *F'* then return *n*

34) **return - 1**

Figure 3: Watermark Detection Algorithm

4. EXPERIMENTAL STUDY

This section focuses on the results of proposed improved watermarking system for relational databases. Experiments are performed using the Forest Cover Type dataset, available from the University of California–Irvine KDD Archive.

http://www.kdd.ics.uci.edu/databases/covertype/covertype.html.

The dataset has 5, 81,012 rows, each with 61 attributes. We have added an extra attribute called *id* to serve as the primary key. We have selected the first ten integer-valued attributes along with first 1,00,000 tuples as candidates for watermarking. We assume that the primary key of the relation *R* consists of a single attribute *P* and 10 attributes are available for watermarking. Forest Cover Type dataset schema is as shown in figure 4 and detailed snapshot is as in figure 5.

Watermark insertion as well as detection algorithms are implemented by using JAVA on Net Beans IDE 6.1RC2, along with database program MySQL server 5.1 running at back end.

Field	Type	Null	Key	Default	Extra
ID	int(11)	NO	PRI	NULL	auto_increment
Elevation	int(11)	YES	MUL	NULL	
Aspect	double	YES		NULL	
Slope	double	YES		NULL	
HorzDistToHydrology	double	YES		NULL	
VertDistToHydrology	double	YES		NULL	
HorzDistToRoadways	double	YES		NULL	
Hillshade9am	double	YES		NULL	
HillshadeNoon	double	YES		NULL	
Hillshade3pm	double	YES		NULL	
HorzDistToFirePoints	double	YES		NULL	

Figure 4: Forest Cover Type Dataset Schema

ID	Elevation	Aspect	Slope	HorzDistToHydr...	VertDistToHydr...	HorzDistToRoa...	Hillshade9am	HillshadeNoon	Hillshade3pm	HorzDistToFire...
1	2590	56	2	212	-6	390	220	235	151	6225
2	2804	139	9	268	65	3180	234	238	135	6121
3	2785	155	18	242	118	3090	238	238	122	6211
4	2595	45	2	153	-1	391	220	234	150	6172
5	2579	132	6	300	-15	67	230	237	140	6031
6	2606	45	7	270	5	633	222	225	138	6256
7	2605	49	4	234	7	573	222	230	144	6228
8	2617	45	9	240	56	666	223	221	133	6244
9	2612	59	10	247	11	636	228	219	124	6230
10	2612	201	4	180	51	735	218	243	161	6222
11	2886	151	11	371	26	5253	234	240	136	4051
12	2742	134	22	150	69	3215	248	224	92	6091
13	2609	214	7	150	46	771	213	247	170	6211
14	2503	157	4	67	4	674	224	240	151	5600
15	2495	51	7	42	2	752	224	225	137	5576
16	2610	259	1	120	-1	607	216	239	161	6096
17	2517	72	7	85	6	595	228	227	133	5607
18	2504	0	4	95	5	691	214	232	156	5572
19	2503	28	5	95	10	741	220	229	144	5555

100000 rows fetched in 2.6191s (0.0299s) | Edit | Apply Changes | Discard Changes | First | Last | Search

Figure 5: Forest Cover Type dataset

4.1 Experimental Results of Improved Watermarking

Some experimental results are reported here. We run experiments on MySQL Server 5.1 using JDBC connectivity on a Windows XP Professional with a Dual Core 1.73 GHZ Intel processor, 2 GB of memory, and a 10 GB disk drive.

4.1.1 Imperceptibility

The impact of watermarking on the mean and variance of values of marked attributes is reported here. This experiment was done by varying γ from 10 to 1000 and by varying ρ from 1 to 8. We found a very negligible or no change in the mean value for all the attributes. Table 1 shows changes in variance for different attributes. The values have been rounded to the nearest integer. An empty entry indicates very little or no change. As expected, greater changes in variance occur when ρ is large and γ is small because of larger disturbances in a greater fraction of tuples. Overall, the changes are insignificant given the amount of original variance. Note that if these changes seem significant, v, ρ and γ parameters can be adjusted to reduce the impact of watermarking on the data.

Table 1: Change in variance of different attributes after watermarking

			$\gamma = 1000$			100			10		
Attribute	Mean	Variance	$\rho = 2$	4	8	2	4	8	2	4	8
Elevation	2862	53534.3	--	--	-0.4	--	--	+1.3	--	--	+7.8
Aspect	138.1	10770	--	--	--	--	--	+2.7	-0.17	--	+16.2
Slope	11.8	42.2	--	--	--	--	--	--	--	--	--
Horz-Dist-To-Hydrology	260.5	4118.4	--	--	+1.8	--	--	+5.6	+0.3	+0.4	+33.5
Vert-Dist-To-Hydrology	35.2	1816.6	--	--	--	--	--	--	--	--	+2.5
Horz-Dist-To-Roadways	3344.3	3157206.7	--	-0.2	-9.3	+0.76	+0.5	-11.53	+1.7	+3.7	-9.5
Hillshade-9am	218.2	437.7	--	--	--	--	--	+0.5	--	--	+4
Hillshade-Noon	225.4	277.3	--	--	+0.3	--	--	+1.7	--	+0.13	+20
Hillshade-3pm	139.3	972	--	--	--	--	--	+0.4	--	--	+5.6
Horz-Dist-To-Fire-Points	3589.6	3173452.7	--	--	+0.8	-0.8	+3	+2.5	--	+3.8	-68

When an attribute value is marked, there is 0.5 probability that the value will not change. A bit with value 1 is converted to 0 with probability 0.25 and vice versa. Thus, an original value v will remain v with probability 0.5 and will become $v+\varepsilon$ or $v-\varepsilon$, each with probability 0.25. Hence, if every value of an attribute is equally likely to be selected and it is as likely that the value will be incremented as decremented then the mean and variance will not be affected significantly.

4.1.2 Robustness

Watermarking algorithms must be developed in such a way to make it difficult for an attacker to remove a watermark from the marked database. In particular, the watermarking algorithm should make the watermarked database robust against the following types of attacks: subset deletion attack, subset addition attack, subset alteration attack, and subset selection attack.

4.1.2.1 Subset Alteration Attack

In this type of attack, the attacker alters the tuples of the database through operations such as linear transformation. The attacker hopes by doing so to remove the watermark from the database. The Table 2 indicates that the watermark will remain in the watermarked database even if 90 % of the tuples of the database are altered. This is due to the fact that the proposed algorithm embeds the same watermark everywhere in the database, making this type of attack ineffective. Percent watermark detections for different subset alterations are shown in Table 2.

Table 2: Percentage of watermark detected after subset alteration attack

% Tuples Altered	Percentage of Watermark Detected								
	γ =11			γ=101			γ=1001		
	ρ=2	4	8	2	4	8	2	4	8
10	100	100	100	100	100	98	95	95	95
20	100	100	100	100	100	97	94	90	94
30	100	100	100	100	98	95	92	84	90
40	100	98	88	97	98	93	89	81	84
50	100	96	83	90	97	90	83	78	80
60	100	82	80	73	91	83	81	72	75
70	100	66	72	66	73	80	72	69	68
80	84	62	61	60	71	73	64	61	61
90	65	56	47	57	62	63	61	53	58
100	53	50	41	52	55	59	59	42	57

4.1.2.2 Subset Selection Attack

In this type of attack, the attacker randomly selects a subset of the original database that might still provide value for its intended purpose. The attacker hopes by doing so that the selected subset will not contain the watermark. However, since the proposed algorithm embeds the watermark in the whole database, this attack has little impact. The entries in Table 3 indicate that the watermark will remain in the watermarked database even if the attacker selects a subset as small as 10% of the original database. That is, no matter how small the subset the attacker selects, the watermark will remain in the selected subset and thus maintain the required copyright protection.

Table 3: Percentage of watermark detected after subset selection attack

% Tuples Selected	Percentage of Watermark Detected								
	γ =11			γ=101			γ=1001		
	ρ=2	4	8	2	4	8	2	4	8
10	100	100	100	90	95	88	72	75	80
20	100	100	100	100	98	93	80	82	81
30	100	100	100	100	98	97	86	85	81
40	100	100	100	100	100	98	86	86	83
50	100	100	100	100	100	100	92	90	83
60	100	100	100	100	100	100	96	90	90
70	100	100	100	100	100	100	100	91	95
80	100	100	100	100	100	100	100	94	98
90	100	100	100	100	100	100	100	98	100
100	100	100	100	100	100	100	100	100	100

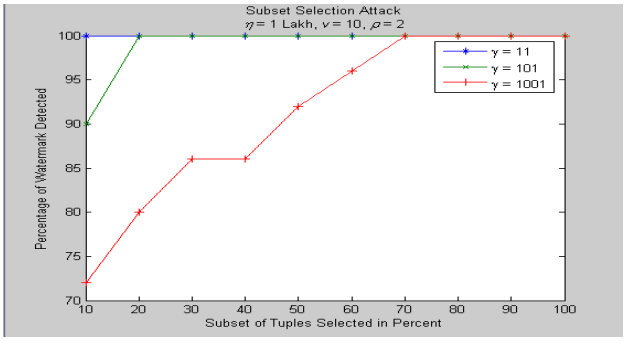

Figure 6: Subset selection attack: as the γ value increases robustness gets decreased

The results of percent watermark detection for selections of different subset of tuples from marked database are plotted as shown in figure 6, and figure 7. These plots show that as the γ value increases the robustness of the system gets decreased.

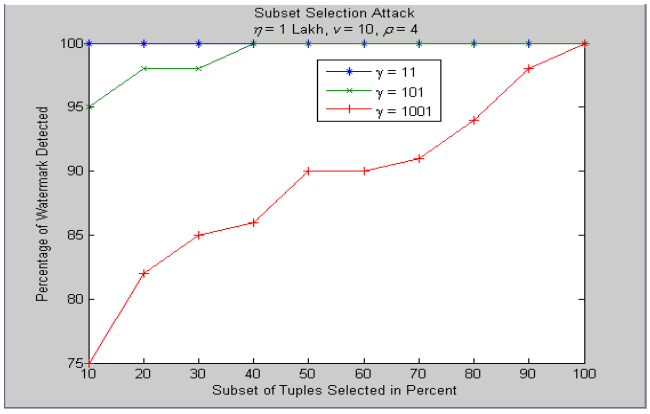

Figure 7: Subset selection attack: as the γ value increases robustness gets decreased

4.1.2.3 Subset Deletion Attack

In this type of attack, the attacker may delete a subset of the tuples of the watermarked database hoping that the watermark will be removed. The entries in Table 4 indicate that the watermark will be removed only and only if most of the database tuples were deleted. That is, even the removal of 90% of the database will not result in removing the watermark. This is due to the fact that the proposed algorithm embeds the same watermark everywhere in the database, making this type of attack ineffective.

Table 4: Percentage of watermark detected after subset deletion attack

% Tuples Deleted	Percentage of Watermark Detected								
	γ =11			γ=101			γ=1001		
	ρ=2	4	8	2	4	8	2	4	8
0	100	100	100	100	100	100	100	100	100
10	100	100	100	100	100	100	100	98	100
20	100	100	100	100	100	100	100	94	98
30	100	100	100	100	100	100	100	91	95
40	100	100	100	100	100	100	96	90	90
50	100	100	100	100	100	100	92	90	83
60	100	100	100	100	100	98	86	86	83
70	100	100	100	100	98	97	86	85	81
80	100	100	100	100	98	93	80	82	81
90	100	100	100	90	95	88	72	75	80

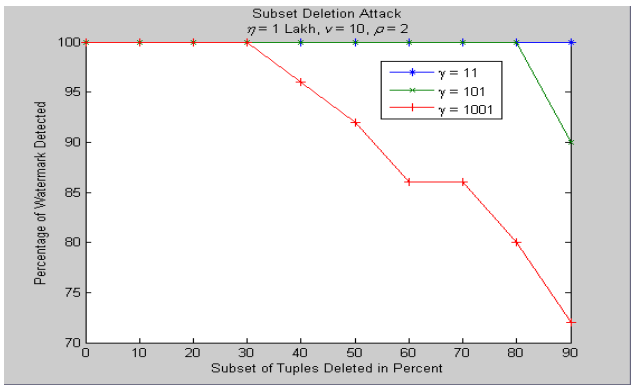

Figure 8: Subset deletion attack: as the γ value increases robustness gets decreased

The results of percent watermark detection for deletions of different subset of tuples from marked database are plotted as shown in figure 8, and figure 9. These plots show that as the γ value increases the robustness of the system gets decreased.

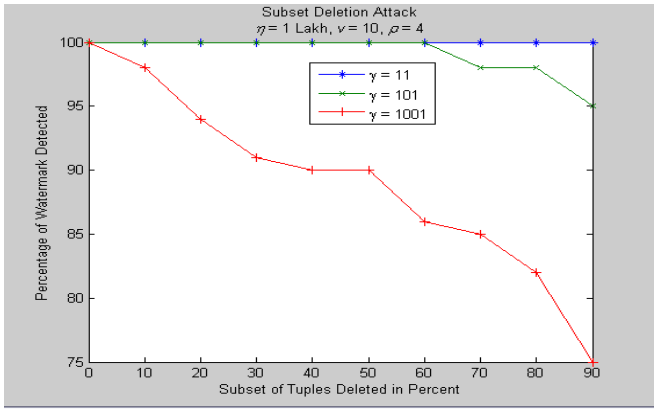

Figure 9: Subset deletion attack: as the γ value increases robustness gets decreased

5. CONCLUSION

In watermarking technique we have implemented two algorithms, first is watermark insertion algorithm and second is watermark detection algorithm. Insertion algorithm first generates a bit string of fixed length and then selects a certain bit for insertion, therefore a single bit may get inserted number of times. Detection algorithm uses majority voting technique for detection of mark bits.

We have tested the watermarking algorithms for different attacks on the same real life Forest Cover Type dataset, with 1,00,000 tuples and 10 attributes available for marking. Our results show that watermark inserted by our improved watermarking technique is more robust and imperceptible than other watermarking techniques.

Our system cannot preserve the key relationships of a database and the join constraints; also the uniqueness and relative constraints of the values are not carefully considered.

Presently it marks only numeric attributes data. In the future, the proposed watermarking technique can be extended to also mark non-numeric attributes.

6. ACKNOWLEDGMENTS

Our thanks to all the experts who have helped us to solve the queries while developing the Improved Watermarking System. Also, special thanks to my guide Dr. Mrs. A. M. Rajurkar, for her consistent and valuable guidance during the development of the system.

7. REFERENCES

[1] I. J. Cox and M. L.Miller. February 1997. A review of watermarking and the importance of perceptual modeling. In Proc. of Electronic Imaging.

[2] N. F. Johnson, Z. Duric, and S. Jajodia. 2000. Information Hiding: Steganography and Watermarking – Attacks and Countermeasures. Kluwer Academic Publishers.

[3] S. Katzenbeisser and F. A. Petitcolas, editors. 2000. Information Hiding Techniques for Steganography and Digital Watermarking. Artech House.

[4] Joseph J. K. ´O Ruanaidh, W. J. Dowling, and F. M. Boland. 1996. Watermarking digital images for copyright

protection. IEEE Proceedings on Vision, Signal and Image Processing, 143(4):250–256.

[5] F. Hartung and B. Girod. 1998. Watermarking of uncompressed and compressed video. Signal Processing, 66(3):283–301.

[6] L. Boney, A. H. Tewfik, and K. N. Hamdy. June 1996. Digital watermarks for audio signals. In International Conference on Multimedia Computing and Systems, Hiroshima, Japan.

[7] S. Czerwinski. Digital music distribution and audio watermarking. Available from http://citeseer.nj.nec.com

[8] M. Atallah and S. Wagstaff. January 1999. Watermarking with quadratic residues. In Proc. of IS&T/SPIE Conference on Security and Watermarking of Multimedia Contents.

[9] C.S.Collberg and C.Thomborson. Feb 2000. Watermarking, Tamper-Proofing, and Obfuscation— Tools for Software Protection. Technical Report 2000-03, University of Arizona.

[10] S. Benjamin, B. Schwartz, and R. Cole. 1999. Accuracy of ACARS wind and temperature observations determined by collocation. Weather and Forecasting, 14:1032–1038.

[11] J. Kiernan, R. Agrawal, 2004. Watermarking Relational Databases, Proc. 28th Int'l Conf. Very Large Databases VLDB.

[12] Y. Li, V. Swarup, and S. Jajodia, January-March 2005. Fingerprinting Relational Databases :Schemes and specialities, IEEE transactions, Vol. 2, No. 1.

[13] J. L. Dugelay and S. Roche. 2000. A survey of current watermarking techniques. In S. Katzenbeisser and F. A. Petitcolas, editors, Information Hiding Techniques for Steganography and Digital Watermarking, chapter 6, pages 121–148. Artech House.

[14] B. Schneier. 1996. Applied Cryptography. John Wiley.

[15] A. Kerckhoffs. La cryptographie militaire. January 1983. Journal des Sciences Militaires, 9:5–38.