Military Technical College
Kobry El-Kobbah,
Cairo, Egypt

8th International Conference
on Electrical Engineering
ICEENG 2012

Hybrid Database Performance Evaluation Achieving Security

By

Wasim Khalil Shalish * A. Z. Ghalwash **, H. M. El-Deeb ***, K. Badran****

Abstract:

The main goal of database security mechanisms is to protect the data stored in the database from unauthorized access or malicious actions in general. Typical database security attacks can be classified as: malicious actions executed by authorized or unauthorized users, and an inference attack occurs in multilevel secure database. The focus of this paper is to investigate the effect of malicious transactions detection and association rule mining implementation on database performance. This paper presents implementation of three mechanisms for detection of malicious transactions in the Oracle 10g DBMS and investigates the performance of the three mechanisms using a telephone database, and implementation of association rule mining using (Apriori) algorithm and investigates the performance using Congress Voting Data set. The experimental results showed that the average performance overhead caused by the activation of malicious transactions detection mechanisms is about 40%, and the hiding association rules performance parameters may be very high to ensure sensitive rules hiding.

Keywords:

DBMS, MTDM (Malicious Transactions Detection Mechanism), MST, MCT.

* Syrian Armed Forces
** Helwan University
*** Modern University for Technology and Information (M.T.I)
**** Egyptian Armed Forces

1. Introduction:

Due to the growth of networked data, security attacks have become a dominant practical problem all information infrastructures. Security violations are typically categorized as malicious actions executed by authorized or unauthorized users, and an inference attack [1].

Typical database security attacks can be classified as: malicious actions executed by authorized or unauthorized users attempts to access or destroy private data, and an inference attack occurs in multilevel secure database when a low level user is able to infer sensitive information through common knowledge and authorized query responses.

Malicious transactions may corrupt data items in the database systems, which decreases the integrity level of the database. Typical Malicious transactions attack can be classified as three categories the first is intentional unauthorized attempts to access or destroy private data; second is malicious actions executed by authorized users to cause loss or corruption of critical data; and third is external interferences aimed to cause undue delays in accessing or using data, or even denial of service.

In fact, several mechanisms needed to detect malicious transactions executed by authorized or unauthorized users have been proposed and/or consolidated in the database arena. Most of these mechanisms can be implemented either externally as an autonomous subsystems separated from the DBMS (sharing the same machine or, perably, in a dedicated machine), internally to the DBMS using database triggers, or internally by using database procedures by compiling them into native code residing in shared libraries.

On the other hand, inference problems are security concerns that arise when users deduce sensitive information about the database from relatively trivial information. Inference problems differ from other security problems in that it is not an issue of unauthorized access to data or leakage of information. Rather, unauthorized inferencing is the result of the nature of the information and the semantics of the application itself. The first step in detecting inference attacks is to analyze whether the results of the user queries reveal any information that is sensitive. Association rule mining helps in detecting inference by providing information about possible associations between the attributes and areas where the database is exposed to inference risks. An effective inference detection system should be able to ascertain that certain specific information can be inferred from the database, and if so, what information was used to perform the inference [2].

One of important classical aspects in the process of association rule mining is true mining of real world knowledge. Recent issue in association rule mining is keeping the confidence of data [3, 4]. Most of information systems contain private information, such as social security numbers, income, disease type, etc. therefore these information should be correctly protected and hided from unauthorized access. Although the security of data has been permanent goal in database management

systems, mining of knowledge and preventing of sensitive knowledge disclosure becomes the most important and highest priority goal in data mining process.

Thus, a complete solution to data security must meet the following three requirements [1]: 1) secrecy or confidentiality refers to the protection of data against unauthorized disclosure, 2) integrity refers to the prevention of unauthorized and improper data modification, and 3) availability refers to the prevention and recovery from hardware and software errors and from malicious data access denials making the database system unavailable. These three requirements arise in practically all application environments.

The structure of the paper is as follows: Section 2 introduces an overview of the mechanisms for malicious transactions detection and prevention in DBMS and problem of security in inference by association rule mining, The implementation of malicious transactions detection mechanisms and inference by association rule mining is presented in section 3 and 4 respectively, and Section 5 reports the experimental results to investigate the effects of the implemented malicious transaction detection mechanisms in a database performance and hiding sensitive association rules mining using a telephone database and Congress Voting Data set respectively. Finally, section 6 concludes the work and introduces future work for further investigation.

2. Related Work:

Ayushi, et al [5] proposed a new mechanism for the detection of malicious transactions in DBMS, the Database Malicious Transactions Detector (DBMTD) mechanism. The DBMTD mechanism is an autonomous application that runs separately from the DBMS in a dedicated machine. The results show that more than 99% of randomly generated transactions (simulating malicious transactions) can be detected and subsequently rolled back while the performance penalty in normal conditions is less than 10%.

Chirag, et al[6] proposed two heuristic blocking based algorithms named ISARC (Increase Support of common Antecedent of Rule Clusters) and DSCRC (Decrease Support of common Consequent of Rule Clusters) to preserve privacy for sensitive association rules. Proposed algorithms cluster the sensitive rules based on some criteria and hide them in fewer selected transactions by using unknowns (“?”). They preserve certain privacy for sensitive rules in database, while maintaining knowledge discovery.

M. Atallah , et al [7] addressed The problem of security in inference by association rule mining, After this beginning, researchers conduct so many methods to solve the security issue of mining results. Generally, modification/sanitization techniques can be categorized into two groups: data blocking and data distortion approaches, some blocking-based techniques are addressed in [8, 9]. The major concept of blocking approaches, are replacing the actual values of the items with “unknown” symbols in the proper transactions. The main reason of using blocking techniques is that

algorithms do not add artificial information in the database. This is so important when the source database contains critical information that extracting wrong known will consequences dangerous effects.

3. Security and Malicious Transactions Detection Trade:

The database malicious transaction mechanism consists of two different phases: transaction profiling corresponds to the identification of the sequence of commands that constitute each valid transaction, and malicious detection conducted during the detection of users executing sequences of commands that potentially represent intrusion attempts.
The mechanism for the detection of malicious transactions in DBMS can be implemented in three ways; externally as an autonomous subsystems separated from the DBMS, internally to the DBMS using database triggers, and internally to the DBMS using database procedures by compiling them into native code residing in shared libraries. Figure (1) presents the basic architecture of a database system with the malicious transactions detection mechanism.

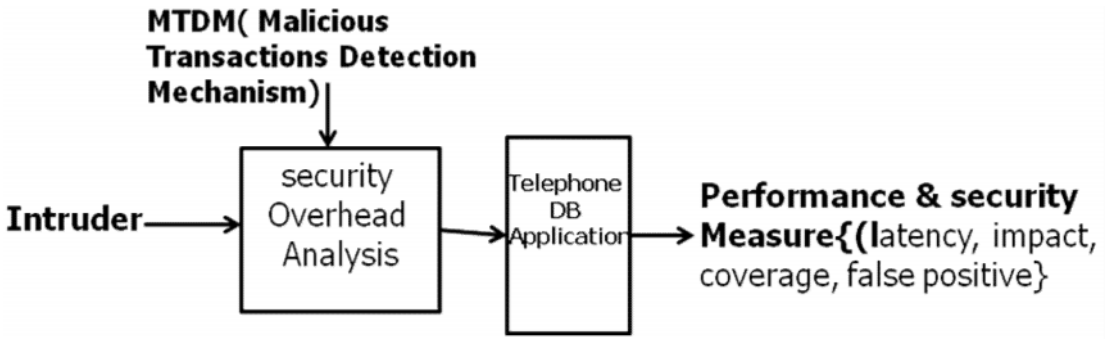

Figure (1): *Architecture of a database application with the malicious transactions detection mechanism*

4. Security and Association rule Mining Trade:

The association rule hiding problem can be considered as a deviation of the well identified database inference control problem in statistical and multilevel databases. The primary goal in database inference control is to guard access to sensitive information that can be obtained through non sensitive data and inference rules. In association rule hiding, we think about that it is not the data itself but somewhat the sensitive association rules that produce a breach to privacy.
For the simplicity of presentation and without loss of generality, we make the following assumptions in this implementation:
We want to extract all association rules which satisfy minimum support transaction (MST), minimum confidence transaction (MCT) .support is a measure of the frequency of a rule. The confidence is a measure of the strength of the relation between sets of rules. Association rule mining algorithms scan the database of

transactions and calculate the support and confidence of the candidate rules to determine if they are considerable or not. A rule is considerable if its support and confidence is higher than the user specified minimum support and minimum confidence threshold. In this way, algorithms do not retrieve all possible association rules that can be derivable from a dataset, but only a small subset that satisfies the minimum support and minimum confidence requirements set by the users. Apriori association rule-mining algorithm works as follows. It finds all the sets of rules that appear frequently enough to be considered relevant and then it derives from them the association rules that are strong enough to be considered interesting. The major goal here is to preventing some of these rules that we refer to as "sensitive rules", from being revealed. We want to hide association rules using the best way by deleted all records witch containing sensitive rule from data set. We are interested in investigating the performance of association rules (hiding failure (HF), dissimilarity (DIS), artificial pattern (AF), side effect (SEF), and miss cost (MC)).
Figure (2) presents the basic architecture of a database system with the association rule mechanism.

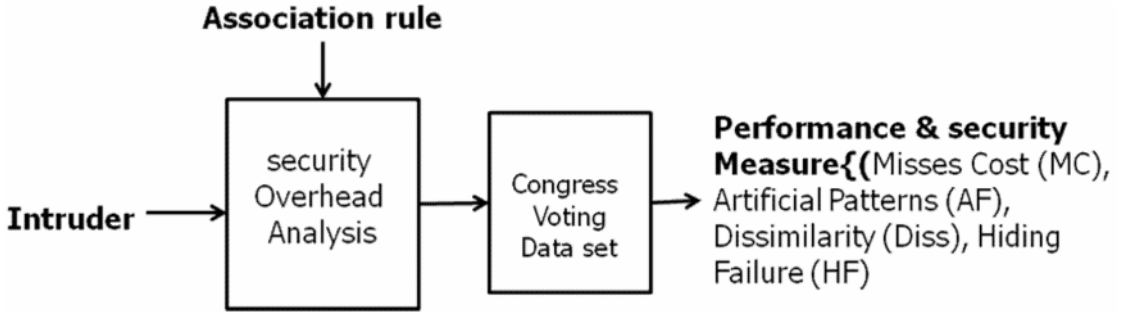

Figure (2): *Architecture of a database application with the association rule procedure*

5. Discussion and Experimental Results:

5.1 Experimental Setup:

The Oracle™ DBMS is one of the leading databases in the market and as one of the most complete and complex database. It represents very well all the sophisticated DBMS available today. For that reason, we have chosen the Oracle 10g DBMS [10] as a case study.
The telephone database is designed to simulate telecommunication centrals, and based on a database with 11th tables with several relationships and specifies four different types of transactions: T1 (a set of eight commands represents a telephone bill payment), T2 (a set of two commands represents delete subscriber), T3 (a set of three commands represents information about subscriber), and T4 (a set of four commands represents adding subscriber), otherwise it is considered to be malicious transaction.
The data set Congress Voting Data set [11] includes votes for each of the U.S. House of Representatives Congressmen on the 16 key votes identified by the CQA. The CQA lists nine different types of votes: voted for, paired for, and announced for

(these three simplified to yea), voted against, paired against, and announced against (these three simplified to nay), voted present, voted present to avoid conflict of interest, and did not vote or otherwise make a position known (these three simplified to an unknown disposition). Number of Instances: 435 (267 democrats, 168 republicans) Number of Attributes: 16 + class name = 17 (all Boolean valued)

5.2 Performance evaluation measures for the MTDMs:

The efficiency of the MTDMs mechanisms can be characterized by the following measures:
Impact on the database performance (performance overhead introduced by the mechanism)

$$\text{Impact\%} = \frac{(\text{tpm (auditing)} - \text{tpm (MTDM)}) * 100}{\text{tpm (auditing)}} \quad (1)$$

Where tpm: the number of transactions executed per minute.
Latency: (time between the execution of the malicious transaction and its detection). (2)

5.3 Performance evaluation measures for the Association Rules:

The efficiency of the Association rule mechanisms can be characterized by the following measures:
Dissimilarity quantifies the difference between the original and the sanitized datasets by comparing their histograms, where the horizontal axis contains the items in the dataset and the vertical axis corresponds to their frequencies. It is calculated as follows:

$$\text{Diss(D,D}') = \frac{1}{\sum_{i=1}^{n} fD(i)} * \sum_{i=1}^{n} [fD(i) - fD'(i)] \quad (1)$$

Where fD(i),fD'(i) represents the frequency of the ith item in the dataset D, and D' respectively, and n is the number of distinct items in the original dataset D.
Misses Cost (MC) This measure quantifies the percentage of the nonrestrictive patterns that are hidden as a side-effect of the sanitization process. It is computed as follows:

$$\text{MC} = \frac{|R'_P(D)| - |R'_P(D')|}{|R'_P(D)|} \quad (2)$$

Where R'_P (D) is the set of all non sensitive rules in the original database D and R'_P (D') is the set of all non sensitive rules in the sanitized data base D'. As one can notice, there exists a compromise between the misses cost and the hiding failure,

since the more sensitive association rules one needs to hide, the more legitimate association rules is expected to miss.

Side-Effect Factor (SEF) Similarly to the measure of misses cost, is used to quantify the amount of non-sensitive association rules that are removed as an effect of the sanitization process. It is defined as follows:

$$\mathrm{SEF} = \frac{|P| - (|P'| + |Rp(D)|)}{|P| - |Rp(D)|} \tag{3}$$

Artificial patterns (AF) this measure quantify the percentage of the discovered patterns that are artifacts. It is computed as follows:

$$\mathrm{AP} = \frac{|P'| - |P \cap P'|}{|P'|} \tag{4}$$

Where P is the set of association rules discovered in the original dataset D and P' is the set of association rules discovered in D'.

Hiding Failure (HF) This measure quantifies the percentage of the sensitive patterns that remain exposed in the sanitized dataset. It is defined as the fraction of the restrictive association rules that appear in the sanitized database divided by the ones that appeared in the original dataset, formally:

$$\mathrm{HF} = \frac{|R_P(D')|}{|R_P(D)|} \tag{5}$$

where RP (D') corresponds to the sensitive rules discovered in the sanitized dataset D', RP (D) to the sensitive rules appearing in the original dataset D. Ideally, the hiding failure should be 0%. The performance metrics for privacy preserving association rule mining algorithms are given in [12].

5.4 Experiment 1: Impact of MTDMs on database performance

Understanding the impact that the implemented MTDM and the auditing mechanisms have on the database performance is one of the most important features. Especially if you have a high-volume environment and you have stringent auditing requirements that include auditing activities that happen a lot. An important note is that the auditing mechanism affects performance and that the impact is directly proportional to how much you audit [13].

Three configurations have been considered in the evaluation of the impact on the database performance: First, Baseline (Oracle 10g fully tuned for performance and without using the audit mechanism); Second, Audit (Oracle 10g using the audit mechanism but without malicious transactions detection); and finally, the MTDM (Oracle 10g using audit with the malicious transactions detection mechanism), As the impact in the performance caused by the audit and the implemented MTDM may depend on the transactions submitted to the database, we have decided to measure the

performance under the considered four authorized transactions. Figure (3) presents the results for the baseline configuration. The Y axis corresponds to the number of transactions executed per minute (tpm), and the X axis represents the four authorized transactions (T1 to T4).

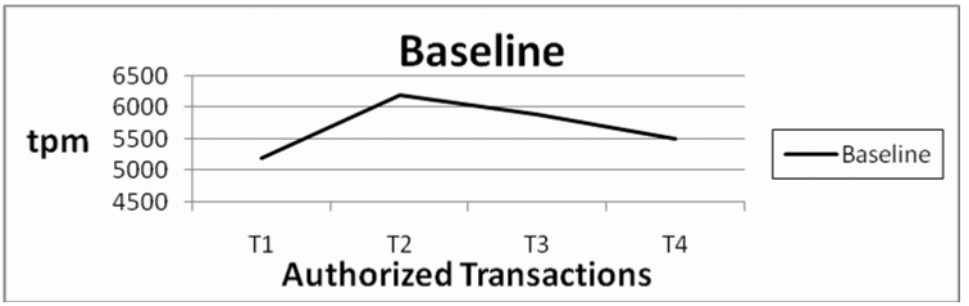

Figure (3): *Baseline configuration Performance*

Results in Figure (3), show that for the baseline configuration, the number of transactions executed per minute (tpm) is related to the number of commands in each transaction. For example, transactions T2 and T3 achieve an execution rate of about 6200 tpm and 5900 tpm respectively, as transaction T2 contains two commands and transaction T3 contains three commands. While transactions T1 and T4 achieve an execution rate of about 5200 tpm and 5500 tpm respectively, as transaction T1 comprises of eight commands and transaction T3 comprises of four commands. Figure (4) presents the results for the audit configuration.

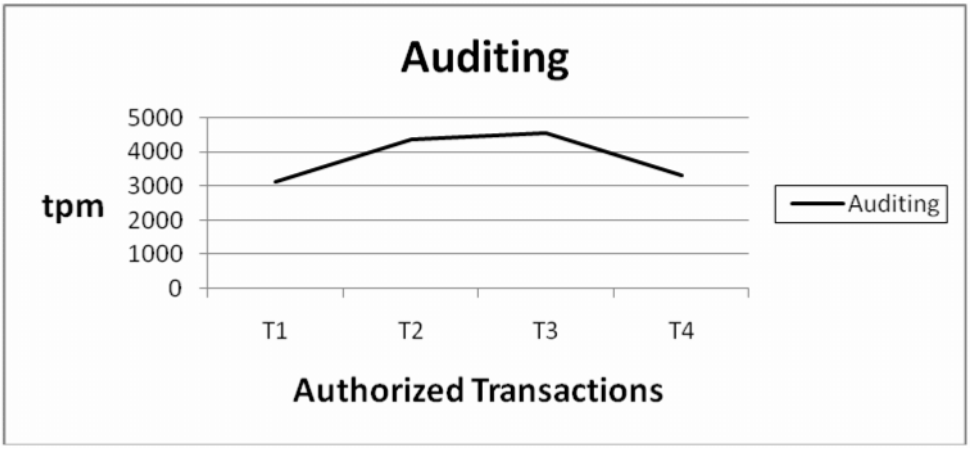

Figure (4): *Performance for auditing configuration*

Results in Figure (4) show that the impact in the database performance caused by the auditing mechanism is highly related to how much you audit. As shown, the number of transactions executed per minute (tpm) for transactions T1 and T4 are about 3100 and 3200 respectively, as the number of tables that you audit in transactions T1 and T4 are 4 tables. While the number of transactions executed per minute (tpm) for transactions T2 and T3 are about 4200 and 4300 respectively, as the number of tables

that you audit in transactions T2 is 1 table and T3 is 2 tables. Figure (5) presents the results for the three configurations considered.

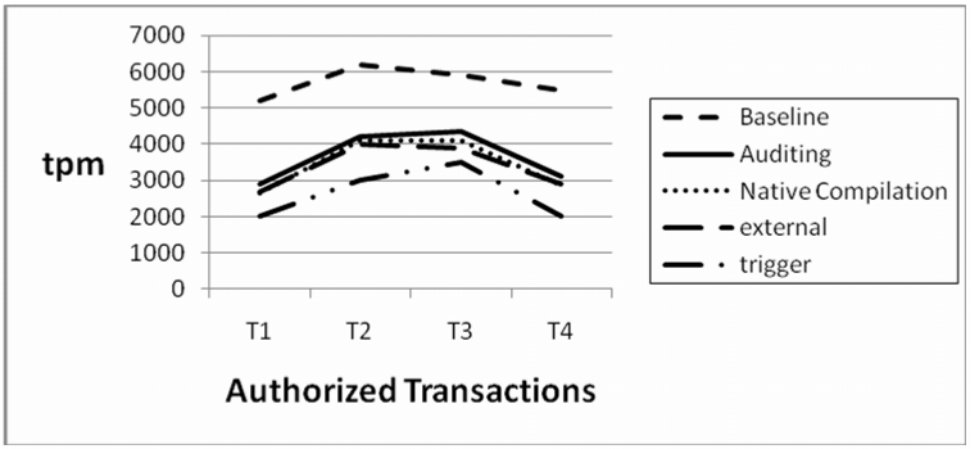

Figure (5): *Performance for the three configurations*

From the analysis of the results presented in Figure (5), we can derive the performance overhead introduced by the standard auditing mechanism (without malicious transactions detection) and by the three mechanisms. Table (1) shows the resultant overhead.

Table (1): *Performance overhead*

Configuration	Impact %			
	T1	T2	T3	T4
Auditing	40%	33%	35%	40%
Native MTDM	45%	36%	37%	45%
External MTDM	45%	35%	38%	44%
Trigger MTDM	55%	41%	42%	54%

As shown in the previous table, the average performance overhead caused by the activation of the auditing mechanism is fairly high (about 37%), about 40% for the external and the native mechanisms, and about 48% for database trigger mechanism. This explains the expected better performance of the external and native mechanisms over the database trigger mechanism.
Detailed investigations could lead to the following remarks:

- The performance penalty added by the native and external mechanisms is quite small (less than 5% independently on the transaction submitted with respect to the allowable authorized transactions). In other words, the performance penalty is due to the database auditing mechanism.

- The impact on performance is related to how much you audit, so try to decrease auditing (as long as it does not have an adverse impact to database security or your ability to pass an audit). This will also make your reviewing process less tedious, and will require less disk space, etc.
- If your requirements or implementation changes and you change your audit policy in a substantial way, you must go through an entire change management process and comprehensive testing to avoid surprises in production.
- If you want to avoid this overhead or if you have extreme audit requirements including auditing of DML or SELECT, You either have to accept the performance impact or consider an auditing solution that is not based on the database doing more I/O.
- Finally, remember that the performance impact is not only in the generation of the audit trail itself. The audit trail needs to be moved elsewhere because it can't stay in the database or on the OS. It should be moved fairly quickly for better separation of duties. This means that you also need a process that keeps reading these records, copying them elsewhere, and deleting them. This will add to the impact on performance [13].

5.5 Experiment 2: Latency

Different behaviors concerning database performance and functionality are to be expected for the different mechanisms considered. Figure (6) shows the results obtained for latency time of the three implemented mechanisms. Malicious transactions are submitted by an external application that connects to the DBMS using valid credentials (i.e., valid username, password, and user privileges). The malicious transactions are simulated by randomly generating transactions that access and modify the telephone database tables. An important aspect is that, the number of commands in each transaction ranges from 1 to 8.

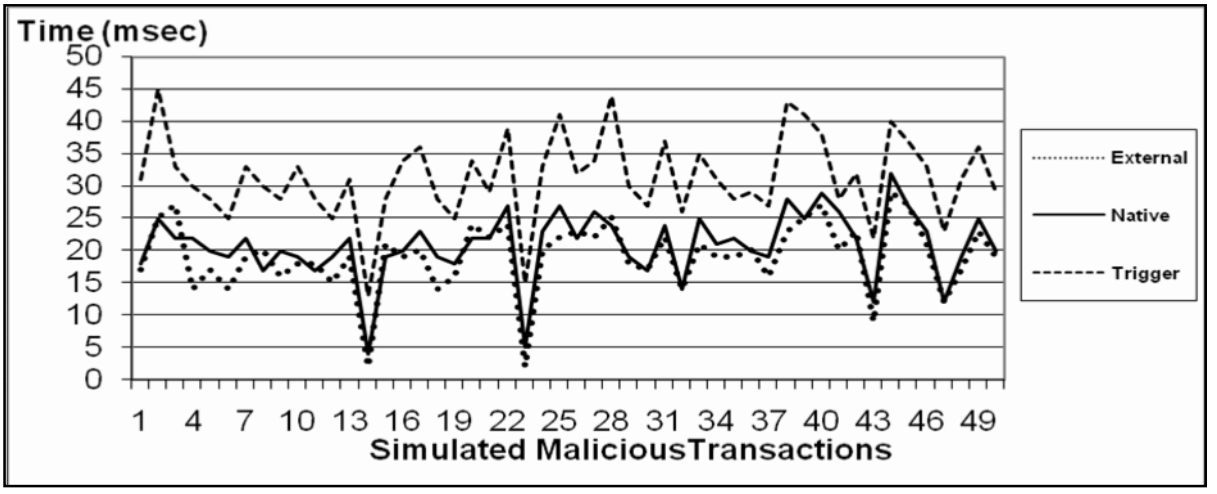

Figure (6): *The three mechanisms Latency*

From Figure (6), it could be noticed that for all the simulated malicious transactions, the latency of the database trigger mechanism is more than the latency of both the external and the native mechanisms. Also, it could be seen that the latency varies between 2 msec to 29 msec for the external mechanism, between 4 msec to 32 msec for the native mechanism, and between 13 msec to 45 msec for the database trigger mechanism, as expected, the external and native mechanisms outperform the database trigger mechanism in term of database performance. The average latency for the native mechanism is about 20 msec, about 19 msec for the external mechanism, and about 31 msec for the database trigger mechanism, Moreover, for all the considered transactions, the latency varies as the number of commands in each transaction varies, the higher the number of commands the higher the latency.

5.6 Experiment 3: Association Rules Mining Methodology

A sample transaction database D taken from [11] is shown in Table (2). TID shows unique transaction number, Suppose MST and MCT are selected 25% and 58% respectively.

***Table (2):** sample data set*

TID	Class Name	handicapped-infants	water-project-cost-sharing	adoption-of-the-budget-resolution	physician-fee-freeze	el-salvador-aid	religious-groups-in-schools
1	republican	N	Y	N	y	Y	Y
2	republican	N	Y	N	y	Y	Y
3	democrat	?	Y	Y	?	Y	Y
4	democrat	N	Y	Y	n	?	Y
5	democrat	Y	Y	Y	n	Y	Y
6	democrat	N	Y	Y	n	Y	Y
7	democrat	N	Y	N	y	Y	Y
8	republican	N	Y	N	y	Y	Y
9	republican	N	Y	N	y	Y	Y
10	democrat	Y	Y	Y	n	N	N

Table (3) shows frequent rules satisfying MST, generated from sample database D, in following; the possible number of association rules satisfying MST and MCT, generated by Apriori algorithm are given: (20). Suppose the rule: (el-salvador-aid=y 212 ➔ religious-groups-in-schools=y 197) are specified as sensitive and should be hidden in sanitized database, the main approach to hide sensitive association rules is to reduce the support or the confidence of the rules. However in my approach I was deleted all records witch containing sensitive rule from data set, then we evaluated

the performance and security metrics (hiding failure, dissimilarity, artificial pattern, side effect, miss cost).

***Table (3):** Best rules inference extracted from original dataset with MCT=0.58 and MST=0.25*

TID	Rules
1	adoption-of-the-budget-resolution=y physician-fee-freeze=n 219 ➔ Class Name=democrat
2	adoption-of-the-budget-resolution=y physician-fee-freeze=n aid-to-nicaraguan-contras=y 198 ➔ Class Name=democrat
3	physician-fee-freeze=n aid-to-nicaraguan-contras=y 211 ➔ Class Name=democrat 210
4	physician-fee-freeze=n education-spending=n 202 ➔ Class Name=democrat 201
5	physician-fee-freeze=n 247 ➔ Class Name=democrat 245
6	Class Name=democrat el-salvador-aid=n 200 ➔ aid-to-nicaraguan-contras=y 197
7	el-salvador-aid=n 208 ➔ aid-to-nicaraguan-contras=y 204
8	el-salvador-aid=y 212 ➔ religious-groups-in-schools=y 197

Table (4) shows the association rule evaluation performance results:

***Table (4):** performance results*

Parameters	Hiding by Deleting Sensitive Rules
HF	0%
MC	0%
AP	20%
DISS	60%
SEF	1.10

As shown in Table (4), the number of sensitive rules in sanitized data set equal to zero, most of the developed privacy preserving algorithms are designed with the goal of obtaining zero hiding failure. Thus, we hide all the patterns considered sensitive from the original data set. The number of non- sensitive patterns discovered from the original database D, and the sanitized database is the same, since we hide all the patterns considered sensitive from the original data set, thus the MC is equal to 0%. The percentage of the discovered patterns that are artifacts (AP) is 20%. The percentage of the dissimilarity (DISS) between the original and the sanitized datasets is 60%. The amount of non-sensitive association rules that are removed as an effect of the sanitization process is (1.10). As shown, performance results of this

experiment may be very high in terms of artificial patterns (AP), dissimilarity (DISS), and side effect factor (SEF) to ensure sensitive rules hiding.

6. Conclusion and future work:

This work presents investigating the Performance Evaluation of malicious transactions detection mechanisms and inference by association rules mining. The experimental results for MTDM showed that the average performance overhead caused by the activation of the external and the native mechanisms is 40%, and 48% for the database trigger mechanism. As a result, the external and the native mechanisms outperform the database trigger mechanism in term of database performance. Also, the experimental results for inference by association rules showed that overhead measurements are very high to ensure hiding of all sensitive rules.

As a future work, we are planning to propose solution approach to optimize between hiding failure as security over head and ((AF), (Diss), (SEF), (MC)) as database performance metrics.

References:

[1] E. Bertino, R. Sandhu, Database Security Concepts, Approaches, and Challenges, IEEE Transactions on Dependable and Secure Computing, Vol. 2, No. 1, 2005.

[2] Harry S. Delugach and Thomas H. Hinke: Wizard: A Database Inference Analysis and Detection System , Vol. 8, NO. 1, 1996

[3] Y.-H.Wu, C.-M. Chiang, and A. L. P. Chen. Hiding sensitive association rules with limited side effects. IEEE Transactions on Knowledge and Data Engineering, 19(1):29–42, 2007.

[4] J. Natwichai, X. Li, and M. Orlowska. A reconstructionbased algorithm for classification rules hiding. In Proceedings of the 17th Australasian Database Conference (ADC 2006), pages 49–58, 2006.

[5] Ayushi., Sharma, A., and Bansal, R., "Detection of Malicious Transactions in DBMS" ,International Journal of Information Technology and Knowledge Management," July-December 2010, Vol. 2, No. 2, pp. 675-677

[6] Ch.modi, U.rao An Efficient Solution for Privacy Preserving Association Rule Mining, (IJCNS) International Journal of Computer and Network Security,Vol. 2, No. 5, May 2010.

[7] M. Atallah, E. Bertino, A. Elmagarmid,M. Ibrahim and V. Verykios. Disclosure limitation of sensitive rules. Proc. Of IEEE Knowledge and Data Engineering Exchange Workshop (KDEX), November 1999.

[8] Y. Saygin, V. Verykios and C. Clifton. Using unknowns to prevent discovery of association rules. ACM SIGMOD

Record, vol. 30, no. 4, 2001.

[9] V. Verykios, A. Elmagarmid, E. Bertino, Y. Saygin and E. Dasseni. Association Rule Hiding. IEEE Trans. On Knowledge and Data Engineering, 16(4), 2004.

[10] Michele, C., Oracle® Database Concepts 10g Release 1 (10.1), Redwood, 2003.

[11] Schlimmer, J. C., Concept acquisition through representational adjustment. Doctoral dissertation, Department of Information and Computer Science, University of California, Irvine, CA, 1987.

[12] C.C. Aggarwal, P.S. Yu. Privacy-Preserving Data Mining: Models and Algorithms, Springer, Heidelberg, pp. 267–286, 2008.

[13] Natan, B.N., How to Secure and Audit Oracle 10g and 11g , New work, pp. 228-229, 2009.