Journal of mathematics and computer science 13 (2014), 212-225

Contents list available at JMCS

Journal of Mathematics and Computer Science

Journal Homepage: www.tjmcs.com

Controlling the False Alarm in an Intrusion Tolerant Database System Using Significance Degrees of Data Objects

Zeinab Falahiazar[1] , Mohsen Rohani[2], Alireza Falahiazar[3]

[1, 3]*Computer Engineering Department, Islamic Azad University, Science and Research Branch, Tehran, Iran*

[2]*Computer Engineering Department, Islamic Azad University, South Tehran Branch, Tehran, Iran*

zfalahiazar@srbiau.ac.ir[1], m.rohani@niopdc.ir[2], a_falahiazar@srbiau.ac.ir[3]

Article history:
Received August 2014
Accepted September 2014
Available online October 2014

Abstract

Traditional database security mechanisms focus on either protection or prevention. However, in practice all attacks are not avoidable. To solve this problem, Intrusion Tolerant Database Systems (ITDBs) were introduced. An ITDB uses new generation database security mechanisms to guarantee specified levels of data availability, integrity and confidentiality in the presence of successful attacks. A key part of an ITDB is the intrusion detection (ID) which informs the system about attacks. One of the problems in using ID is the false alarm that will lead to the reduction of the "availability" or "integrity". This paper presents an intelligent method to control false alarm. In this method, we will use the significance degrees of data objects to determine the anomaly threshold adaptively, as the "availability" and "integrity" required by the data objects are satisfied.

Keywords: Database Security, Intrusion Tolerance, Intrusion Detection, false alarm, anomaly threshold, adaptive controller.

1. Introduction

Today because of the widespread use of computer systems, internet websites and web based applications; database security issues have become very important worldwide. Traditional database security mechanisms have some limitations in providing the required security for modern information systems. Such mechanisms only focus on protection of data against different attacks and are not capable of detecting attacks or taking the proper action against them. For example, in most cases attacks like SQL Injection pass from all access control mechanisms which Database Management Systems (DBMS) provide traditionally. These attacks are usually done through web applications

which have vast accessibility [1, 2, 25]. For confronting with such attacks, we need mechanisms like intrusion tolerant database systems [3, 4, 5].

ITDB is considered as a framework for a transaction based self-healing database system. This framework will empower a database, aiming to deliver a continuous reliable service, to heal itself under malicious (but authorized) transaction attacks [21, 22, 23, 24]. Intrusion Detector is regarded as one of the main ITDB parts.

The Intrusion detection systems [6, 7, 8, 9, 28, 31] are used as monitoring systems to detect actions which threaten the integrity, confidentiality and availability of a resource. In ITDB the task of ID is to identify malicious transactions.

There are two general approaches to intrusion detection systems: anomaly detection and misuse detection. The anomaly detection approach has two modes: training mode and detection mode. In training mode, a profile is built that shows what is expected to be seen. In detection mode, the system alerts on any input which is outside of the profile. The second approach, misuse detection, is based on comparison with the behaviors of known attacks. Unlike the second approach, the first approach can detect any novel attack. However, a key problem of anomaly detection is the huge number of false positive alarms. In this paper we have focused on the anomaly detection approach.

Using ID in ITDB has certain limitations one of which is that, in general, ID is much slower than the transaction processing. Therefore, when a malicious transaction is recognized, a great number of transactions, which have been affected by malicious transaction, have committed and the damage has been spread. Despite this substantial delay in detection, a self-healing database system must be able to live.

Recent researches present two new techniques to solve this problem: attack isolation technique [10, 12, 26] and multi phase damage confinement technique [11]. Although both techniques have the same goal, they employ different methods in their functions. The multi phase damage confinement technique guarantees that no damage will spread after detecting a malicious transaction. In this technique, however, no action is taken to decrease the damage spread during the detection delay time. On the contrary, the suspicious users are isolated in the attack isolation technique and as a result, there will be fewer damages during the detection delay time. Using this method will result in a decrease in the repair costs as well. Moreover, it is not possible for other users to see the updates of suspicious users while using this technique. All other users can see the updates only when the innocence of suspicious users is proven. For this reason, this technique is considered as a more secure method.

Another limitation of ID is false alarm. False negative alarm, which is generally created due to the low level of Anomaly Threshold, reduces “integrity”. This is done by allowing malicious transactions to execute. Positive false alarm, which is usually created due to the high level of threshold, reduces “availability”. This is done by rejecting user’s following requests.

In this paper, we intend to present a practical solution to control false alarm in ITDB (which uses an ID), as a tradeoff done between “integrity” and “availability” of the data objects.

The rest of the paper is organized as follows: In section 2 the problems of using ID in ITDB will be explained. Section 3 introduces our method to control the false alarm rate. We will explain the implementation and evaluation of the proposed method in section 4. Finally, the conclusion will be presented in section 5.

2. Problem

ITDB uses the new generation of database security techniques to encounter the attacks which succeed to pass the traditional protection mechanisms. One of these techniques is Intrusion Detection which

warns the system about the attacks. However, intrusion detection has limitations such as detection delay and false alarm. Intrusion tolerance techniques are used in ITDB architecture to overcome these limitations. These techniques include multiphase damage confinement, attack isolation and optimized attack isolation. This section explains the abovementioned techniques in brief.

2.1. Multiphase damage confinement

Multiphase Damage Confinement technique contains an initial confinement phase which ensures no damage (caused by the malicious transaction) leaks out. When a malicious transaction is detected, this phase prevents accessing any data object which is updated after the malicious transaction committed. Furthermore, active transactions, which may have read a number of confined data object, are aborted to prevent damages to spread.
This technique has three unconfining phases to unconfine the data objects that are mistakenly confined during the initial confinement phase and the data objects that have been cleaned. In these phases certain dependency graphs are used to find mistakenly confined data objects. The first and second phases of relaxation are used to reduce the time an undamaged object is mistakenly confined. In the third phase, the damaged objects are cleaned.

In this technique, no method has been presented to determine anomaly threshold that is proportional to the change of environmental conditions.

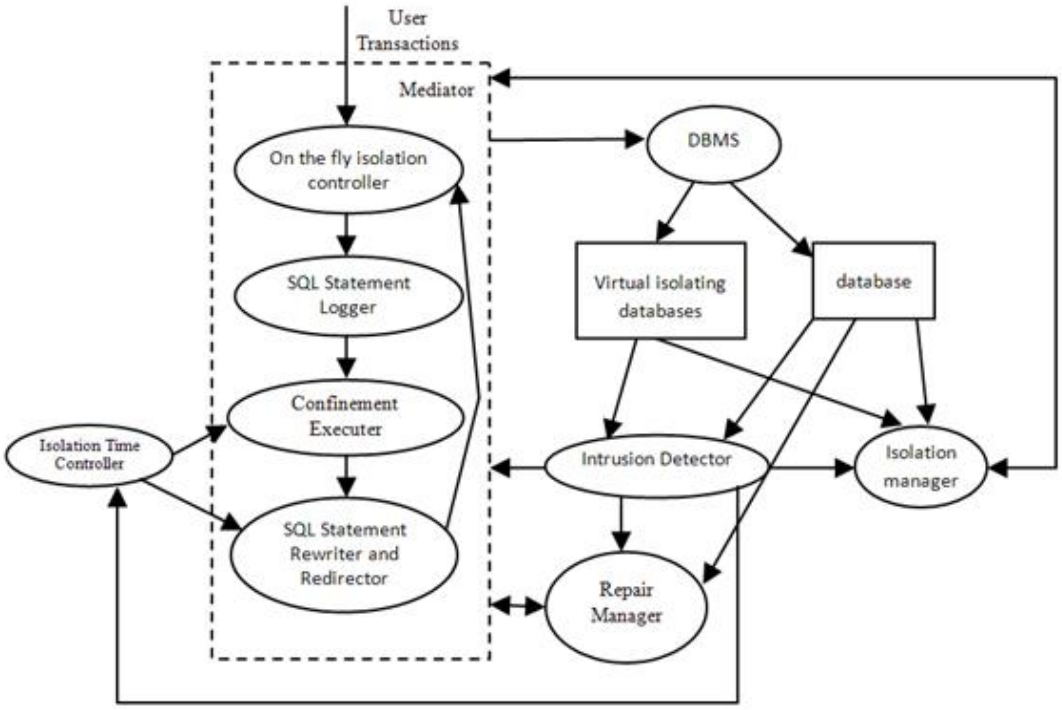


Figure1. Architecture of ITDB system

Figure 1 shows the main components of the ITDB architecture which use a combination of Intrusion Tolerance techniques. In this architecture, ID is responsible for the report of malicious or suspicious transactions (users) which have to be isolated. The Containment Executer is responsible for the initial confinement phase and the confinement enforcement. The Repair Manager is responsible for the unconfining phases.

2.2. Attack isolation technique

In this technique suspicious activities are isolated before an intrusion is reported as unquestionable. Isolation is done based on users. ID is responsible for reporting suspicious users that should be isolated. For this purpose, ID will assign an anomaly degree which specifies the way of abnormality of transaction (based on the transaction's behavior) for each transaction. The anomaly degree of each user is obtained from combination of anomaly degree of his/her transactions in a session. ID contains two alarm thresholds. If anomaly degree is above the first threshold (and below of the second), user is reported as suspicious. If the degree of anomaly is above the second threshold, then that user is reported as malicious. The first threshold is determined based on the probability that a transaction turns out to be suspicious. This can be concluded on the previous detections to some extent.

In this technique, no intelligent method is presented to determine anomaly threshold according to the change of environmental conditions as well. Furthermore, transactions of a suspicious user update suspicious versions of data objects which are produced for this user. However these transactions read main versions of data objects if suspicious versions have not thus far been produced for this suspicious user. If the innocence of an isolated user was proved then the updates of that user must be merged back into the real database.

The merging process may cause conflicts between the real database and the isolated one. In this technique a precedence graph has been used for identification of inconsistencies. If a precedence graph (obtained as a result of merging the history of the isolated user and the history of real database after isolation of that user) is acyclic, it will mean that there are no inconsistencies. Otherwise, if the graph has cycles, all the cycles are to be broken by running compensation transactions of certain transactions.

After resolving of inconsistencies, values of suspicious versions, maintained for the suspicious user, are replaced with values of trustworthy versions and then suspicious versions are omitted.

As shown in figure 1, the Mediator proxies transactions of each user. The SQL Statement Logger keeps each SQL statement, in addition to its variables, in SQL Statement Table. The SQL Statement Rewriter and Redirector does isolation with keeping additional versions for data objects that have been updated by an isolated transaction. The On-the-fly Isolation Controller will impose two restrictions for accuracy of a merging process. First, during the merge, no new transaction of the once isolated user can be executed; and second, no new transaction can be executed on the locked part of database. The Merging Manager is responsible for resolving inconsistency and merging process.

2.3. Optimized attack isolation

This technique makes it possible to use a combination of both attack isolation and multiphase damage containment techniques simultaneously in the system [12, 26]. In this technique an isolation time controller is used in the architecture of an optimized ITDB. Using this controller has two main advantages. The first positive point is that the isolation process is enforced for each suspicious user in a certain amount of time. Therefore inconsistency, caused by prolongation of the isolation time, will be decreased. The second advantage of using such controller is that we can prevent a long isolation time in case the list of suspicious user is changed by an attacker. Thus, security will be increased.

It should be noted that the isolation time for each suspicious user is estimated by parameters like the average of detection time at previous attacks and the average of the users' session time.

When the isolation time for a suspicious user terminated If ID could not prove maliciousness/innocence of the user a message announcing the user's innocence would be sent to the SQL Statement Rewriter and Redirector, despite the user has initially been regarded as suspicious.

Under such circumstances, if maliciousness of a user is proven later, the isolation time controller will inform the Confinement Executer and then the multiphase damage containment technique is used.

In the attack isolation technique, suspicious transactions can be executed on their own version of the requiring objects. In real world, as a result of execution of these transactions, damages may be incurred to the owners of the system. For example, in a banking system, certain amount of an account may withdraw as a result of the execution of a suspicious transaction in a way that to compensate being nontrivial. Therefore, a weight factor can be imposed in the anomaly degree of transactions for faster detection of attacks and reducing damages incurred to systems. This weight factor is obtained from the significance degree of data objects which is placed at the write set of transactions. The significance degree of a data object shows its importance in terms of the system owner. The more a data object is significant, the more security should be provided for it. A numerical value is considered for the degree of significance.

Depending on the application and the required security, different methods can be applied for obtaining the weight factor. In financial systems like banking ones which require a high security level, the weight factor can be the total of significance degrees. In other words, if W1, W2, W3 …, Wn are significance degrees of a transaction's write set, the weight factor W will be calculated as follows:

$$W = W1 + W2 + W3 + \ldots + Wn \qquad (1)$$

Therefore, the security level of every data object is taken into account to calculate the weight factor. Using this method, transactions which do suspicious activities on data objects with a high significance degree are identified rapidly and the system will be saved from incurring more damages. For example, in a banking system, a user who does suspicious activities on the accounts with high balances enjoying high importance is identified as a suspicious or malicious user rapidly, although using this technique may aggravate the false alarm problem. Of course, the false alarm problem may rise due to the incorrect selection of the anomaly threshold. By decreasing the anomaly threshold, the false positive alarm rate increases. By increasing the anomaly threshold, the false negative alarm rate increases. To cause a trade off between availability and integrity a mechanism must be introduced to control false alarms.

2.4. Our approach and contribution

In this paper, we present a false alarm controller as a solution to control false alarm problem, which is created due to considering significance degrees of data objects in the optimized attack isolation technique. In this approach, we use the idea of significance degrees of data objects to determine anomaly threshold adaptively. So, false alarm rate will be controlled proportional to the change of environmental conditions and a tradeoff will be done between availability and integrity.

3. The false alarm controller

Using an ID causes a conflict between "availability" and "integrity". By increasing anomaly threshold, "availability" decreases and "integrity" increases. Opposite condition will occur if anomaly threshold decreases. As explained in the previous section, neither attack isolation nor multiphase damage containment techniques presents a method to determine anomaly threshold adaptively based on the change of environmental conditions, which controls false alarm. Luenam et al. presented an adaptive controller to solve this problem [14]. This adaptive controller aims to improve the adaptation of an ITDB system so that the trustworthiness-to-cost range settles at an appropriate level.

This adaptive controller adjusts 4 main control parameters. The control parameters of ID are THm (malicious anomaly level threshold) and THs (suspicious anomaly level threshold). The control parameter of the Damage Container represents the damage leakage and is denoted as DL. If DL=0, the multiphase damage confinement technique will be enforced. If there is no restriction on DL, the one

phase damage confinement technique will be enforced. The control parameter of the mediator represents the transaction delay time and is denoted as DT. If DT=0, it means that the transactions are executed with the maximum speed. Otherwise, the transactions are executed with a slower speed.

Luenam and his colleagues have introduced 2 groups of metrics to adjust these control parameters. The first group is "trustworthiness metrics" which include:

1. Level of data integrity, (denoted as LI) which is indicated by the percentage of the damaged data objects (to all of the data objects).

2. Level of data availability from the perspective of containment (denoted as LDA), which is indicated by the percentage of the data objects contained by the Damage Container.

3. Level of data availability from the perspective of false alarms (denoted as LTA), which is indicated by the percentage of the harmless transactions that are mistakenly rejected due to false alarms to all of these transactions.

4. Level of data availability from the perspective of isolation (denoted as LIA), which is indicated by the percentage of the innocent transactions that are backed-out during merging processes.

The second group is used to measure the cost-effective of an ITDB system which include:

1. Level of system workload (denoted as LSW), which is indicated by the degree of workload decreasing.

2. Level of the system effectiveness (denoted as LSE), which is indicated by a group of effectiveness to cost ratios.

3. Level of the system attacks (denoted as LSA), which is indicated by how intense the attacks are.

These two groups of metrics are obtained through values of monitor parameters and are used as the input vector of adaptive controller.

As you notice, this method does not differentiate between data objects as far as "availability" and "integrity" requirements are concerned. A metric like LTA, which is affected by the false alarm monitoring parameters and is used to determine THm, examines the environmental conditions generally. In addition, the value of threshold control parameters (THm and THs) to detect malicious or suspicious transactions (users) in the whole system is the same. On the other hand, all the data objects in a database may need not the same level of "integrity" and "availability". For example, a data object may need a high level of "availability" with the "integrity" at the second degree of significance whereas the exact opposite condition may be acceptable for another data object.

We use the idea of significance degrees of data objects to achieve the following objectives:

1. Dividing the data objects in terms of "availability" and "integrity" requirements.
2. Applying separate control parameters for each group of the data objects.

By using this idea, false alarm will be controlled in accordance with the "availability" and "integrity" requirements of data objects. As mentioned in the earlier section, in the Optimized Attack Isolation technique, a significance degree is considered for each data object. These significance degrees are imposed through weight factor in the anomaly degree of transactions. Using this approach, we can adjust the ID effect in detecting suspicious and/or malicious transactions. In other words, depending on the value of weight factor, a transaction is identified suspicious (malicious) slower or faster. This is similar to changing the anomaly threshold which causes "availability" and "integrity" increment or decrement in opposite directions. Therefore, by changing the significance degrees of data objects in proportion to their "integrity" and "availability" requirements, the anomaly threshold can be

determined indirectly at any time in an adaptive manner. In other words, the anomaly threshold is determined for different transactions (users) with respect to the data that is accessed by them.

As there are usually a considerable number of data objects with similar security and "availability" requirements in a system, to increase the speed of calculations and to decrease complexities, data objects are classified based on some of their features .So, a significance degree is allocated to each class. We use a neural network for classifying data objects. Neural networks are non-linear data driven self-adaptive approach in a way that they can adjust themselves to the data without any explicit specification of a functional or distributional form for the underlying model. Neural networks are able to estimate the posterior probabilities, which provide the basis for establishing classification rules and performing statistical analysis [15, 16, 17, 18, 19, 27].

The significance degree is a new control parameter which denoted by Wi for each class. This control parameter is adjusted by an adaptive controller. The input monitoring parameters are determined for each class at the end of each control time interval. So, the definition of trustworthiness metrics for a certain class such as A is modified as follows:

1. LI is indicated by the percentage of the damaged data objects of class A (to all the data objects of class A).

2. LDA is indicated by the percentage of the data objects of class A which are contained by the Damage Container.

3. LTA is indicated by the percentage of the harmless transactions that are mistakenly rejected due to false alarms and have some data objects of class A in their write set to all such transactions which have some data objects of class A in their write set.

4. LIA is indicated by the percentage of the innocent transactions that are backed-out during merging processes and have some write set which is member of class A.

We will evaluate the proposed approach in more details in the following section and compare it with the approach presented by Luenam et al. to determine anomaly threshold.

4. Implementation and evaluation

We have implemented the proposed false alarm controller to determine the anomaly threshold. In this section, we compare it with the neuro-fuzzy adaptive controller (NFAC) which is offered by Luenam et al.

4.1. Implementation

To implement the proposed false alarm controller (FAC), we use the Neuro-fuzzy architecture, which have been presented by Luenam et al. This architecture is based on the Multiple Adaptive Network-based Fuzzy Inference System (MANFIS). In this architecture, a combination of Fuzzy logic and neural networks is used, as they cover shortcomings of each other. Fuzzy logic provides a robust mathematical framework for dealing with the imprecision and uncertainty of ITDB input. Also this technique has a high ability to adapt itself to dynamic environments. On the other hand, neural networks are capable of learning that will enable the fuzzy approach to automatically construct its rule structure and membership functions.

As shown in Figure 2, the fuzzy system is implemented as a five-layered neural network. To prepare the training data in a control time interval, we keep the environmental parameters such as the level of system workload (LSW), the level of system attacks (LSA), the mix of transaction type and the number of concurrent users at a constant level. Random values are created for the significance degree of each class. A significance degree is chosen as the significance degree of Class i (Wi) which

provides availability (LTAi + LIAi + LDAi) and integrity (LIi) required by Class i. The selected Wi and the values of the input parameters are used as the training data set. These steps are repeated for all possible levels of environmental parameters.

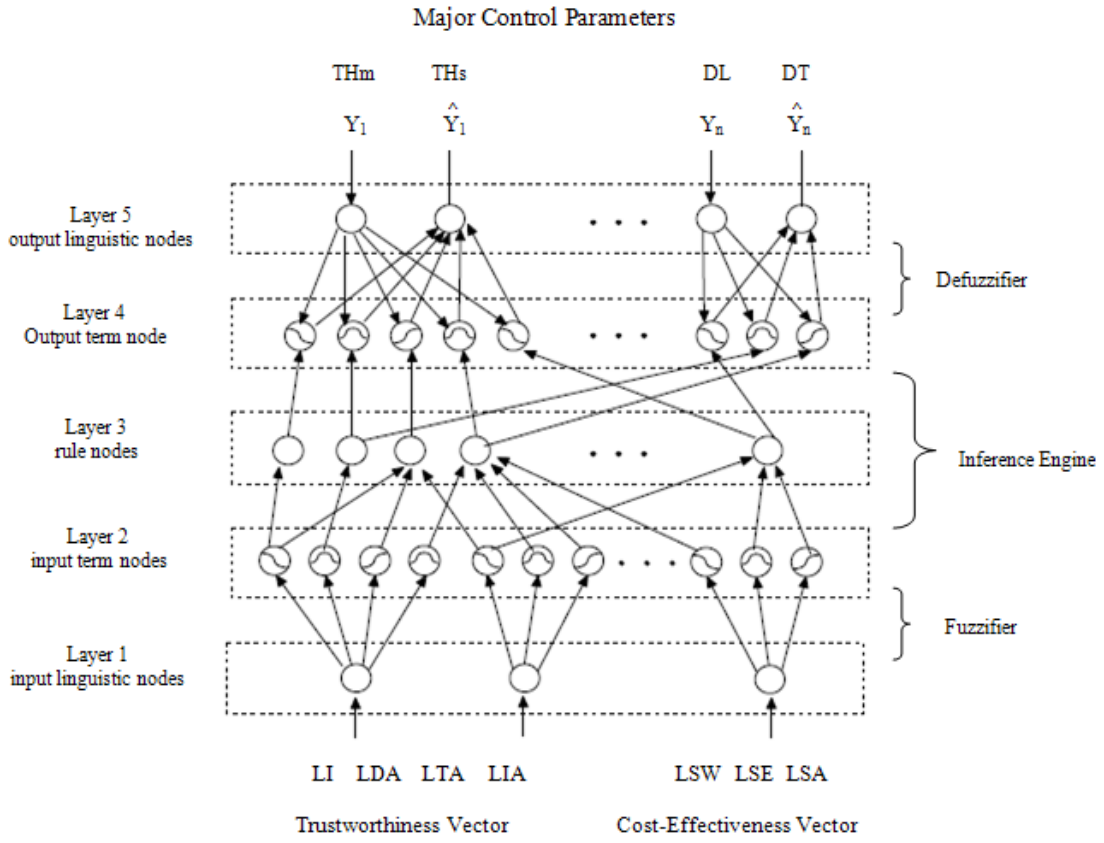


Figure2. The structure of neuro-fuzzy adaptive controller

FAC is executed beside ITDB architecture on a server. Therefore, we implement a prototype of an ITDB architecture, which uses the Optimized Attack Isolation technique (the time of waiting queue is considered as zero).

As explained in the earlier section, FAC aims at determining THm and THs through significance degrees of data objects without changing these thresholds generally. We also assume that, no reconfiguration operators will be performed to switch to the other intrusion tolerance technique during the experiments. Therefore, under this condition, the only output parameter of FAC is Wi that is related to data objects of class i. We also set the value of LDA metric, which is related to multiphase damage containment, on a constant value.

In the ITDB architecture, the mediator imposes the weight factor in the anomaly degree of transactions. However, prior to this action the significance degrees of data objects which exist in the write set of a transaction must be determined. As mentioned in the earlier section, data objects are classified based on some of their features and a significance degree is assigned to each class. We use a neural network with multilayer perceptron structure as well as the Levenberg-Marquadt back propagation training algorithm (which is a supervised algorithm) to determine the class of data objects. In order to prepare the training data, the system owner should determine the class of some data objects

according to their availability and integrity requirements. The inputs of neural network are the features of a data object and the desired output of neural network is a class which is proportional to the data object requirements. Both inputs and outputs are determined by the system owner in the training stage. This data set is used for the training of the neural network. After training, the neural network will be able to determine the class of each data objects. The Class Determiner component which is shown in figure 3 is responsible for determining a class for each record of the write log. The results are saved in the Classes Table. This component sends the features of each data object as inputs to a training neural network. The neural network output determines the class of data objects. After obtaining the significance degrees of data objects, WF is calculated according to a predetermined method. We use the average of significance degrees to calculate weight factor. The WF is imposed at the anomaly degree of transactions. Therefore, anomaly threshold is determined for each transaction (user) with respect to the significance degrees of its write set data objects.

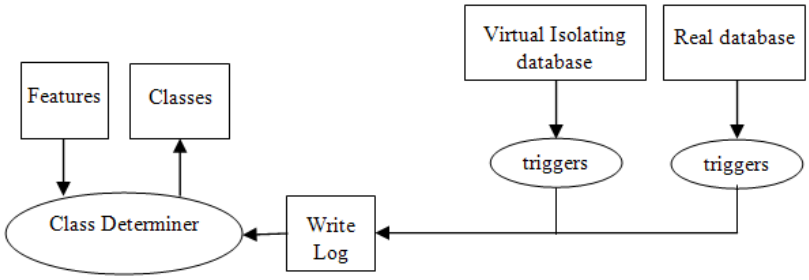


Figure3. Significance degree determiner

4.2. Experimental environment

The experimental environment includes one server and four clients which are connected by a 10/100 Mbps switch LAN. The specifications of these computers are shown in Table 1. The database server is Microsoft SQL Server 2008.

Table1. System specification

	Server	***Clients***
CPU	Pentium (R) D 3.0 GHz	Pentium 4 2.0 GHz
Memory	1 GB	512 MB
Storage	300 GB	80 GB
OS	Windows Server 2003	Windows XP Pro

4.3. Transaction generation

In our experiments, TPC_C benchmark [13] is used for generating transactions. TPC-C benchmark models the order processing operations of a wholesale supplier with some geographically distributed sales districts and associated warehouses. Five basic transactions that represent essential performance

characteristics of the application are defined by the benchmark. These transactions and percentage of their mixture are listed in Table 2.

Table2. TPC-C Transactions

Name	Characteristic	Percentage
New-Order	read-write, mid-weight	45%
Payment	read-write , light-weight	43%
Order-Status	read-only, mid-weight	4%
Delivery	read-write	4%
Stock-Level	read-only	4%

As Table 3 shows, four users transfer the transactions to the server simultaneously in the Experiments. We suppose that one of the users is malicious. The level of attack exceeds 20%. Therefore, more than 20% of the transactions are sent by the malicious user at any time. The system workload is considered to be between 50 to 100 transactions per second.

Table3. Environmental parameters

Parameter	value
The number of concurrent users	4
Level of workload (LSW)	$50 \leq LSW \leq 100$
Level of attacks (LSA)	LSA > 20%

Malicious transactions are simulated with abnormal inputs. The database will include 2.65 million records. It should be noted that the largest size of read/write set for a transaction is 33 reads as well as 33 writes while the smallest is 3 reads and 4 writes. The transaction generator is executed on clients. Transactions are submitted to server in batch-style (non-interactive) with a normal distribution.

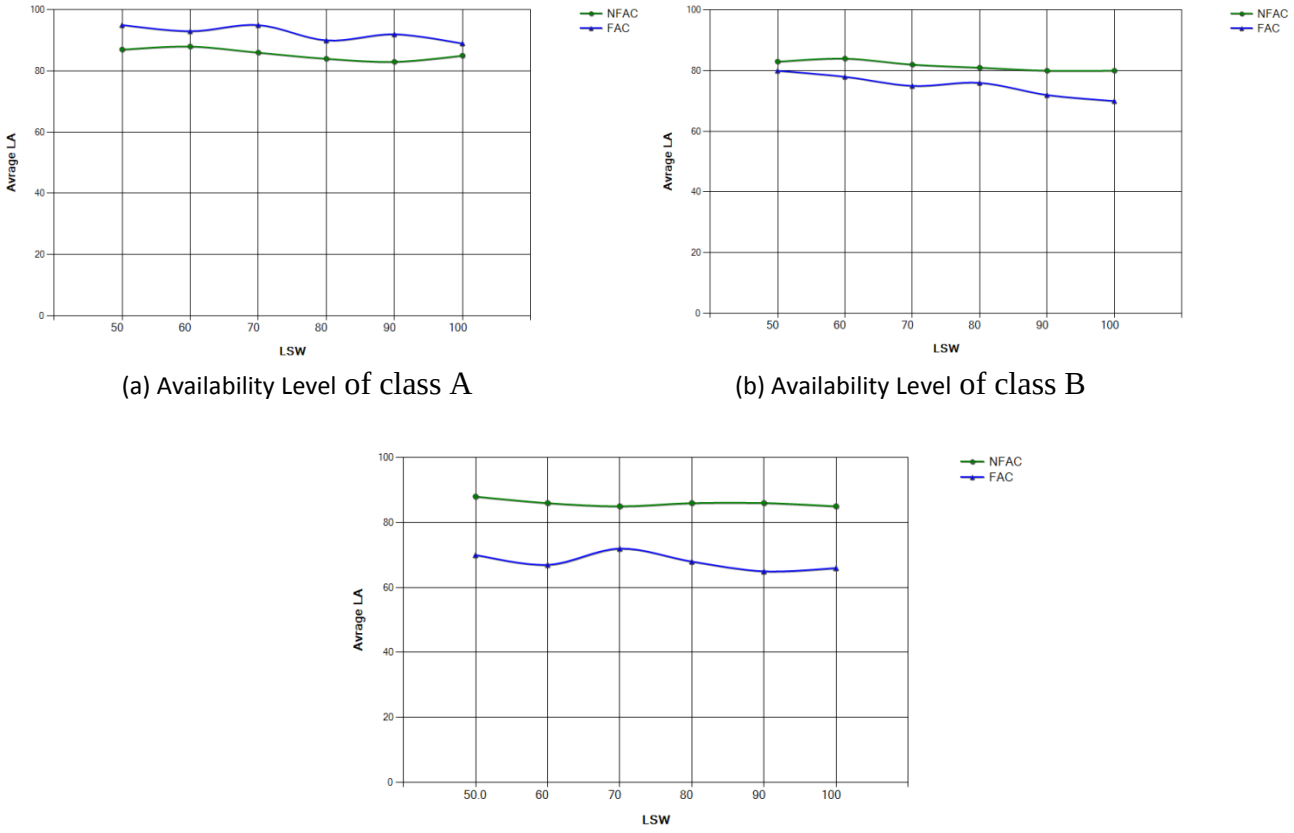
4.4. FAC vs. NFAC

Here, we study operation of FAC and NFAC controllers to determine THm and THs through some experiments. To compare these two controllers, we classify all the data objects available in the system into three classes. In this way, a FAC controller will be implemented for each class. Table 4 shows the level of "availability" and "integrity" required by each class.

Table4. The Classes of system

	Availability	Integrity
Class A	High (LA > 90)	Low (LDDA < 0.8)
Class B	Medium (LA > 60)	Medium (LDDB < 0.5)
Class C	Low (LA > 40)	High (LDDC < 0.2)

As mentioned earlier, FAC considers the "availability" and "integrity" requirements of the data objects to determine both THm and THs control parameters in proportion to the environmental conditions. Therefore, THm and THs will be different for the users who have access to the data objects with different significance degrees. However, NFAC determines these two control parameters for the whole system.



(a) Availability Level of class A

(b) Availability Level of class B

(c) Availability Level of class C

Figure4. Availability Level of data object classes

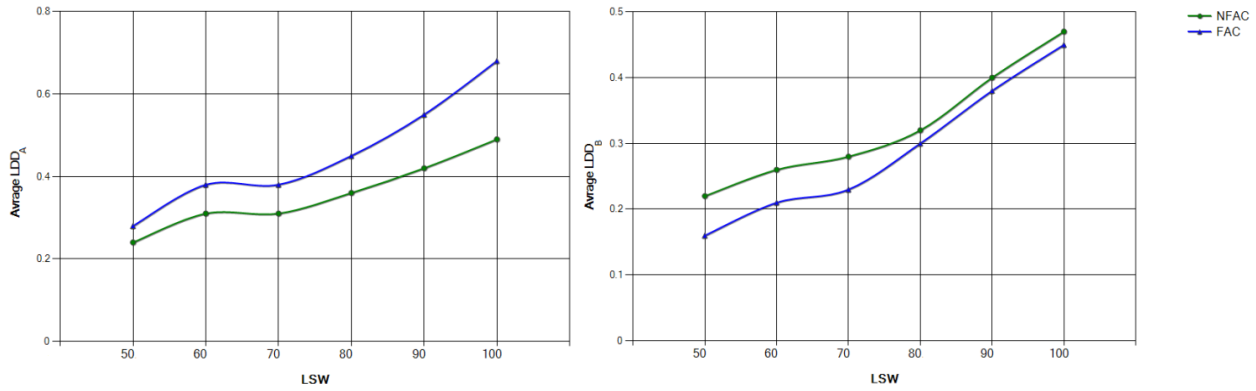
We do tow experiments to compare the effect of these two controllers on “availability” and “integrity” of data objects. In the first experiment, we apply FAC to determine the significance degree of classes. In the second experiment, which is carried out under similar conditions of the first experiment, NFAC is applied to determine THm and THs.

In these experiments, the average of LA (Level of Availability) is used to evaluate availability. LA is calculated for each class at the end of each control time interval as follows:

$$LAi = LTAi + LIAi + LDAi \qquad (2)$$

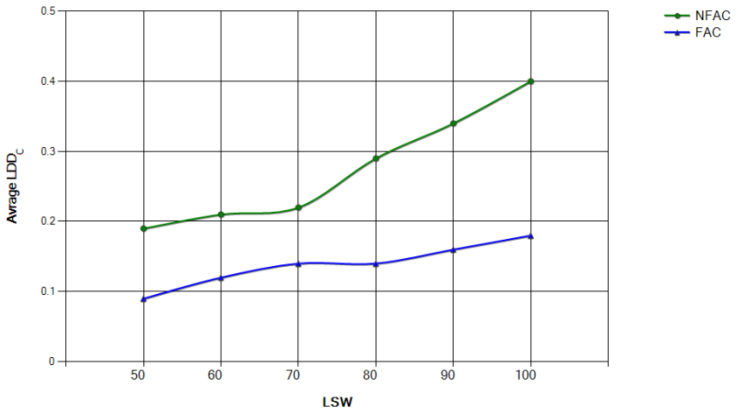
Since the multi-phase damage containment technique has not been implemented in these experiments, we ignore LDAi. Also, the average of LDDi (level of damaged data objects) which shows the damaged data objects percentage of each class is used to evaluate the integrity.

Figure 4 shows the Level of Availability achieved for each class in the two experiments. FAC provides a rate of availability with 7% more than NFAC for data objects of class A. However, on Classes B and C, FAC respectively provides a rate of availability with 6% and 18% less than NFAC.



(a) Integrity Level of class A

(b) Integrity Level of class B



(c) Integrity Level of class C

Figure5. Integrity Level of data object classes

Figure 5 shows the level of integrity achieved for each class in two experiments. FAC provides the data object, which are members of class A, with 19% less "integrity" than NFAC. However, on Classes B and C, FAC respectively provides the data objects, which are members of these classes, with 11% and 48% more "integrity" than NFAC.

The results of the experiments show that FAC provides the required availability and integrity of all Classes. However, NFAC does not favorably satisfy the "integrity" and "availability" requirements of Classes A and C.

Integrity is highly important for Class C. However, NFAC does not satisfy the integrity required by this Class. Instead, NFAC provides it with more availability than it requires. Also, NFAC does not provide Class A with its required availability. Instead, it provides this class with more integrity, which is of less importance for this Class. Both controllers provide the data objects of Class B with the required availability and integrity.

5. Conclusion

Intrusion Tolerant Database Systems apply novel techniques to react against attacks which succeed to pass the traditional protection systems. "Intrusion Detection" is one of these techniques. One of the major limitations of using ID in ITDB architecture is the false alarm. In this paper, we used the idea of significance degrees of data objects to control false alarms regarding the "availability" and "integrity" requirements of data objects. Using this method, the anomaly threshold for each transaction (user) will be determined with respect to the significance degrees of its write set data objects.

In addition to significance degrees of data objects, for future studies, we can use new parameters such as user trust degree to control false alarm more effectively. Applying this parameter, each user has a trust degree with respect to his/her own characteristics such as the way he/she is connected to system and the history of attacks that has been done by this user. This trust degree will be imposed to the anomaly degree of the transactions of this user.

11. References

[1] C. Kruegel and G. Vigna, Anomaly detection of web-based attacks, In CCS'03, pp. 251–261, Washington, USA,October 27-31 2003.

[2] T. Ryutov, C. Neuman, D. Kim, and L. Zhou, Integrated access control and intrusion detection for web servers, IEEE Transactions on Parallel and Distributed Systems, 14(9):814–850, Sep 2003.

[3] P. Liu, Architectures for intrusion tolerant database systems, Proc. 2002 Annual Computer Security Applications Conference, Dec 2002, pp. 311-320.

[4] P. Luenam and P. Liu, The design of an adaptive intrusion tolerant database system, InProc. IEEE Workshop on Intrusion Tolerant Systems, 2002.

[5] P. Liu, J. Jing, P. Luenam, Y. Wang, L. Li, and S. Ingsriswang, The Design and Implementation of a Self-Healing Database System, Journal of Intelligent Information Systems, 23 (3), pp. 247-269, 2004.

[6] T. Lunt, A survey of intrusion detection techniques, Computers & Security, 12(4):405–18, Jun. 1993.

[7] F. S. Rietta, Application layer intrusion detection for sql injection, In ACM-SE 44: Proc. 44th annual Southeast regional conference, pp. 531–536. ACM Press, New York, NY, USA, 2006.

[8] C. Y. Chung, M. Gertz, and K. Levitt. Demids, A misuse detection system for database systems, In 14th IFIPWG11.3 Working Conference on Database and Application Security, 2000.

[9] S. Stolfo, D. Fan, andW. Lee, Credit card fraud detection using meta-learning: Issues and initial results, In AAAI Workshop on AI Approaches to Fraud Detection and Risk Management, 1997.

[10] P. Liu, H. Wang, and L. Li, Real-time data attack isolation for commercial database applications, Elsevier Journal of Network and Computer Applications, 29(4):294–320, 2006.

[11] P. Liu and S. Jajodia, Multi-phase damage confinement in database systems for intrusion tolerance, In Proc. 14th IEEE Computer Security Foundations Workshop, Nova Scotia, Canada, June 2001.

[12] Z.Falahiazar, M.Rohani, The architecture of an intrusion tolerant database system, Proc. 2010 International Conference onEducational and Information Technology, September 2010.

[13] TPC BenchmarkTM C. http://www.tpc.org/tpcc/.

[14] P. Luenam , P. Liu , A. F. Norcio, Adaptive Intrusion Tolerant Database Systems, VDM VERLAG DR. MULLER, Germany, December 2008.

[15] Freeman J, Skapura D. Neural Networks: Algorithms, applications and programming techniques. Addison-Wesley, Reading, MA, 1991

[16] HAYKIN S. Neural Networks: A Comprehensive Foundation, second edition. Prentice Hall 1999.

Chapter 4 Multilayer Perceptrons, pp. 156−255

[17] Rumelhart DE, Hinton GE, Williams RJ. Learning internal representations by error propagation. In: Rumelhart DE, McClelland JL (eds). Parallel Distributed Processing, Vol. I. MIT Press, Cambridge,

[18] Madan M. Gupta, Liang Jin, and Noriyasu Homma, Static and Dynamic Neural Networks From Fundamentals to Advanced Theory, chapter 4, 2003 John Wiley & Sons.

[19] Oliver Nelles, Nonlinear System Identification From Classical Approaches to Neural Networks and Fuzzy Models, chapter 11, Springer, Springer, Verlag Berlin Heidelberg 2001.

[20] Martin T. Hagan, HowllId B. Demuth,Neural Network design,chapter 11, 1996 , PWS Publishing Company.

[21] P. Ammann, S. Jajodia, and P. Liu. Recovery from malicious transactions. IEEE Transaction on Knowledge and Data Engineering, 14(5):1167–1185, 2002.

[22] T. Chiueh and D. Pilania. Design, implementation, and evaluation of an intrusion resilient database system. In Proc. International Conference on Data Engineering, pages 1024–1035, April 2005.

[23] R. Sobhan and B. Panda. Reorganization of the database log for information warfare data recovery. In Proceedings of the fifteenth annual working conference on Database and application security, pages 121–134, Niagara, Ontario, Canada, July 15-18 2001.

[24] M. Yu, P. Liu, and W. Zang. Self-healing workflow systems under attacks. In The 24th International Conference on Distributed Computing Systems(ICDCS'04), pages 418–425, 2004.

[25] Y.-W. Huang, S.-K. Huang, and C.-H. Tsai. Web application. In WWW 2003, pages 148–159, Budapest, Hungary, 2003. ACM, ACM.

[26] Z. Falahiazar, M. Rohani, L. Falahiazar, and M. Teshnelab. Optimizing An Intrusion Tolerant Database System Using Neural Network . In International Journal of Database Theory and Application, Vol. 5, No. 2, June, 2012.

[27] Guoqiang Peter Zhang. Neural Networks for Classification: A Survey. In IEEE Transactions on systems, MAN, and Cybernetics—Part C: applications and reviews, Vol. 30, No. 4, November 2000.

[28] Mohammad M.Javidi, Mina Sohrabi, and Marjan Kuchaki Rafsanjani, Intrusion Detection in Database Systems. In Communication and Networking, Communications in Computer and Information Science, Volume 120. Springer Berlin Heidelberg, 2010, p. 93

[29] M. Hassanzadeh and G. Ardeshir, Optimal Membership Function for Creating Fuzzy Classifiers Ensemble, Journal of mathematics and computerscience, 12 (2014) pages 73–84

[30] Shiva Zahedian, Aspect-Oriented Software Maintainability Assessment Using Adaptive Neuro Fuzzy Inference System (ANFIS), Journal of mathematics and computerscience, 12 (2014) pages 243–252

[31] Neda Noori, Leila Boti, Ebrahim Nowzarpoor Shami, Surveying Different Aspects of Anomaly Detection and Its Applications, Journal of mathematics and computerscience, Vol. 4 No.2 (2012) pages 129-138