Optimizing an Intrusion Tolerant Database System Using Neural Network

Zeinab Falahiazar[1], Mohsen Rohani[1], Leila Falahiazar[2] and Mohammad Teshnelab[2]

[1,2]Department of computer engineering
[1]Islamic Azad University South Tehran Branch, Tehran, Iran
[2]Islamic Azad University Science and Research Branch, Tehran, Iran
zfalahiazar@gmail.com, m.rohani@niopdc.ir, leilalfa@gmail.com,
Teshnehlab@eetd.kntu.ac.ir

Abstract

Traditional database security mechanisms focus on either protection or prevention. However, these mechanisms have not any strategy in the presence of successful attacks. To solve this problem, the Intrusion Tolerant Database System (ITDB) was introduced. ITDB uses the new generation of database security mechanisms to guarantee specified levels of data availability, integrity and confidentiality in the presence of successful attacks. These mechanisms include Attack Isolation and Multiphase Damage Confinement. In this paper, we will present a practical model to utilize the combination of intrusion tolerance techniques for managing the ITDB architecture. Using this practical model, we will be able to secure the system's required integrity and availability levels considering the changes in the environment. We will also introduce an intelligent method for determining the significance degrees of data objects in the optimized attack isolation technique.

Keywords**:** *Intrusion Tolerance; Database Security; Intrusion Detection; Attack Isolation; Damage Confinement; Neural Network*

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 to heal itself under malicious transaction (but authorized) attacks [21, 22, 23, 24] aiming to deliver a continuous reliable service. Intrusion Detector (ID) is regarded as one of the main ITDB parts. The task of ID [6, 7, 8,9] is to identify malicious transactions. ID is much slower than transaction processing in general. 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] 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.

In the previous work [12], certain approaches were presented to optimize the attack isolation technique. Then an ITDB architecture based on the proposed technique was evaluated. This piece of research intends to introduce a practical and effective method to determine the significance degrees of data objects in the optimized attack isolation technique. A model will also be investigated to manage the ITDB architecture to ensure the system's required security and performance considering the changes in the environment.

The rest of the paper is organized as followings: In this section, a brief description of two main techniques used in ITDB architectures will be presented. The optimized attack isolation technique will be briefly explained in section 2 where a practical method for determining the significance degrees of data objects is introduced. In section 3, we will evaluate the effects of some environmental parameters on the system's performance and will present a combinational model of intrusion tolerance techniques to manage the ITDB architecture. Finally, the conclusion will be presented in section 4.

1.1. Multiphase Damage Confinement

The one phase damage confinement technique consists of one damage confinement phase. In this phase, the damage assessor does not give the permission to transactions to read the data objects which are identified as damaged ones. The only problem with this technique is that if the damaged data objects (as a result of damage spreading) are not yet to be identified, a lot of transactions may read them and as a result, the damage may spread. To solve this problem, the multi phase damage confinement technique has been presented.

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 some 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.

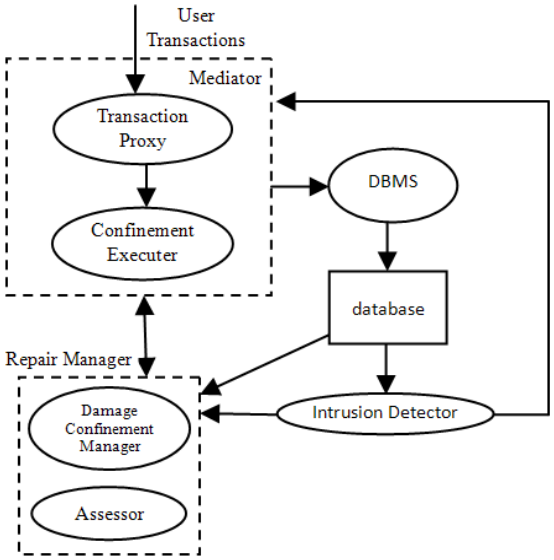

Figure 1. Architecture of Multiphase Damage Confinement

To optimize this technique we can preserve an on-the-fly dependency graph based on the history log for each suspicious transaction. Therefore, more data objects with less delay will be unconfined. Consequently, more availability will be obtained. Nevertheless, the processing overhead should be evaluated to assign an appropriate time for preserving such on-the-fly graph for suspicious transactions.

In Figure 1, main components of the architecture for an intrusion tolerant database system which has used this technique are shown.

In this architecture, ID is responsible for reporting malicious transactions. Mediator is responsible for the initial confinement phase and the confinement enforcement. The Damage Confinement Manager is responsible for the first and the second unconfining phases. Damage assessor is responsible for the third unconfining phase.

1.2.Attack Isolation

In attack isolation technique suspicious activities are isolated, before an intrusion is reported as unquestionable. In this technique isolation is done based on users. 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.

In this technique 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 it is proven that an isolated user has been innocent, updates of this user should be merged back into the real database.

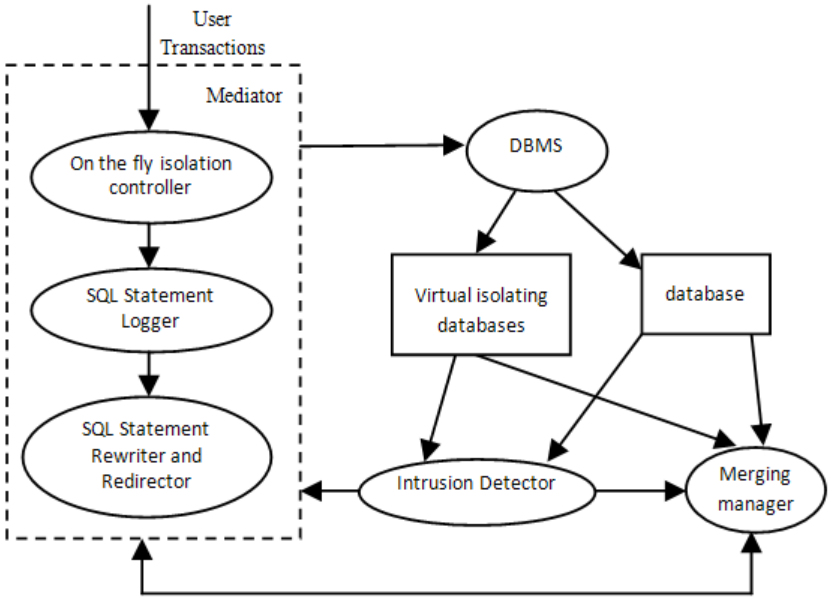

Figure 2. DAIS Architecture

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 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 means that there are no inconsistencies. Otherwise, if the graph has cycles, all the cycles are 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.

In Figure 2, the architecture of a data attack isolation system (DAIS) is shown. In this architecture, ID is responsible for reporting suspicious users that should be isolated. The Mediator, which includes three components as follows, 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. Optimized ITDB Architecture

We have presented some approaches to solve DAIS problems and to improve the integrity and security of database in optimized ITDB architecture [12]. In this section, we will describe the optimized ITDB architecture which employs the optimized attack isolation technique. We will also present a practical and effective method to determine the significance degrees of data objects used in optimized attack isolation technique. The optimized ITDB architecture has three distinct characteristics as follows.

2.1. Isolation Time

As shown in Figure 3, 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, we will have a more secure DAIS.

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.

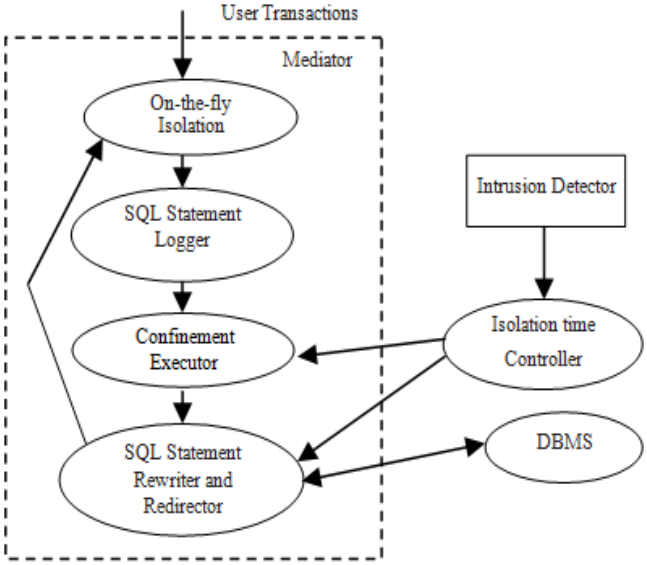

Figure 3. Components of Mediator

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 is 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 informs the Confinement Executer and then the multiphase damage containment technique is used.

2.2. Significance of Data Object

In DAIS, 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. We are now going to present an applicable and intelligent method to determine the significance degree of a data object.

2.2.1. Determining the Significance Degree of a Data Object: Determining the significance degree of every data object when enters the system is not feasible for system owners. Using an artificial neural network is a fast and applicable solution to determine the significance degrees of data objects. Neural networks, with their remarkable ability to derive meaning from complicated or imprecise data, can be used to extract patterns and detect trends that are too complex to be noticed by either humans or other computer techniques [15, 16, 17, 18, 19, 20].

In terms of training algorithms, neural networks are usually divided into two main categories: supervised networks and unsupervised networks. The unsupervised training method is aimed at building some representations from the input data. This is done by organizing the input data properly. Such data can later be used in decision making. In supervised methods of learning, neural network learns from examples. The training set consists of a set of sample inputs and the desired outputs corresponding to those inputs. Successfully trained neural network can then be used to find most suitable output for any valid input. The goal of supervised learning is to find a function f, given a set of points of the form (x, f(x)).

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 significance degrees of data objects. The Levenberg-Marquardt optimization technique has fast convergence to the final answer.

To prepare the training data in this method, the system owner should determine the significance degree of some data objects from every type that should be kept safe and secure. For a better understanding of the matter, we describe this technique at the record level (where each data object denotes a record and each type of data object denotes a table) on the TPC-C benchmark [13].

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. One of the transactions is the payment transaction. This transaction updates the customer balance and transmits the payment details to the warehouse sales list as well as the relevant district.

The significance degrees of data objects which exist in the write set of a transaction must be determined. One of the existing data objects in the write set of the payment transaction is of the customer type and two other data objects are of warehouse and district types. We assume that in terms of the system owner, the significance degree of a customer type object depends on its balance (C_BALANCE) and its number of payments (C_PAYMENT_CNT). In a table called “Features”, we keep the features of a data object type which is used to determine the significance degree. In Table 1, the features of a data object from customer type which is inserted into the feature table, are shown.

Table 1. Features Table

Obj_Type	Feature
Customer	C_BALANCE
Customer	C_PAYMENT_CNT

In the last example, the significance degree of a data object was related to the feature of only one type of object. Depending on the application and the importance level of the data object in terms of the system owner, we can extract different strategies.

So, the input of the neural network is the features of data objects and the desired output of the neural network is the significance degree corresponding to input data objects which are determined by the system owner. This set of numbers is used for training the neural network.

After the training, the neural network will be able to determine the significance degrees of new data objects. Thus, a neural network should be trained for each type of object.

To determine the significance degree of a data object, the type of that object should be identified first. Depending on the type of object, we may have two different situations:

- If the object type has a significance degree (a neural network is trained for this type), a "select" statement will be executed to retrieve the data object features from "Features" table. Values of these features are sent as input parameters to the neural network which is trained for this type of object.
- Otherwise, we take a number as the significance degree in the way that this number should not have any effects on the weight factor.

The Significance Degree Determiner component is shown in Figure 4. This component determines a significance degree for each record of write log table in the way described above. The results is saved in the Significance Degree Table.

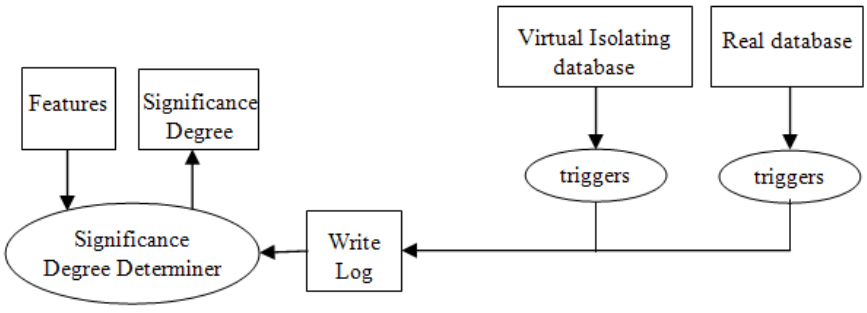

Figure 4. Significance Degree Determiner

2.2.2. Calculating the Weight Factor: Depending on the application and the required security, different methods can be applied for obtaining the weight factor. In financial systems like a banking system, which requires a high security level, the weight factor can be the total of significance degrees. In other words, if W_1, W_2, W_3,…,W_n are significance degrees of a transaction's write set, the weight factor W is calculated as follows:

$$W= W_1+ W_2+ W_3 +\ldots+W_n. \quad (1)$$

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.

To implement this method the significance degree table is used for the calculation of the weight factor of each suspicious transaction sent by ID.

2.2.3. Evaluation: To start the evaluation process, we implement a prototype of optimized ITDB architecture. For more information regarding the implementation process refer to [12]. The experimental environment includes one server and two clients which are connected by a 10/100 Mbps switch LAN. The specifications of these computers are shown in Table 2.

The database server is Microsoft SQL Server 2008. The transaction generator is executed on clients. All parts of the optimized architecture are executed on the server. In these experiments, TPC_C benchmark is used for generating transactions. 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.

Table 2. System Specification

System specification	***Server***	***Client 1***	***Client 2***
CPU	Pentium (R) D 3.0 GHz	Pentium 4 2.0 GHz	Pentium 4 2.0 GHz
Memory	1 GB	512 MB	512 MB
Storage	300 GB	80 GB	80 GB
OS	Windows Server 2003	Windows XP Pro	Windows XP Pro

For the measurement of the performance loss, we measure the average response time of transactions (per 1,000 transactions). Transactions are submitted to server in batch-style (non-interactive) with a normal distribution. The same set of transactions is used at all experiments. Response time of each transaction is the interval between the timestamp taken before the last

character of the required input data is entered by the emulated user and the timestamp taken after the transaction's commit. It is obvious that this time includes the time spent for the controls in ITDB.

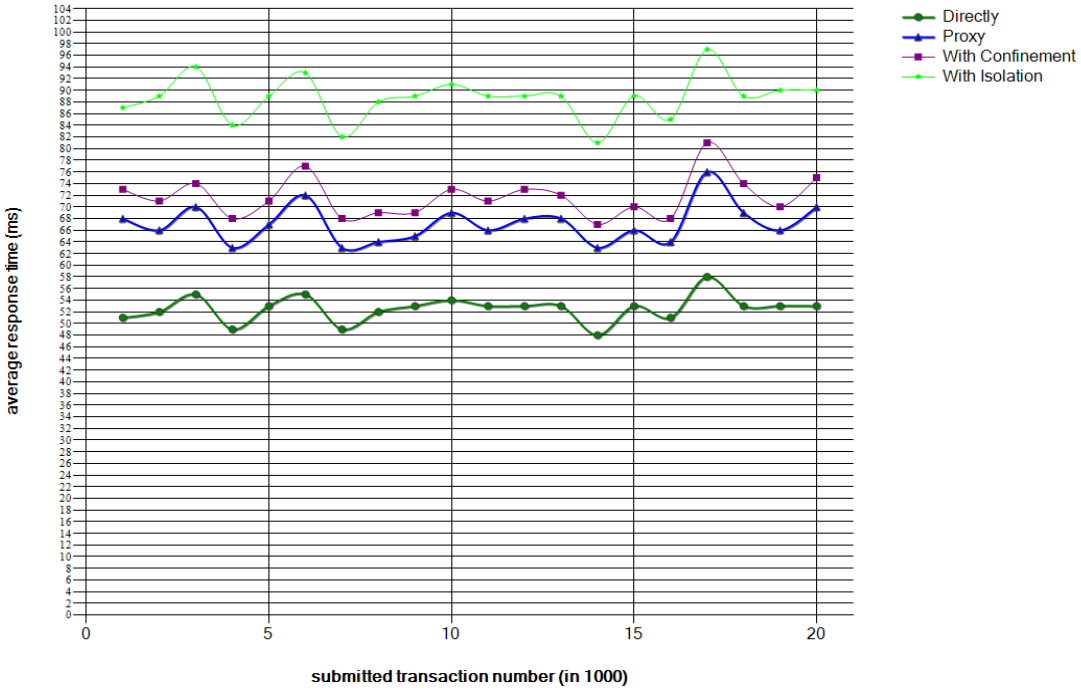

Figure 5. Average Response Time (waiting time of each transaction before sending is 0 to 5 milliseconds)

In Figure 5, the effect of these controls is displayed on the transactions' response time within four experiments. In these experiments, the transactions are sent from two different clients. We consider 0 to 5ms as the waiting time before each transaction is sent.

We use the randn() function in MATLAB simulator software to make a normal waiting time. This function can make random numbers from a normal distribution with mean 0 and standard deviation 1.

Transactions, sent by a trustworthy user, are submitted directly to the server for execution without any control in the first experiment. In the second experiment, the read and write set of transactions, sent by a trustworthy user, are extracted by the proxy subsystem. The results of this experiment show that this subsystem will reduce performance up to 37.2%.

Transactions submitted by a trustworthy user beside a proxy subsystem are scanned by the Confinement Executer in the third experiment. In this experiment, the Confinement Executer scans the write set of transactions. Since there is not any suspicious version of objects in this experiment, no transaction will be rejected. The results of this experiment show that the Confinement Executer reduces performance up to 7.3%.

In the fourth experiment transactions are submitted by a suspicious user. In this experiment, in addition to controls related to the isolation of suspicious transactions, controls which were enforced at the previous experiment are applied as well. In this experiment, controls, related to the isolation phase, reduce performance up to 21.8%.

The results of experiments show that the optimized ITDB, compared with DAIS, has a negligible performance loss (7%). This is due to the existence of confinement executer.

In the previous set of experiments, the significance degree of transaction's write set was not determined. In coming set the Significance Degree Determiner component is enforced to evaluate the system performance. For this purpose, a neural network is trained for the type of customer which was described in previous section. Significance Degree Determiner does not cause a direct affect on the response time of the transactions. However, this unit will increase the system workload which may cause the performance loss.

The experiments are repeated with previous conditions, the only difference is that the significance degree of data objects which will be kept in the significance degree table, are determined. In Figure 6, the effect of controls under the new circumstances is displayed on the transactions' response time.

In the first experiment, the write log table is empty. So, the Significance Degree Determiner unit does nothing and the average response time does not change. In the second and third experiments, the Significance Degree Determiner unit reduces the system performance by about 1% .In the fourth experiment this unit reduces the system performance by about 2%.

The result of experiments shows that the performance loss which is created by the Significance Degree Determiner unit, is negligible (about 1%).

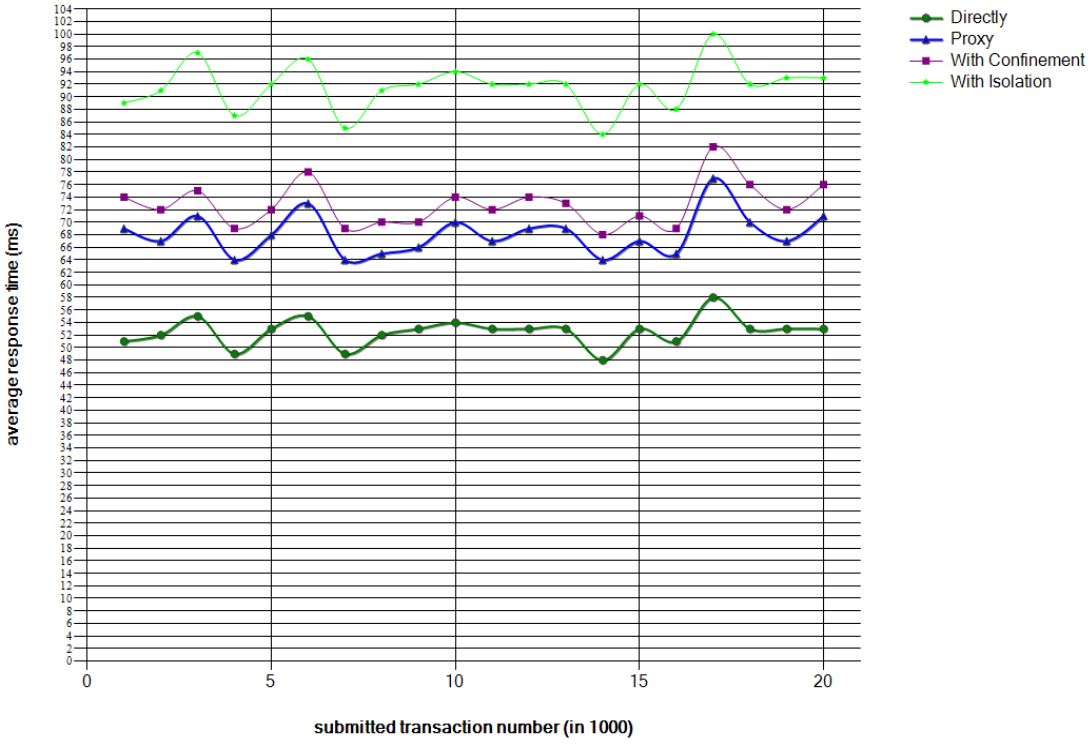

Figure 6. Average Response Time (with enforcing significance degree determiner)

2.3. Waiting Queue

In DAIS, during the time of isolation, inconsistency may be appeared. Some of these inconsistencies are not resolvable. For example, in a bank application, a suspicious transaction may withdraw fund from an account and a transaction of an authorized user also may withdraw fund from same account in the long run of isolation time. If suspicious user is

identified as innocent, data versions of the suspicious user are merged into the real database. These operations may cause the account balance being negative. At such a case, if the application did not accept negative balances, the integrity of database will be compromised. In fact, with simultaneous updating of a data object on numerous versions, integrity rules may be violated.

To solve this problem, we prevent updates with suspicious versions. This will have some effects:

1. As every data object can only have one suspicious version at any given time, we will lose a large amount of data availability in our system.

2. Using this method, there will be fewer inconsistencies during the isolation time. Therefore, while the merging process is done, fewer transactions will be backed out.

For making a tradeoff between integrity and availability a waiting queue will be considered for each data object which has suspicious version. The waiting time initially is equal to the isolation time. As the waiting time gets shorter, the availability level of the data objects with more referrals increases. Availability is considered as a very important factor for such objects. These data objects can have several suspicious versions at any given time. Depending on the application, we can use different strategies to decrease the waiting time. Under such circumstances, the possibility of outbreak of insolvable inconsistencies will be increased and consequently, the integrity is reduced.

As shown in the Figure 3, the confinement executer scans the write set of transactions. If the write set of a transaction contains any suspicious object version, the confinement executer will put the transaction in the waiting queue and will prevent it from being executed. These waiting transactions can be restarted after terminating of waiting time.

3. Managing the ITDB Architecture

Although the preventive controls of optimized ITDB architecture provide more security and integrity, they can degrade the system's performance. All architectures of ITDB, due to the controls which conduct on execution of transaction, will reduce performance to some extent. The extent of performance loss in our systems is influenced by changes in the environment. In this section, we are going to evaluate the performance overhead of the optimized ITDB architecture under different circumstances and present a combinational model of intrusion tolerance techniques for managing the ITDB architecture.

3.1. Evaluation of Environment Parameters

In the experiments shown in Figure 5 the waiting time of each transaction being sent to the server is very short. So, results of experiments display the system performance under the maximum workload. For evaluating the system performance under different circumstances (different workloads), the waiting time of each transaction before sending is changed and experiments are repeated. This time, we consider 0 to 50ms as the waiting time for each transaction.

In Figure 7, the effect of controls under the new circumstances is displayed on the transactions' response time.

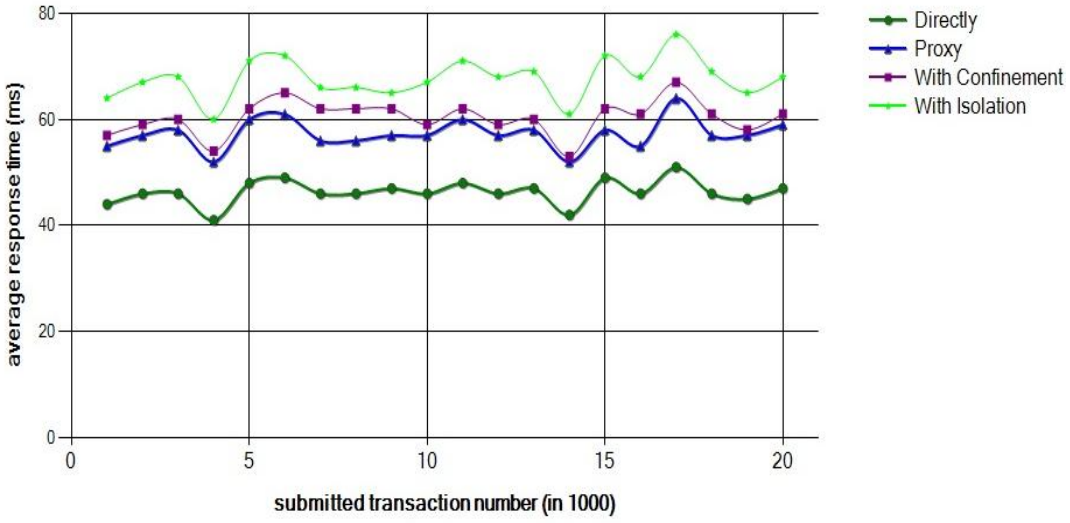

Figure 7. Average Response Time (waiting time of each transaction before sending is 0 to 50 milliseconds)

In the first experiment, transactions are submitted directly to the server for execution without any control, the average response time is reduced by 13% compared to the first state. In the second experiment, the proxy subsystem reduces the system performance by 30% which is 7% less than the first state. In the third experiment, however, the confinement executer unit reduces the system performance by 5.1%. In this experiment, the performance loss is 2% less than the first state. In the fourth experiment, the isolation controls reduce the system performance by 17.3%. In this experiment, the performance loss is 4.5% less than that in the first state.

As is inferred from the result of experiments, with decreasing the system workload, the performance loss which is the result of the security subsystem controls, is decreased. Indeed, concurrent executing of transactions is reduced by decreasing the system workload and as a result, the transactions response time will be reduced. However, this is not a linear relationship. The performance loss decrease will eventually stop at the point where there will be a minimum possibility of any confliction in the execution of transactions.

In these experiments, we studied the effect of system workload on the extent of performance loss in our system. Another factor which could impress the system performance is the number of attacks. As the number of attacks increases, the security subsystems make more restrictions to prevent the attacks and to repair the damages caused by them. As a result, the system performance will be reduced. Therefore, a proper security subsystem should be used to provide the system's required security and performance to make a tradeoff between the integrity and the availability of system.

3.2. Adaptive Controller

The architecture of an ITDB needs a controller to provide the required integrity and security for the system according to the changes in the environment. An adaptive ITDB architecture is presented by Liu and his colleagues for this purpose, which is denoted as AITDB [4]. An adaptive controller is used in this architecture. This controller uses a number

of monitoring parameters to monitor the environmental circumstances. These parameters represent the integrity and availability levels, number of attacks and the workload of the system.

This adaptive controller implements a number of reconfiguration (vector) according to the values of the monitoring parameters. In fact, each reconfiguration is a set of control parameters (and their values). This set is related to one of the ITDB components. The control parameters determine the behavior of ITDB components for the next adaptation interval. In other words, the adaptive controller scans the monitoring parameters after a certain amount of time (the adaptation interval) and set the values of the control parameters according to the environmental circumstances.

The control parameters which are set by the adaptive controller are related to the ID, the damage container and the mediator.

The major control parameters for the ID are either TH_m (malicious anomaly level threshold) or TH_s (suspicious anomaly level threshold). The control parameter of the Damage Container represents the damage leakage and is denoted as DL. If DL=0, the multi phase 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.

In this adaptive controller, however, there is no control parameter to switch between the attack isolation technique and the multi phase damage containment technique. It has not mentioned clearly in the aforementioned research that how we can use both techniques at the same time. In this part of the paper, we are going to present a practical model to utilize the combination of intrusion tolerance techniques.

3.2.1. Combinational Model: In Figure 8 a practical model for using the combination of intrusion tolerance techniques is shown. According to this model, switching from one technique to another is possible by determining proper values for the control parameters at any given time (this job is done by the adaptive controller). It is also possible to use several techniques simultaneously to provide the system's required integrity and availability levels.

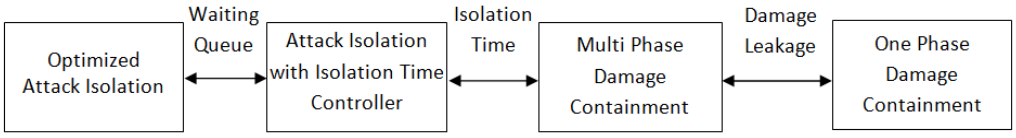

Figure 8. A Combinational Model of Intrusion Tolerance Techniques for Managing the ITDB Architecture

The control parameters which are used to switch between the intrusion tolerance techniques are shown in this model. As we move toward the left-hand side techniques, the availability level decreases and the integrity level increases. For a better understanding of this model, we are going to compare the integrity and availability levels of different intrusion tolerance techniques.

As we explained earlier, the multi phase damage confinement technique guarantees that no damage will be leaked out after identifying a malicious transaction. Therefore, the integrity

level of this technique is higher than the one phase damage confinement technique. However, since some of the data objects may mistakenly be confined in this technique, the availability level of this technique is lower than the one phase damage confinement technique. The control parameter used to switch from the multi phase damage confinement technique to the one phase damage confinement technique is the amount of damage leakage. If the integrity has the highest priority in our system, no damage should be leaked out. Thus, the value of DL is set to 0 to enforce the multi phase damage confinement technique.

In the attack isolation technique, the effects of suspicious transactions on a virtual database are isolated. For this reason, this technique provides an appropriate level of integrity. A data object can have several suspicious versions at the same time in this technique. Therefore, this technique provides an appropriate degree of availability too. Since there are some preventive actions taken in this technique, the attack repair costs will decrease. For this reason, the attack isolation technique has the priority over the multi phase damage confinement technique. However, there is no limitation for the isolation time in the attack isolation technique and this is count as a security weakness point of this technique. If the suspicious users list is attacked, the isolation time may get longer and as a result, the availability level may decrease and the merging cost may increase. To eliminate this security weakness point, we need to allocate a primary isolation time (as described in the previous section) to each user. The attack isolation technique with isolation time controller and the optimized attack isolation technique, partially switch to the multi phase damage confinement technique for the users whose isolation time is finished but their innocence is not proved yet. Partially switching in this manner is done by the isolation time controller. In other words, a combination of both techniques is used by ITDB architecture in such a situation.

If there is a high growth in the number of users who are switched to the multi phase damage confinement technique, switching totally to the multi phase damage confinement technique will be a good idea. In case there is an increase in the number of suspicious users whom isolation time is finished, we can conclude that the users' session time has exceeded its average value. If we use the attack isolation technique in such situation, the availability level will be decreased and the merging costs will be increased. For this reason, the multi phase damage confinement technique seems to be more appropriate for use. In this case, the average session time of users is utilized as a monitoring parameter. We use the Isolation Time (IT) as a control parameter to switch to the multi phase damage confinement technique. If the value of this parameter is equal to zero, the isolation time controller will set the value of users' primary isolation time to zero and as a result, the system will totally switch to the multi phase damage confinement technique.

In the attack isolation technique, if a data object is updated in several different versions simultaneously, the integrity rules may be violated. In the optimized attack isolation technique this problem is eliminated by using a waiting queue for the data objects with suspicious versions. As mentioned earlier, for making a tradeoff between the integrity and the availability levels of system, we can adopt different strategies to adjust the time of waiting queue. Using such strategies, the waiting queue of data objects with more referrals will be defused in a shorter amount of time. In other words, these data objects can have several suspicious versions at the same time. Using this method, the system can partially switch from the optimized attack isolation technique to the attack isolation technique. We use a control parameter called Waiting Queue (WQ) for performing such a total switch. If the integrity level of system has a high priority, we then will set the WQ to 1 and switch to the optimized attack isolation technique. In case the availability level of system is below the required one, we can set the WQ to 0 and switch to the attack isolation technique. When the value of the WQ parameter is zero, the Confinement Executer will permit all the transactions that must

have been placed in the waiting queue to execute. As a result, the system will switch to the attack isolation technique. Otherwise, the Confinement Executer will not give the execution permission to the transactions which intend to update the data objects with suspicious versions. The Confinement Executer will place such transactions in the waiting queue and as a result, the system will switch to the optimized attack isolation technique.

3.2.2. Implementation and evaluation: To reconfigure the ITDB architecture dynamically, the adaptive controller must find the best configuration vector (which consists of control parameters) according to the monitoring parameters. Since the adaptation space of the ITDB architecture consists an exponential numbers of configurations, it is a difficult job for the adaptive controller to find such vector. In other words, if the configuration vector is [TH_m,TH_s, DL, DT,IT,WQ], its relevant adaptation space will be d(TH_m×TH_s× DL× DT×IT×WQ) which is actually huge. To solve this problem, the heuristic adaptation algorithms are used. The adaptive controller should select the proper heuristic adaptation algorithm according to the environmental circumstances. Luenam and his colleagues have presented a rule-based mechanism as well as a neuro-fuzzy technique for implementation of this adaptive controller [14]. This research shown that using the adaptive controller will improve the performance (Trustworthiness-to-Cost Ratio). Using the new parameters (WQ, IT) in this adaptive controller provides a high flexibility without changing the performance. High level of the integrity can be achieved with more cost. So the Trustworthiness-to-Cost ratio does not change so much.

4. Conclusion

The intrusion tolerant database system presents a new paradigm in the field of database security. This system uses new techniques to react against attacks that cannot be identified by traditional security systems. Each intrusion tolerance technique provides different levels of integrity and availability under the same circumstances. In this paper, we presented a model which is a combination of several intrusion tolerance techniques. According to this model, the system can switch from one technique to another to provide the required levels of integrity and availability for the system at any time.

We used also a neural network for determining the significance degrees of data objects in the optimized attack isolation technique attempting to deploy other intelligent methods can form future research in this arena.

References

[1] Kruegel C and Vigna G, “Anomaly detection of web-based attacks”, In CCS’03, Washington, DC, USA, **(2003)** October 27-31, pp. 251–261.

[2] Ryutov T, Neuman C, Kim D and Zhou L, “Integrated access control and intrusion detection for web servers”, IEEE Transactions on Parallel and Distributed Systems, vol. 14, no. 9, **(2003)** September, pp. 814–850.

[3] Liu P, “Architectures for intrusion tolerant database systems”, Proc. 2002 Annual Computer Security Applications Conference, **(2002)** December, pp. 311-320.

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

[5] Liu P, Jing J, Luenam P, Wang Y, Li L and Ingsriswang S, “The Design and Implementation of a Self-Healing Database System”, Journal of Intelligent Information Systems, vol. 23, no. 3, **(2004)**, pp. 247-269.

[6] Lunt T, “A survey of intrusion detection techniques”, Computers & Security, vol. 12, no. 4, **(1993)** June, pp. 405–18.

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

[8] Chung CY, Gertz M and Demids KL, "A misuse detection system for database systems", In 14th IFIPWG11.3 Working Conference on Database and Application Security, **(2000)**.

[9] Stolfo S, Fan D and Lee W, "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] Liu P, Wang H and Li L, "Real-time data attack isolation for commercial database applications", Elsevier Journal of Network and Computer Applications, vol. 29, no. 4, **(2006)**, pp. 294–320.

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

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

[13] TPC Benchmark^{TM} C, http://www.tpc.org/tpcc/.

[14] Luenam P, Liu P and Norcio AF, "Adaptive Intrusion Tolerant Database Systems", VDM VERLAG DR. MULLER, Germany, **(2008)** December.

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

Haykin S, "Neural Networks: A Comprehensive Foundation", second edition. Prentice Hall, Chapter 4 Multilayer Perceptrons, **(1999)**, pp. 156−255

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

[17] Gupta MM, Jin L and Homma N, "Static and Dynamic Neural Networks From Fundamentals to Advanced Theory", chapter 4, **(2003)** John Wiley & Sons.

[18] Nelles O, "Nonlinear System Identification From Classical Approaches to Neural Networks and Fuzzy Models", chapter 11, Springer, Springer, Verlag Berlin Heidelberg **(2001)**.

[19] Hagan MT and Demuth HB, "Neural Network design", chapter 11, **(1996)**, PWS Publishing Company.

[20] Ammann P, Jajodia S and Liu P, "Recovery from malicious transactions", IEEE Transaction on Knowledge and Data Engineering, vol. 14, no. 5, **(2002)**, pp. 1167–1185.

[21] Chiueh T and Pilania D, "Design, implementation, and evaluation of an intrusion resilient database system", In Proc. International Conference on Data Engineering, **(2005)** April, pp. 1024–1035.

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

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

[24] Huang YW, Huang SK and Tsai CH, "Web application", In WWW 2003, Budapest, Hungary, **(2003)**, pp. 148–159, ACM.