ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2015

A Study on Secure Transactions in DBMS

Er. Bhavna Sharma

M.Tech Student, Dept. of CSE, Kurukshetra University, Kurukshetra , India

ABSTRACT: Database security is also a specialty within the broader discipline of computer security. Information is the most critical resource for many organizations. Existing intrusion detection systems use logs to detect malicious transactions. Logs are the histories of the transactions committed in the database. The disadvantage of using logs is that they require lot of memory. In addition to this sometimes even after a transaction is detected as malicious it cannot be rolled back. In this paper we present a method by which we can overcome the uses of logs and can detect malicious transactions before they are committed. We use specific user-profiles to store the sequence of commands in a transaction and use a prevention model for instant detection of malicious transactions.

KEYWORDS: null affinity algorithm; database security; intrusion detection system; transactions; malicious access; SQL commands

I. INTRODUCTION

Database Security is an concept that includes the following properties: authenticity, confidentiality, integrity), and availability. Due to the growth of networked data, security attacks have become a dominant problem in practically all information infrastructures. Simulation was carried out for a single as well as multiple users providing sequence of queries varying the size of the Database. A new approach for detecting malicious access to a database system is proposed and tested in this work. The proposed method relies upon manipulating usage information from database logs into three dimensional null-related matrix clusters that reveals new information about which sets of data items should never be related during defined temporal time frames across several applications. If access is detected in these three dimensional null-related clusters, this is an indication of illicit behavior, and further security procedures should occur. In this thesis, we describe the null affinity algorithm and illustrate by several examples its use for problem decomposition and access control to data items which should not be accessed together, resulting in a new and novel way to detect malicious access that has never been proposed before.

Our proposed method also has several benefits over other traditional methods of insider threat detection in that it requires little storage space, can be easily adjusted to reflect new applications, is very quick in operation once the historical data has been properly clustered, and requires only a moderate amount of computational time to calculate.

Intrusion literally means interrupting or interfering in others work. In a better way it can be defined as any set of actions that attempts to compromise integrity, confidentiality and availability of resource. Intrusion detection is a security technology that attempts to identify either individual who is trying to break into system and misuse information without authorization and\or those who have legitimate access to the resource but are taking undue advantage of their rights. The job of Intrusion Detection System (IDS) is to dynamically monitor the events occurring in a system and alert when any suspicious activity occurs so that defensive action can be taken to prevent or minimize damage. In general, the main goal of IDS is to detect malicious transactions before they are being committed and then dropping and rolling them back. If the malicious transactions have been committed and have caused damages, then locating the damaged parts and repairing them on time will be much more problematic. Intrusion detection systems serve three essential security functions: they monitor, detect and respond to unauthorized activity.

II. RELATED WORK

In a typical database environment the profile of the transactions that each user is allowed to execute is usually known by the DBA, as the database transactions are programmed in the database application code. In other words, the transactions are not ad hoc sequences of SQL commands. On the contrary, database transactions are well defined sequences of commands performing a finite set of predefined actions. For example, in a banking application users can only perform the operations available at the application interface (e.g., withdraw money, check account balance, etc). No other operation is available for the end users. Particularly, end users cannot execute ad hoc SQL commands.

The DBMTD mechanism uses the profile of the transactions defined in the database applications (authorized transactions) to identify user attempts to execute malicious transactions. DBMTD is built on top of the auditing mechanism implemented by most commercial DBMS. The audit lo is used by DBMTD to obtain the sequence of commands executed by each user, which is then compared with the profile of the authorized transactions to identify potential malicious transactions.

IDS are based basically on two models Anomaly Model and Misuse Model. Anomaly model establishes a normal activity profile for the system and if any activity fails to match the profile of the normal profile then the IDS considers it as an intrusion attempt. The misuse model is based on the assumption that there are ways to represent attacks in the form of a pattern or a signature so that even variation of the same attack can be detected. Here the IDS maintain a database of all the known attack signatures. It raises an alarm whenever the attack signature matches the one that the IDS have in its database. There are possibilities that the IDS might be unable to detect an intrusion attempt (false negative) or might catch a normal behavior as intrusion (false positive). IDS are of three types namely Network based, Host based, combined IDS. A network based IDS deployed outside the firewall monitors the data packets traveling over the network and any possible attack on the data packets to modify or read them are recorded by the IDS. A host based IDS is deployed on the host machine.

One more Intrusion detection system was proposed in which explained how to identify malicious transaction by checking for data dependency. Before any updating of data, it has to be read and after updating it has to be written back. There exists the pre-write set and post write set for a updating. Then identification of a malicious transaction is done by comparing the consistency in the pre-write set and post-write set of the user transactions. The detection model proposed in used to detect the malicious transaction after the transaction has been committed.

Another intrusion detection model was proposed in which detect the malicious transactions before they are committed. It considers a situation when an intruder performs a malicious transaction by transferring some amount from account A to account B without the intervention of the account A’s owner. Before the intrusion is detected and repaired the money may be drawn away. This transaction cannot be repaired. It is better to prevent a malicious transaction than detecting a curing it later which is an additional burden. Logs are used in recovery purposes to maintain the ACID properties. Logs are difficult to manage especially when systems of different systems are using. Logs cannot be employed in embedded systems. Since logs are usually stored in buffer if buffer is turned off then the logs are lost. If at all the logs are stored in the secondary storage the time is taken more to fetch and compare with the transaction profile. If a masquerader gets a log, he can know all the sequence of work done by the user.

III. PROPOSED WORK

The early research mainly focused on network-based and host-based intrusion detection. However, in spite of the significant role of databases in information systems, very limited research has been carried out in the field of intrusion detection in databases. We need intrusion detection systems that work at the application layer and potentially offer accurate detection for the targeted application. The approaches used in detecting database intrusions mainly include data mining and Hidden Markov Model (HMM). Chung presents a misuse detection system called DEMIDS which is tailored to relational database systems. DEMIDS uses audit logs to derive profiles that describe typical behavior of users working with the DBS. The profiles computed can be used to detect misuse behavior, in particular insides abuse. DEMIDS sue “working scope” to find frequent item sets, which are sets of feature with certain values. They define a notation of distance measure that captures the closeness of set of attribute with respect to the working scopes. These distance measures are then used to guide the search for frequent item-sets in the audit logs. Misuse of data, such as tampering with the data integrity, is detected by comparing the derived profiles against organizations security police or new audit information gathered about users. The main drawback of the approach presented is a lack of implementation and experimentation. The approach has only been described theoretically, and no empirical evidence has been presented of its performance as a detection mechanism.

DOI: 10.15680/ijircce.2015.0306059

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2015

First Approach: Database Intrusion Detection System for Role Based Enabled Database

The proposed approach in this section is, as from query based approach to transaction based approach. The main advantage of this approach is to extract the information among queries in the transaction. For example consider the following transaction:

Begin transaction
Select: a1, a2, a3, a4, a5 from t1, t2;
Update: t2 set a4= a2+1.2(a3);
End transaction

Where t1 and t2 are tables of the database and a1, a2, a3 are the attributes of table t1 and a4, a5 are the attributes of table t2 respectively. This example shows the correlation between the two queries of the transaction. It states that after issuing select query, the update query should also be issued by same user and in the same transaction. The approach based on the RBAC database uses the Naïve Bayes classifier as a learning algorithm to generate the role profiles on training data, and the training data which one is extracted from the log file and Users (Local/Remote) Database server Audit Server Application Layer stored into the form of particular representation to represent the user transaction behavior.

Second Approach: Database Intrusion Detection System Using Legal Transaction Profiles

Basically this proposed approach is divided into three steps: Auto-generation legal profile phase, Detection phase, Action phase. It takes the advantage over the manual transaction profiles mechanism. As in this case the time to generate the legal transaction profile is reduced, also it overcomes the disadvantage of the existing system based on manual profile generation. The log file is used from which the history of the transactions are extracted and stored into the offline audit trail and this can be done using the inclusion of existing auditing mechanism. Later the generated legal transaction profiles from offline audit trail are used at the detection phase to match with the executable transactions; if any deviation is there then particular executable transaction is marked as malicious otherwise committed into the database. The last phase is the action phase and it may take the action based on the alarm generated by the database IDS.

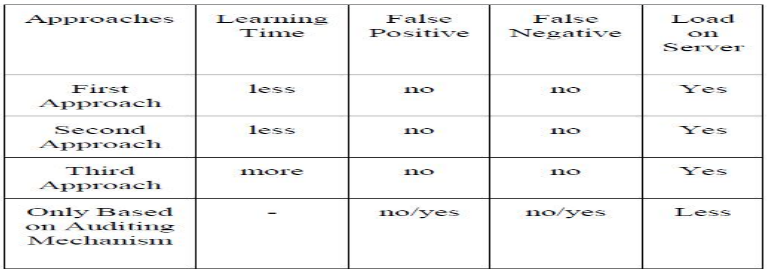
Third Approach: Database Intrusion Detection System Using Counting Bloom Filter (CBF)

A Bloom filter is used to define the bit array of m elements of n bits size and initially all set to 0. The filter uses a group H of k independent hash functions 1,........, k h h with range {1, . . . , n} that independently map each element in the universe to a random number uniformly over the range. For each element xÎS , the bits B [hi(x)] are set to 1 for 1 _ i _ k. (A bit can be set to 1 multiple times.) To answer a query of the form "Is yÎS ?", we check whether all ( ) i h y are set to 1. If not, y is not a member of S, by the construction. If all ( ) i h y hi(y) are set to 1, it is assumed that y is in S, and hence a Bloom filter may yield a false positive. The main problem with the bloom filter is the false positive i.e. it gives the wrong answer with correct query, and it is resolved using the counting bloom filter (CBF) where insertion and deletion of the set of the elements are possible. It also uses as similar to the bloom filter, k (random hash) functions, each of which maps or hashes some set element to one of the n bits array positions. To insert an element into a set, the element is passed into k hashing functions and k index values are obtained. All counters in counting bloom filter at corresponding index values are incremented. The overall approach based on the CBF is divided into the three phases.

IV. COMPARISON OF ALL THREE PROPOSED APPROACHES

For the comparison we consider the set of parameters to evaluate each approach with other one. The complete details of comparison are given in below table 1.

Table 1. Comparison of proposed approaches for database IDS



Approaches	Learning Time	False Positive	False Negative	Load on Server
First Approach	less	no	no	Yes
Second Approach	less	no	no	Yes
Third Approach	more	no	no	Yes
Only Based on Auditing Mechanism	-	no/yes	no/yes	Less

Based on the information in the above table as we can see the proposed approaches are very much useful to handle the malicious transaction once it is executed by the unauthorized user. The proposed approached also applicable to handle the internal misuse over the database. If we see the load on the database server for proposed mechanisms then it is quite high because of the inclusion of one additional layer of security into the database but it is less in auditing mechanism. The security in the DBMS is one of the main concerns of the researchers now-a-days and there is an interest to develop the possible database intrusion detection systems. We discuss the three approaches for database IDS and basic design of such architectures. We further intend to extend the work to support the actual implementation of action phase and then further for database recovery.

V. Conclusion and Future Work

We have presented the concepts and underlying architecture and shown how they can be applied. Our proposal relies on using historical data stored by the database logs on what data items were used at a particular time by various applications. We have used specific user-profiles to store the sequence of commands in a transaction and use a prevention model for instant detection of malicious transactions.

This information is then processed to reveal clumps of data items that should not be used together during certain time frames, resulting in a three dimensional usage matrix. This matrix allows a better prediction of potential misuse by allowing quicker and more precise prediction of items that should not be used together across the time, data item, and application dimensions Suspicious queries are then compared to the maximized usage array and a distance value is calculated for each non conforming action. These distances are summed to reveal how far from what was expected this access is. If the access is above a certain threshold, further security procedures are performed.

References

1. Marco Vieira and Henrique Madeira, “Detection of Malicious Transactions in DBMS”, IEEE Proceedings- 11th Pacific Rim International Symposium on Dependable Computing, Dec 12-14,2005, PP: 8.
2. Korra Sathya Babu, “Prevention of Unwanted Transactions in DBMS”, Department of computer Science and Engineering, NIT Rourkela, 2008.
3. E. F. Codd, "A Relational Model of Data for Large Shared Data Banks ", Comm. of the ACM(1970).
4. Ravi Sandhu and Pierangela Samarati, “Access Control: Principles and Practice”, IEEE Communications Magazine, September 1994.
5. Yi Hu and Brajentra Panda, “Identification of malicious transactions in Database Systems”, Proceedings of 7th International database engineering & Applications symposium, 16-18 July, 2013, PP 329-335.
6. L. Fan, P. Cao, J. Almeida, and A. Z. Broder. “Summary Cache: A Scalable Wide-Area Web Cache Sharing Protocol”, IEEE Transactions on Networking, 2000, PP 281-293.
7. Gordon, L. Loeb, M., Lucyshyn, W. and Richardson, Computer crime and security survey, R. Computer Security Institute, 2006.
8. Fonseca, J., Vieira, M., and Madeira, H. Online detection of malicious data access using DBMS auditing. In Proceedings of the 2008 ACM Symposium on Applied Computing. SAC'08. ACM, New York, NY, 1013-1020, 2008.
9. Chung, C. Y., Gertz, M., Levitt, K. DEMIDS: a misuse detection system for database systems.In integrity and internal Control information Systems: Strategic Views on the Need For Control, Norwell, MA, 159-178, 2000.

BIOGRAPHY

Er. Bhavna Sharma got her Bachelor of Technology (B.Tech) degree in 2013, from Kurukshetra University, Kurukshetra, India in Computer Science and Engineering Department. She is M.Tech Student at Kurukshetra University, Kurukshetra India in the Computer Science and Engineering Department.