International Journal of Information Technology and Knowledge Management
July-December 2010, Volume 2, No. 2, pp. 675-677

Detection of Malicious Transactions in DBMS

Ayushi[1], Anisha Sharma[2] & Reena Bansal[3]

Database management systems (DBMS) are a key component in the information infrastructure of most organizations and represent the ultimate layer in preventing unauthorized data accesses. Several mechanisms needed to protect data, such as authentication, user privileges, encryption, and auditing, have been implemented in commercial DBMS. In fact, malicious transactions executed by unauthorized users that may gain access to the database by exploring system vulnerabilities and unauthorized database transactions executed by authorized users cannot be detected and stopped by typical security mechanisms. In this paper we propose a new mechanism for the detection of malicious transactions in DBMS.

1. Introduction

Database security is the system, processes, and procedures that protect a database from unintended activity. Unintended activity can be categorized as authenticated misuse, malicious attacks or inadvertent mistakes made by authorized individuals or processes. Database security is also a specialty within the broader discipline of computer security.

Information is the most critical resource for many organizations. In many cases, the success of an organization depends on the availability of key information and, therefore, on the systems used to store and manage the data supporting that information. The protection of data against unauthorized access or corruption due to malicious actions is one of the main problems faced by system administrators. Due to the growth of networked data, security attacks have become a dominant problem in practically all information infrastructures.

Security breaches can be classified as unauthorized data observation, incorrect data modification and data unavailability. Unauthorized data observation results in the disclosure of information to users not intended to gain access to such information. Incorrect modification of data either intentional or unintentional, results in an incorrect database state. Data unavailability results in improper and untimely functioning of the organization.

Security is an integrative concept that includes the following properties: confidentiality (absence of unauthorized disclosure of a service or piece of information), authenticity (guarantees that a service or piece of information is authentic), integrity (protection of a service or piece of information against illicit and/or undetected modification), and availability (protection of a service or piece of information against possible denials of service caused by malicious actions).

In database management system the above solutions are met with different components or methods. Initially to gain access to a database a user has to provide identification and authentication. Identification is the means by which a user provides a claimed identity to the system (i.e. using a username). Authentication is the means of establishing the validity of this claim (password). Issuing identification and initial authentication key to the users is the job of DBA. The user can change his identification and authentication to enter into the database later on.

Access Control Methods ensures confidentiality in DBMS. If anybody tries to access the data object the Access Control Methods checks for the rights of the user against the data object with the set of authorizations stated by the Administrators. Access is the ability to do something with a computer resource.

Several mechanisms needed to protect data have been proposed and/or consolidated in the database arena. Some examples include user authentication, user privileges, data encryption, auditing, etc. The main goal of database security mechanisms is to protect the data stored in the database from unauthorized accesses or malicious actions in general. The success of a security attack depends on the vulnerabilities of the system (attacks are harmless in a system without vulnerabilities) and vulnerabilities are not dangerous if the system is not subject of security attacks.

Typical database security attacks can be classified as:

1. Intentional unauthorized attempts to access or destroy private data;

[1]Lecturer, Hindu College of Engg., Sonipat, Haryana

[2]Lecturer, JIMS, GGSIPU, Vasant Kunj, Delhi

[3]Student, M.Tech (IT-W) GGSIPU, Delhi

Email: [1]ayushibmiet@gmail.com, [2]anisha_shrma@yahoo.co.in, [3]bansalrina@yahoo.co.in

2. Malicious actions executed by authorized users to cause loss or corruption of critical data.
3. External interferences aimed to cause undue delays in accessing or using data, or even denial of service.

There are many cases where the execution of malicious sequences of SQL1 commands (transactions) cannot be detected (or avoided). The following points present some examples:

- The DBA does not activate the necessary security mechanisms (e.g., authentication, user privileges, data encryption, auditing, etc), which allows intruders to get access to database data.
- The security mechanisms available are incorrectly configured permitting potential intruders (hackers) to access the database.
- Hidden flaws in the database implementation may allow hackers to connect to the database server by exploring those defects.
- Unauthorized users "still" the credentials of authorized users in order to access the database server.
- Authorized users take advantage of their privileges to maliciously access or destroy data.

Malicious transactions may damage the database integrity and availability. However, in spite of the pertinence of the detection of malicious database transactions, the reality is that no practical mechanism able to identify users executing malicious transactions has been proposed so far. This paper proposes a new mechanism for the detection of malicious transactions in DBMS. As in a typical database environment it is possible to define the profile (sequence of SQL commands) of all the transactions that each user is allowed to execute, the proposed mechanism (named DBMTD - Database Malicious Transactions Detector) uses that profile of valid transactions to identify user's attempts to execute invalid sequences of commands.

Security in DBMS

A DBMS allows users to define the data to be stored in terms of a data model, which is a collection of high-level metadata that hide many low-level storage details. Most DBMS are based on the relational data model. The relational data model defines a database as a collection of relations, where each relation is a table with rows and columns. The user actions are translated into SQL commands by the client application and sent to the database server. The results are sent back to the client to be displayed in the adequate format by the client application.

The main goal of security in DBMS is to protect the system and the data from intrusion, even when the potential intruder gets access to the machine where the DBMS is running. To protect the database from intrusion the DBA must prevent and remove potential attacks and vulnerabilities.

A database server manages tables from multiple database applications stored in different database schemas (a database schema is a collection of tables and other related objects such as views, sequences, etc).

Users connect to a DBMS via a client application. To access the server the client application must gain access by passing through an authentication process. DBMS can provide internal user authentication (by using internal usernames and passwords) or use external user authentication (e.g., Kerberos, NTS, etc.).

To guarantee that users only do what they are authorized to do, the DBMS normally implement a security mechanism based on user privileges. These privileges allow the system to control the actions that the users are allowed to perform. Two types of user privileges are normally provided: system privileges and object privileges. Systems privileges are related to the actions that users can perform (e.g., create users, create procedures, create tables, create indexes, etc.). Object privileges are related to the access to the database tables (e.g., selecting, deleting, updating, inserting, etc).

To provide communication in a secure way and avoid the access to data transferred over the network, sophisticate DBMS provide encryption mechanisms for data communication.

A New Approach to Concurrent Intrusion Detection in DBMS

This paper proposes a new mechanism for the detection of malicious transactions in DBMS, the Database Malicious Transactions Detector (DBMTD) mechanism. In practice malicious database transactions are related to security attacks carried out either externally or internally to the organization. External security attacks are intentional unauthorized attempts to access or destroy the organization's private data. These attacks are perpetrated by unauthorized users (hackers) that try to gain access to the database by exploring the system vulnerabilities (e.g., incorrect configuration, hidden flaws on the implementation, etc). On the other hand, internal security attacks are intentional malicious actions executed by authorized users. The DBMTD mechanism uses the profile of the transactions defined in the database applications (authorized transactions) to identify user attempts to execute malicious transactions.

The audit log 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. DBMTD is a generic mechanism that can be implemented in any DBMS that provides auditing functionalities.

Figure 1 presents the basic architecture of a database system with the DBMTD mechanism.

Using the DBMTD mechanism consists of two different phases: transaction profiling and intrusion detection.

Transaction profiling corresponds to the identification of the sequence of commands that constitute each valid transaction and intrusion detection consists in the detection of users executing sequences of commands that potentially represent intrusion attempts.

Transactions Profiling

Profiling consists in representing the authorized transactions as sequences of valid commands. A database transaction can be represented as a direct graph describing the different execution paths (sequences of selects, inserts, updates, and deletes) between the beginning of the transaction and the commit or rollback command.

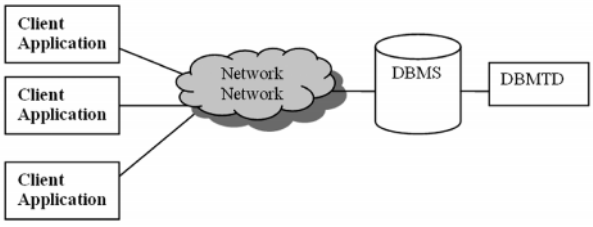

Fig. 1: DB SYSTEM Using the DBMTD

Intrusion Detection

Our mechanism for the detection of database malicious transactions is built on top of the auditing mechanism.

Our proposal is to run DBMTD as an autonomous subsystem separated from the DBMS (sharing the same machine or, preferably, in a dedicated machine). The DBMTD mechanism can also be implemented internally to the DBMS using triggers.

The auditing mechanism collects information about the commands (e.g., login, select table, insert into table, delete from table, etc.) executed by the database users. Some typical information logged for each command includes: username, session identification (tag that identifies the user session), sequence number (number that identifies the sequence of the command in the corresponding session), command type (e.g., login, select table, insert table, create table, etc), name of the target object (table, index, cluster, etc), owner of the target object, transaction identification (label that identifies the transaction to which the command belongs), etc.

The efficiency of an intrusion detection mechanism like DBMTD can be characterized by the following measures: coverage (percentage of successful malicious transactions detected), latency (time between the execution of the malicious transaction and its detection), false positives (number of valid transactions identified as malicious transactions), and impact in the performance (performance overhead introduced by the mechanism).

Coverage

The coverage represents number of malicious transactions detected. As already discussed before each pair of size of CBF and number of hashing functions used represents a different configuration of IPS mechanism.10000 malicious transactions are submitted for every configuration of system and number of malicious transactions detected are recorded. Then Coverage percentage is calculated by using following formula

Coverage% = ((No. of malicious transactions detected)*100) / (No. of malicious transactions submitted)

2. CONCLUSIONS AND FUTURE WORK

This paper proposed a new mechanism for the detection of malicious transactions in DBMS. The proposed mechanism uses a graph that represents the profile of valid transactions to detect unauthorized transactions and consists of two different phases: transaction profiling and intrusion detection. Intrusion detection consists in the detection of users executing sequences of commands that potentially represent intrusion attempts.

In the present version of DBMTD the profiles of the valid transactions are defined manually by the DBA. The inclusion of the proposed mechanism as a native feature of an open source DBMS (PostgreSQL) is also being considered for future work.

REFERENCES

[1] Szewczyk, R., Mainwaring, A., Polastre, J., Anderson, J., Culler, D.: “An Analysis of a Large Scale Habitat Monitoring Application”, In: 2nd ACM Conference on Embedded Networked Sensor Systems (2004).

[2] Simon, G., Maroti, M., Ledeczi, A., Balogh, G., Kusy, B., Nadas, A., Pap, G., Sallai, J., Frampton, “K.: Sensor Network-based Countersniper System”. In: 2nd ACM Conference on Embedded Networked Sensor Systems (2004).

[3] Aslam, J., Butler, Z., Constantin, F., Crespi, V., Cybenko, G., Rus, D.: “Tracking a Moving Object with a Binary Sensor Network”, In: 1st ACM Conference on Embedded Networked Sensor Systems (2003).