International Journal of Computer Applications (0975 – 8887)
Volume 87 – No.7, February 2014

Countermeasures to Database Security: A Survey

Shagufta Rajguru
Assistant Professor
F.C.Rodrigues Institute of Technology
Vashi, Navi-Mumbai, India

Deepak Sharma
Associate Professor
Vidyavihar,Mumbai,India
K.J.Somaiya College of Engg

ABSTRACT

A database is a collection of data of different types, while DBMS is a set of specifications that tells how data is to be stored in a database and how data should be accessed by the components of the database. The data and the metadata when exposed to the outside world may endanger the security of the DBMS. Therefore precautions should be taken that the data should be available to the right user at any point of time.

In this study the architecture of the database is understood and tried to identify the vulnerabilities and threats related to them. Also it is noticed that if the architecture of the database is secured it would provide secure data management system. Different security methods implemented for different scenarios are discussed in this paper. With the help of these tools, a proposal for the security at the application level (View level) of the database architecture is suggested.

Keywords

Database, DBMS, architecture of DBMS, vulnerabilities, threats

1. INTRODUCTION

A. Types of Databases

Databases are classified depending on the types [1] of contents stored in it for example graphics, sound, images etc. One can also categorize the database depending on the application area, structure or interface. The following are some of the areas where one can classify them accordingly.

- Individual or microcomputer
- Company or shared
    1. Operational
    2. User
- Distributed
- Proprietary
- Web Database

An Individual database also called as microcomputer database is stored on a local hard drive or on a LAN file server. Company or shared database is often stored on a mainframe and managed by the company administrator. Common operational databases are used for inventory, production, sales. Common user operational database combine internal operational data with outside private (proprietary) databases. In a distributed database data is stored in several locations and combined via a communication network. Proprietary database is an enormous database from an outside vendor also called information utilities or data banks

B. Abstraction of Database

The data must be retrieved efficiently by the user. A major purpose of a database system is to provide users with an abstract [2] view of the data. It should hide the complexity from users through several levels of abstraction, to simplify user's interactions with the system:

- Physical level
- Logical level.
- View level.

C. Database Architecture

The architecture [2] of a database is greatly influenced by the underlying computer system on which the database runs. The organization of this paper is as follows. The paper presents different vulnerabilities and threats at various components of the database architecture in section 2 while section 3 illustrates some countermeasures to handle them .Section 4 discusses the proposed idea of developing a secured application level (view level) of the architecture of the database. Finally, section 5 presents concluding remarks and outlines future work.

2. VULNERABILITIES AND THREATS AT DIFFERENT COMPONENTS OF DATABASE ARCHITECTURE

The vulnerabilities faced at different components of the database as explained are illustrated. Also some of the threats faced because of the vulnerabilities are explained.

A. Vulnerability in authorization and integrity manager

The security at authorization and integrity manager [3] can be violated due to unauthorized access by insiders, brute force attack, incorrect usage and personal hardware collection. The other threats are data at rest (unencrypted information), sensitive data, poor application architecture, password vulnerability, unlocked database and vendor bugs. Due to these vulnerabilities the very common threat is misuse of sensitive information like credit card information.

- *Credit Card hacking technique [3]*

To acquire credit card number, the hacker may use a technique called as phishing. He may pose himself as an important company personal and send an email or SMS to receivers for sending their credit card details since there is some governmental inquiry. The receivers will give away their details to him in response thinking that he is a legitimate person. Credit cards can be read through clothes and wallets. Therefore Radio frequency identification (RFID) is an easy process to read the credit card information.

B. Vulnerability and Threats in administration tools

Earlier for securing the database, sniffers were used but using the sniffers 100% logging is not achieved [4]. Therefore gateways were introduced along with the access control. But if the policies of the gateways are broken, then the person may get access to entire database. Hence to overcome this drawback of gateways, data masking became the important aspect of database security. Data masking is explained in section C.

C. Vulnerability and Threats in data (Disk Storage)

Data masking is the process of hiding (masking) specific data elements within data stores. The main reason for applying masking to a data field is to protect data that is classified as personal identifiable data, personal sensitive data or commercially sensitive data, however the data must remain usable for the purposes of undertaking valid test cycles. It must also look real and appear consistent. There are some challenges that needs to be taken care while implementing data masking [4].

- It is impossible to detect and avoid objects vulnerable to security in the server
- It is impossible to detect misuse of access authorized by security policy.
- It is impossible to identify important information or personal information of an object.
- It is impossible to identify modification information of the object.
- There is no scheme for checking vulnerability information for the entire database itself.

D. Threat vectors for DBMS

Some of the threat vectors [5] that needs to be noted while designing a secure DBMS Configuration are listed below. The threats mainly arises because of the weaknesses in the architecture of the DBMS, any loop hole found in the security of the DBMS or not monitoring the rights of the user to what he or she is authenticated to access the services of DBMS. This may lead to exploitation of the database which may further cause serious hazards to the organization owing to such a database. Therefore care should be taken in preventing the vulnerabilities if detected to avoid any damage the data.

- Legitimate excessive privilege achievement
- Illegitimate privilege elevation
- Denial of Service
- Communication weakness
- Authentication weakness
- Threat Vector
- Side-channel data exposure
- Audit trail weakness
- SQL Injection enhancement

3. SECURITY OF DATA

Database security is based on three main areas: confidentiality, integrity, and availability. Confidentiality represents the protection of data from unauthorized exposure; integrity refers to the prevention of unauthorized data access, while availability is achieved through identification of and recovery from hardware and software errors or malicious activity. Traditionally, database security focused on user authentication and managing user privileges to database objects. For increasing the database security not only the database must be secured, but also protection to the underlying host, operating system, and communication with the clients must be maintained.

In this section some of the tools for the security of the database are discussed.

A. Multi-Layered Approach to data anonymity for database security

Here a multilayered approach [5] to data anonymity is designed on a multilayer security infrastructure. There is an inference engine that provides a means for securing static and dynamic data. Static data consists of name, address, pan card, while dynamic data like age, gender, city, etc is generated by using static data. For creating such dynamic data, a re-identification algorithm is used. The re-identification algorithm is K-anonymity and is triggered after every modification in the database. The k-factor is decided depending on the number of records in the database.

This approach is helpful in providing security for communication, operating system and database. An operating system is chosen and set to MAC module (Mandatory Access Control).The DBMS system that supports authentication, authorization, integrity with columnar level of encryption is supported. For securing the communication between client and server an encrypted channel that uses certificates is used. The client will connect to the server by using the certificate. If the connection succeeds then it can query the server. The server passes the queries to the anonymity module for the query to be checked and parsed and then to the database server. In reply the database server will send it back to the same anonymity module for further processing and then sent back to the client from server. The Java Server uses the SSL-secure socket layer to listen on TCP port2345. The server also maintains a list of usernames and passwords and also a list of accepted client certificates. The usernames are required for the data anonymity engine to detect any possible data privacy.

B. Logging Scheme

A tool of database logging [6], which monitors and log the database activities through analyzing network traffic. It contains three principal components: packets capturing, packets parsing and data storage. First capture the packets forwarded to the database; then, by analyzing database communication protocols, parse the captured packets; finally, use the parsed results to support database audit. The logging scheme is implemented using the agent that sniffs the packet from the network and monitors traffic into and out of the database.

The agent is connected to the database server using the hub and network card is set to promiscuous mode for allowing to capture packets. After packets have been captured they are analyzed using database protocols like TNS (Transparent Network Substrate), TDS (Tabular Data Stream) and DRDA (Distributed Relational Database Architecture) to perform packet parsing. The results are stored in the server for auditing.

C. Trust-Based Benchmark

A benchmark is a term that is used to assess and compare systems according to the given characteristics. A security benchmark [7] for assessing and comparing DBMS configurations, based on a trust-based metric called Minimum Untrustworthiness has been proposed. The goal of the benchmark is help the DBA understanding how untrustworthy (at least) a DBMS configuration is when it comes to comes to prevent the manifestation of relevant threats. The benchmark is computed using the following steps.

1. *Database configuration threat vectors*
The threat vectors for DBMs are listed in section 2 for deciding the database administrator to design the DBMS configuration by keeping the following factors.

- He is expected to apply and configure security mechanisms that guarantee that the policies and rules are enforced to the maximum extent possible.
- He is expected to dissuade attempts to break the rules.
- He is expected to maintain mechanisms that help identifying potential violations of these rules, including being able to pinpoint possible suspects.

2. *Pessimistic Scenarios*
There are many security best practices stated by the experts. But all of them cannot be enumerated; hence an approach to narrow down these elements that can influence the security is needed. It is important to notice that most of them have to be combined with other problems in order to be exploited to perform an attack.

3. *Computing minimum untrustworthiness*
The minimum untrustworthiness computation is based on the reasoning that, if a security recommendation is not properly followed in a given configuration, then the corresponding pessimistic scenarios are feasible. If the scenarios are feasible, then the only elements that prevent the execution of the attacks are intention. The goal is to give general trends of untrustworthiness to help the DBA becoming aware of three critical pieces of information: first, what are the threats for which there are more configuration problems; second, what are the interaction classes that are more likely to accomplish such threats; finally, what are the security recommendations that must be considered to reduce the untrustworthiness level related to a particular threat and/or interaction class. Afonso Araújo Neto have suggested an algorithm to calculate a metric that helps to determine the degree of untrustworthiness.

D. An automatic mechanism for sanitizing malicious injection
This system [8] is like a firewall implemented to protect web sites from attack. There is a hybrid analysis procedures and a bypass testing procedures. By means of testing result, it can find some programs having injection vulnerabilities and create a sanitizing website consisting of meta-programs on security gateway. For example, if there is a vulnerable server-side program named 'verify.exe' on web server, the system can automatically produce a meta-program named 'verify.php' on security gateway. By means of the meta-programs that include an adjustable validation function having proper filtering rules, malicious injection can be sanitized.

The tester in this system generates various test cases having expected output. The verifier compares the actual output obtained from the monitor with the expected output and determines whether it is same or not. If it doesn't match then it is implied that the defense system has authenticated the input. The Reporter is used to store the result of the test cases that can be used for further references.

E. Database Intrusion Detection by Transaction signature
In this proposed approach [9] signatures of valid transactions are created and used to identify the malicious intrusive activities. This scheme has learning phase, detection, and response phase.

1. *Learning Phase*
In this phase normal or legitimate behavior of user's is learnt. There are records in audit log of DBMS which contains users' action. In this proposed system, offline audit log data store and Legitimate Transaction signature generation modules, which both actually used for identifying normal and acceptable behaviour of user's database transactions are stored. All historical transaction with database is accessed to understand the behavior of legitimate transactions. Various techniques like trigger generation or enabling audit log with database is used for same purpose.

2. *Generating signature for user's action*
Pre-processing module is used for extracting necessary information from transactions user performs with database. In pre-processing module extraction of key words, operations and target entities are done and kept in dataset. So output of pre-processing is dataset or transaction set which can be used for next module.

3. *Response*
The response module decides action depending on whether user's behaviour is legitimate or not. Whatever the output of the signature generation algorithm, will be compared with signature derived from historical data. Based on this comparison this module will decide what action should be taken.

F. Security against SQL Attacks
There are some forms of SQL attacks [10] like bypassing authentication, making key operations of the database, or executing system commands of the database. In this approach there are two methods explained. They are as follows.

- *Protection for ordinary users*

Ordinary users can get information what they want from the site, is the main service targets of the web site. The current website generally uses dynamic background management, according to the template technique producing a static front page, so the website presents to users most of static pages and some dynamic interactive pages, such as inquiries. For this situation, the ordinary user protection model, protects dynamically constructed SQL occasions and prevent unauthorized users tampering the database. Server distinguishes requests from the client, if the request is a static page, then go to the normal process of the general page. If the request needs front page to interact with the database, then the corresponding SQL statement is checked. If the SQL statement passed checking, then it goes to the normal process, or else do error handling. It filters illegal characters, to prevent an attacker to modify the meaning of SQL commands, tries to execute the query with a stored procedure, also it restricts user privileges by limiting the implementation of high-privileged SQL statement for users who don't have corresponding privilege.

- *Protection for Administrators*

The administrator should have high data manipulation authority, since they need to maintain and upload contents on the site. This model works on two factor authentication methods: user name plus password authentication and verification of hardware encryption lock. The user name plus password login, filters the user name and password which is entered by user and encrypt user information in the database. The hardware encryption lock when inserted the client sends a request message to the server, the server sends the reply packet, which includes the information that should be tested by client using encryption locks. The client uses encryption locks process information and returns results to the server, server validates whether the information is correct, if correct, then allows to continue process, otherwise forbids all the operations of the administrator.

G. TRDBAC:Temporal Reflective Database Access Control

Temporal Reflective Database Access Control (TRDBAC) [11], this model allows the policy makers and database administrators to design and implement time based database access control policies. RDBAC is a model in which a database privilege is expressed as a database query itself, rather than as a static privilege contained in an access control matrix. In this model access control policy decisions can depend on data contained in other parts of the database, such as attributes of the user, attributes of the data being queried, or relationships between the user and the data. But RDBAC is used at database level to implement the access control policies, while TRDBAC is used to implement policies at application level.

For example if the Exam Coordinator is not available due to certain reason and a necessary change in the results is required then we may allow the instructor to update the results for a certain time period. For instance, the instructor is allowed to update the results from June 20, 2013 10:00 AM to June 30, 2010 6:00 PM. Such type of policies can be implemented using some time constraints on the subjects (e.g. instructor).

H. Design of a New Web Database Security Model

The proposed web database security model [12] contains auditing module for performing auditing, control modules for granting access rights to the users and twice logging module for authenticating the user. Twice login module adopted two connections to web database. When first connect web database, the twice login module logins web database with base rights account. After verifying user identification, the application system need acquire main account and password with corresponding user. Base right account and password are stored in BaseRightUser table.

The audit module set up trace through database storage procedures to collect audit data. The audit module stores and transports audit data in standard XML format to the audit server. Thus, audit log file can be stored securely. Users are divided into many types on basis of function of users, the authorized user are allowed to access the least module in the web database application system. So there would be least damage to the database. When the application logged in second time, the user's rights were examined and accordingly they were granted access by the program control module.

The Database Right control module managed the tables that user shouldn't access, the application system shouldn't grant user any rights. The application system auto-grant user some operation rights of the database with SQL commands. View has characters of table and query. View is similar to query, that is, view can extract some useful data from one or many related tables or views. View is similar to table, view can update data of it and store permanently updated results into disk. The user can perform operations (link, project, union and select) on the data in the view and thus database security could be strengthened by not directly doing changes on it.

I Overheads due to security implementations

In this approach [13], a test on unsecure database placed on an unsecured operating system using plain text communication (unsecured) was run .Over the unsecured text, d different layers of security MAC (Mandatory Access Control), COM (Communication), and ENC (Database storage encryption) is placed in order to observe the overhead into independently by each security layer.

The test consisted in dropping any existing database and then creating a new database with the required tables and constraints. The next step inserts millions of records in the specific tables and finally the integrity of data was checked using pre-computed CRC values. For unsecured database, AppArmor (AppArmor is a Mandatory Access Control (MAC) system which is a kernel (LSM) enhancement to confine programs to a limited set of resources. AppArmor's security model is to bind access control attributes to programs rather than to users) is uninstalled. For the COM test, SSL communication via Java Anonymity engine using X.509 certificate is used. For the MAC test, AppArmor is installed and for the ENC test, testing scripts is modified because MYSQL doesn't natively data encryption, and the encryption function had to be called explicitly from the client. The encryption algorithm used was AES with a key length of 128 bits.

Due to the input-output nature of the database server, as shown in the table 2.3 the mandatory access control layer and the secure communication add a negligible overhead, of less than 10%, because the added amount of security checks are performed simultaneous with other database tasks. On the other hand, the encryption level has an overhead of over 100%, due to intense computation on the server side. As a result, we have chosen not to use storage encryption for our infrastructure.

Table 1. Database System test results (s)

Test type	**Time (s)**			
	Database creation	***Insert queries***	***Integrity checks***	***Total SQL time***
Unsecure	0.05	44.01	12.68	56.74
MAC	0.08	45.90	12.92	58.90
COM	0.08	45.57	12.45	58.10
ENC	0.11	103.26	26.75	130.12
M+C	0.09	45.86	13.07	59.02
M+C+E	0.12	108.79	27.13	136.04

4. PROPOSED MODEL FOR THE APPLICATION LEVEL (VIEW LEVEL) IN THE ARCHITECTURE OF THE DATABASE

Various vulnerabilities and threats have been discussed in the sections above. Also different security measures are enumerated and explained in details. Depending on these factors one can refine the application level of architecture of the database by extending security methods to it.

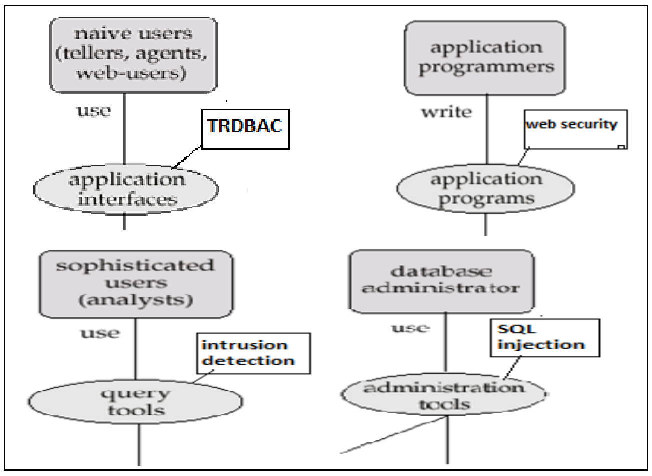

Fig 1. Proposed Model of View Level for database architecture

In the fig 1, some counter measures as discussed above can be extended to the View level. These tools can be implemented at the user interfaces of the application level[2]. The security tools if added to the application level of the architecture of the database, a more secure and preventive database management system can be obtained.

5. CONCLUSION

Some vulnerabilities and threats of components in database architecture are discussed in details. Thus it can be concluded that securing the database at the application level would protect the lower abstraction levels of database. Providing countermeasures between the interface and application levels would make the database architecture more reliable. One can use some of the countermeasures as discussed above to refine the architecture of the database.

We would like to enhance the proposal as a future work for the security of database.

6. REFERENCES

[1] Timothy J O'Leary, Linda I. Oleary Computing Essentials 2005,TMH

[2] Abraham Silberschatz, Henry F. Korth, S. Sudarshan, Database System Concepts, Fifith Edition.

[3] Singh, Sartaj. "Inherent Dangers in Database Security." Computing Sciences (ICCS), 2012 International Conference on. IEEE, 2012.

[4] Baek, Jong-Il, and Dea-Woo Park. "A study on database vulnerable object analysis and control technology." Information Science and Digital Content Technology (ICIDT), 2012 8th International Conference on. Vol. 3. IEEE, 2012.

[5] Popeea, Traian, Anca Constantinescu, and Razvan Rughinis. "Providing data anonymity for a secure database infrastructure." Roedunet International Conference (RoEduNet), 2013 11th. IEEE, 2013.

[6] Huang, Qiang, and Lianzhong Liu. "A Logging Scheme for Database Audit."Computer Science and Engineering, 2009. WCSE'09. Second International Workshop on. Vol. 2. IEEE, 2009.

[7] Neto, Afonso Araújo, and Marco Vieira. "A trust-based benchmark for DBMS configurations." Dependable Computing, 2009. PRDC'09. 15th IEEE Pacific Rim International Symposium on. IEEE, 2009.

[8] Lin, Jin-Cherng, Jan-Min Chen, and Cheng-Hsiung Liu. "An Automatic Mechanism for Sanitizing Malicious Injection." Young Computer Scientists, 2008. ICYCS 2008. The 9th International Conference for. IEEE, 2008.

[9] Rathod, Yagnik A., M. B. Chaudhari, and G. B. Jethava. "Database intrusion detection by transaction signature." Computing Communication & Networking Technologies (ICCCNT), 2012 Third International Conference on. IEEE, 2012.

[10] Yan, Yi, Su Zhengyuan, and Dai Zucheng. "The database protection system against SQL attacks." Computer Research and Development (ICCRD), 2011 3rd International Conference on. Vol. 3. IEEE, 2011.

[11] Rashid, Zahid, Abdul Basit, and Zahid Anwar. "TRDBAC: Temporal reflective database access control." Emerging Technologies (ICET), 2010 6th International Conference on. IEEE, 2010.

[12] Yangqing, Zhu, et al. "Design of a new web database security model."Electronic Commerce and Security, 2009. ISECS'09. Second International Symposium on. Vol. 1. IEEE, 2009.

[13] Popeea, Traian, et al. "Inference Detection and Database Security for a Business Environment." Intelligent Networking and Collaborative Systems (INCoS), 2012 4th International Conference on. IEEE, 2012.

[14] Wang, Shuguang, and Shao Qian. "The Analysis of Database Remote Attack Defense." Intelligence Information Processing and Trusted Computing (IPTC), 2010 International Symposium on. IEEE, 2010.