DATABASE SECURITY

Kamaljeet Kaur
Computer Science & Engineering Department
Guru Nanak Dev Engg. College, Ludhiana.
Punjab-India
meetk.89@gmail.com

***Abstract*— Ensuring the security of databases is a complex issue. The more complex the databases are the more complex the security measures that are to be applied are. Network and Internet connections to databases may complicate things even further. Also, each and every additional internal user that would be added can create further serious security problems. The purpose of this paper is to highlight and identify the main methods and facets of attack on a database, as well as ways to deflect attacks, through focusing on the delicate issue of data inference. This paper will examine the different topics related to database security and see the adaption of the research to the changing environment.**

***Keywords:* Attack, Access Control, Discretionary Access Control, Encryption, Inference, Mandatory Access Control, Role Based Access Control, Security, Threats.**

*****

I. INTRODUCTION

Database Security is one of the broader topics. Database servers are highly complex systems – storing, organizing, and managing data for a wide array of applications. Most mid-sized firms have dozens of them, some embedded in desktop applications, while others serve core systems such as web commerce, financials, manufacturing, and inventory management.
Research and development in the area of database technology during the past decade is characterized by the striving for better support for applications beyond the traditional world, where primarily high volumes of simply structured data had to be processed efficiently. As a result, future DBMS will include more functionality, and explicitly cover more real world semantics (in various forms) that otherwise would have to be included in applications themselves. Advanced database technology, however, is in a sense ambivalent. While it provides new and much-needed solutions in many important areas, these same solutions often require thorough consideration in order to avoid the introduction of new problems. One such area is database security.
Database is an integral part of any information system and they often hold sensitive data. The security of the data depends on physical security, OS security and DBMS security. Database security can be compromised by obtaining sensitive data, changing data or degrading availability of the database. As systems become more modular and sophisticated attacker is presented with more vectors to conduct an attack. More over the attacker motivation have changed during the last 30 years. Over the last 30 years the information technology environment have gone through many changes of evolution and the database research community have tried to stay a step ahead of the upcoming threats to the database security. The changing environment, non secure implementations and user errors is a main contributor to the explosive growth of the IT incidents and vulnerabilities.

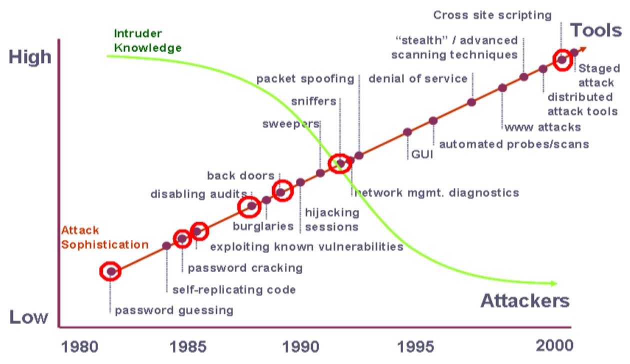

Figure 1. CERT 2000

In early years (pre 1980) database management systems especially the RDBMS was not wide spread. The commercial applications, very few though, were not greatly under threat since a formal security model was still under research. Physical threats were very well understood and security mechanisms were in place like securing the servers in authorized rooms. Logical threats were facing challenges and the security controls were not strong enough.
In the early 1980s, commercial applications using database systems started spreading. The attackers were motivated

more out of the competition. Database systems were required to be stored in shared resources, increasing the need for security across the network and the users.

The digital environment during the 1991-2000 era went through massive transformation driven by the rapid commercialization. The creation of Windows browser along with the realization that World Wide Web can be used for more than just information sharing, but actual commerce had a tremendous impact on the digital information exposure that made the security requirement of early 90's very different from the late 90's. Another dominant development in computer science at this time was the programming methodology of Object Oriented Programming. These concepts have been added to many of the programming language of the time and it was natural for the database community to ascertain the efficient way to deal with complex object data. ODBMS (Object Oriented Database) took roots at this time as an alternative to RDBMS that is better suited for certain tasks commonly associated with OOP. [1]

The period of post 2001 thought data becoming ubiquitous. The new data types such as spatial and continuous sensor data entered the database field. Personal information about individual became readily available through public records.

II. ENVIRONMENT THREATS

In early years (pre 1980), Attackers were mostly sophisticated users who pursued attacks as a challenge. Often, attackers used custom tools mostly developed by them or no tools at all. As a result, the attacks were not very destructive and sophisticated attack tools were not available to non sophisticated users. The main research was focused on the inference of sensitive data in statistical databases.

In the early 1980s, Most of the enterprises did not yet allow external connectivity but the number dumb terminal was replaced with client server model.

In the period 1991-2000, Connectivity to the Internet brought about addition of untrusted connections. While firewalls provided protection against direct database attacks, the application front-end to the database were left vulnerable.

The period of post 2001 saw the emergence of information security attack as a large multi-billion businesses. Attackers have built sophisticated command and control networks running on millions of compromised computers. Attackers have also used sophisticated security techniques. Attackers have also attempted to minimize any impact on the compromised system to limit the possibility of detection. The attackers have developed sophisticated business plans and many different avenues to profitability.

A. Statistical Database & Inference

Initially the database systems were not applied to store data in shared resources; the major threats were in the inference of statistical data by users. Statistical databases were used to store data about the population or military information which were intended for aggregating useful information to be used by statisticians. [2]

The data stored in database system were queried to get the aggregate results. In case of indirect attack, the aggregate results could be interpreted and Trackers could be applied on such results set to draw sensitive information. [2]

III. ACCESS CONTROL

Access control for database was to be expressed in terms of logical data model with authorizations in terms of relations and tuples. It also had to be content aware to allow the system determine whether access should be granted based on the content of the data item.

In early years an important and benchmark work on access control was done for military applications. This was imperative during this period because of the classified data stored in military applications. Data could belong to different levels and people accessing such data were given permissions based on the level of the user and the level of data sought by the user.

This came out to be called the famous lattice model. This came out to be called the famous lattice model or the Bell-La Padula model (BLP). A lattice model consists of various levels and objects belonged to any of these levels based on the sensitivity of the data. Typical classification of military data was Top Secret, Secret, etc. The users were also mapped to one of the level in the user lattice. So, this model ensured that users could not read\write sensitive data higher or lower in the hierarchy. This model was developed to formalize the US Department of Defense multilevel security policy. This model was not completely applied in database management system until the early 1980s. [3]

The focus on security policies and mechanisms improved leading to increased momentum in area like the access control. Extensive research was carried on in access control and several models were proposed.

Another area which also started seeing challenges was the multilevel database security and Mandatory Access Control (MAC) models were researched that govern the access on the basis of classification of subjects and objects in the system. The Discretionary Access Controls (DAC) specified the rules which allowed the users, at their discretion, to create and delete objects, grant and revoke authorization for users. [3]

A. Mandatory Access Control

The shared databases required stringent policies for the users to get clearance in accessing data. It is based on system wide policies that cannot be changed by individual users. Classified data could not be accessed by a user with a simple access right since the data could be highly sensitive to users based on the privilege of the users. No commercial system had implemented this model successfully in multi-level databases and research was still on. A mandatory security policy is considered to be any security policy where the definition of the policy logic and the assignment of security attributes are tightly controlled by a system security policy administrator. MAC enforces access control on the basis of the labels assigned to subjects and objects. Mandatory policies are based on the principles established in the Bell-La Padula model. This model is based on the subject-object paradigm. [4]

B. Discretionary Access Control

Access Matrix Model: The access rights (read, write, modify, delete, etc) were driven by this model which provides the permission for the subjects to act on an object (relations, views, attributes, etc). The rows represent subjects while columns represent objects and the entries represent access rights. Implementations based on this include authorization lists and capability lists. This model served as a powerful tool for implementing, studying and comparing access rights. SQL supports discretionary access control through the GRANT and REVOKE commands. GRANT command gives user privileges to base tables and views. REVOKE is complementary command to GRANT that allows the withdrawal of privileges. [5]
While early work on integration database with mandatory and discretional access control produced break though such as System R Authorization Model, it was too rigid and suited for closed and controlled environments. The research of these models still continued. Mandatory access provided a higher level of security.
During the period of 1991-2000, the development of distributed object systems and e-commerce applications resulted in developments or object oriented access control. In fact, most of access control research at this time centered on role based and object oriented issues.

C. Role Based Access Control

Role Based Access Control was often better suited for organizations than mandatory or discretionarily access control. RBAC is a type of discretionary access control where authorization administration is simplified around

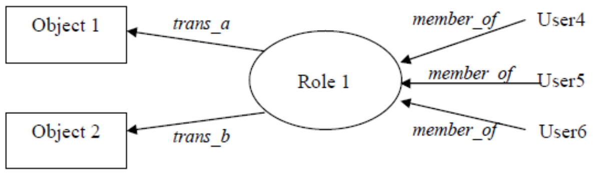

Figure 2. Role Relationship

the notion of a role. Authorizations are given as need arises for an individual to perform a certain activity, rather than granted directly to user. Users are made up for roles thereby acquiring these authorizations. RBAC is centered on roles not users, thereby preventing a single user from receiving excessive authorizations. [6]

IV. USER IDENTIFICATION

Users accessing a database must first connect to DBMS and then establish a session with the database server. Some systems, for example, Oracle, Sybase, and SQLBase, provide a complete identification and authentication mechanism requiring every user connecting to the database to first identify himself to the database server with some assigned user identifier and then provide his password for authentication. Other DBMS such as Ingres and Informix, leave the authentication task to the operating system. [7]
User authentication is the process of confirming a user identity. Weak authentication schemes allow attackers to assume the identity of legitimate database users by stealing or otherwise obtaining login credentials. An attacker may employ any number of strategies to obtain credentials. [7]

A. Brute Force

The attacker repeatedly enters username/password combinations until he finds one that works. The brute force process may involve simple guesswork or systematic enumeration of all possible username/password combinations. Often an attacker will use automated programs to accelerate the brute force process.

B. Social Engineering

A scheme in which the attacker takes advantage the natural human tendency to trust in order to convince others to provide their login credentials. For example, an attacker may present himself via phone as an IT manager and request login credentials for “system maintenance” purposes.

C. Direct Credential Theft

An attacker may steal login credentials by copying post-it notes, password files, etc. The strongest practical authentication technologies and policies should be implemented. Two-factor authentication (tokens, certificates, biometrics, etc.) are preferable whenever possible. Unfortunately, cost and ease of use issues often make two-factor authentication impractical. In such cases, strong username/password policy (minimum length, character diversity, obscurity, etc) should be enforced. Failed Login Detection optionally enforces a failed database login threshold (count and timeframe) to prevent brute force attacks. [7]

V. ENCRYPTION

Though access control model were developed and found to ensure security, there were always chances of those access controls to be bypassed leading to a breach. To enforce the second layer of security, data being stored in the repository could be modified and stored in an encrypted format. The cryptographic technique of using keys to encrypt and store data was applied to achieve security.

It was realized early that whichever access control the database contained the adversary still had a way accessing the underlying data by looking directly in the file system or underlying storage. However, for a long time cryptographic capabilities required a heavy cost burden for database operations and were not included in any implementations. With increases in processing speed supporting encryption of stored data became feasible in early 2000.

A. Database-Level Encryption

Database-level encryption allows enterprises to secure data as it is written to and read from a database. This type of deployment is typically done at the column level within a database table and, if coupled with database security and access controls, can prevent theft of critical data. Database-level encryption protects the data within the DBMS and also protects against a wide range of threats, including storage media theft, well known storage attacks, database-level attacks, and malicious DBAs. Since the encryption/decryption only occurs within the database, this solution does not require an enterprise to understand or discover the access characteristics of applications to the data that is encrypted. enterprises must adopt an approach to encrypting only sensitive fields. The primary vulnerability of this type of encryption is that it does not protect against application-level attacks as the encryption function is strictly implemented within the DBMS. [8]

B. Storage-Level Encryption

Storage-level encryption enables enterprises to encrypt data at the storage subsystem, either at the file level (NAS/DAS) or at the block level SAN. This type of encryption is well suited for encrypting files, directories, storage blocks, and tape media. Current storage security mechanisms only provide block-level encryption; they do not give the enterprise the ability to encrypt data within an application or database at the field level. Consequently, one can encrypt an entire database, but not specific information housed within the database. [8]

C. Encryption Algorithms

Initially we considered several encryption algorithms AES, RSA and Blowfish for the implementation. It is found that the performance and security of the AES algorithm is better than the RSA implementation and the Blowfish algorithm implementation. AES is fast, compared to other well-known encryption algorithms such as DES. DES is a 64-bit block cipher, which mens that data is encrypted and decrypted in 64-bit chunks. This has implication on short data. Even 8-bit data, when encrypted by the algorithm will result in 64 bits. The AES implementation was registered into the database as a user defined function. Once it was registered, it could be used to encrypt the data in one or more fields - whenever data was inserted into the chosen fields, the values are encrypted before being stored. For hardware level encryption, DES algorithm is the option. [9]

D. Encryption Key Management

One of the essential components of encryption that is often overlooked is key management - the way cryptographic keys are generated and managed throughout their life. Because cryptography is based on keys that encrypt and decrypt data, your database protection solution is only as good as the protection of your keys. Security depends on two factors: where the keys are stored and who has access to them.

When evaluating a data privacy solution, it is essential to include the ability to securely generate and manage keys. This can often be achieved by centralizing all key management tasks on a single platform, and effectively automating administrative key management tasks, providing both operational efficiency and reduced management costs. Data privacy solutions should also include an automated and secure mechanism for key rotation, replication, and backup. [9]

Current commercial RDBMSs support many different kinds of identification and authentication methods, password-based authentication, host-based authentication, PKI (Public Key Infrastructure) based authentication, third party-based authentications such as Kerberos, DCE (Distributed Computing Environment) and smart cards. Essentially, all methods rely on a secret known only to the connecting user. It is vital that a user should have total control over her/his own secret. [9].

VI. CONTROL METHODS

A. Designing Security Plan

Data security plans must be designed to match the requirements of each unique project. Not all projects may require a complex security plan. The first step for any researcher who is collecting data is to conduct an assessment of data security needs, which includes determining confidentiality risks, potential harm from breaches of confidentiality and the likely demand for use of the data. [10]

B. Prevention From SQL Injection

SQL injection is a technique often used to attack databases through a website. This is done by including portions of SQL statements in a web form entry field in an attempt to get the website to pass a newly formed rogue SQL command to the database (e.g. dump the database contents to the attacker). SQL injection is a code injection technique that exploits security vulnerability in a website's software. The vulnerability happens when user input is either incorrectly filtered for string literal escape characters embedded in SQL statements or user input is not strongly typed and unexpectedly executed. SQL commands are thus injected from the web form into the database of an application (like queries) to change the database content or dump the database information like credit card or passwords to the attacker. SQL injection is mostly known as an attack vector for websites but can be used to attack any type of SQL database. [11]

There are a number of things that database administrators should do to limit risk and respond to possible incidents on the code and infrastructure they manage:

Review IIS logs and database tables for signs of previous exploits. Since this exploit takes place via the URI query string, administrators can review IIS logs to find anomalous queries that may be attempts to exploit this.

If IIS logs show that the server has possibly been exploited, the next step would be to inspect tables in databases that are used by the associated web applications, looking for <script> tags appended to cells in text columns.

NOTE: IIS servers should never run in production with logging disabled. While the storage and administration requirements of IIS logging can be significant, the lack of IIS logs makes it very difficult to respond to security incidents.

If running 3rd party code that uses a database back-end, consult ISV about susceptibility to SQL injection

In cases where 3rd party ASP web applications are being used, administrators should contact the application vendors to ensure that they are not susceptible to SQL injection attacks. Validate that the account(s) that are used from the web application have least possible privilege in the database

Administrators should make sure that the SQL users that the web application uses have the least privilege necessary. Web applications should never connect as users with administrative privilege such as “sysadmin” at the server level or “db_owner” at the database level. [12]

VII. CONCLUSION

In this paper we briefly reviewed the various database security trends from its inception to the recent. This would be helpful to focus on the various issues of database security. The classical methods of ensuring database security, the partition of database, cards, user authentications, encryption, etc. are able mostly to accomplish their tasks.

Databases are a favourite target for attackers because of the data these are containing and also because of their volume. Efforts to ensure database security are considerably higher than for the other types of data. It is easier to implement an access list for a great number of files than an access list for the elements of a database.

In this paper we have covered two types of access control policies. Mandatory access control inspired by the Bell-La Padula model, govern the access by user to data on the basis of classification assigned to them. Discretionary access control is based on the concept of access rights or privileges and mechanisms for users such privileges. While Mandatory Access Controls (MAC) is appropriate for multilevel secure military applications, Discretionary Access Controls (DAC) is often perceived as meeting the security processing needs of industry and civilian government. Another type of non-discretionary access control - role-based access control (RBAC) -that is more central to the secure processing needs of non-military systems than DAC

The user identification is based on user name and password and access can be controlled by assigning various privileges to the users. User passwords should be protected or stored in the personal configuration files.

On the other hand, to protect the data within an information system, a reliable encryption, such as CrypTIM, RSA, etc. has to be implemented on all information system layers and within a DBMS.

VIII. FUTURE SCOPE

Information Technology security and data protection problems are a real challenge to the current system development. Database security presents features that must be seriously taken into account.

An active area of research in discretionary access control is represented by extension to the expressive power of authorization model. The goal is to provide extended functionalities to direct support to a large variety of application security policies. There are many advanced application such as CSCW or CAD with authorization requirements different than those of traditional applications. Two notable extensions in this direction are represented by temporal authorization model and sophisticated role models

Work on finding order preserving encryption for different data types is an active area of research. Performing geometry encryption in such a way that operands such as intersect can be conducted on encrypted data may be the subject of future research.

Furthermore a disadvantage of any security and data protection system is its low performance, which has to be investigated and monitored at all the levels of a system design and development.

REFERENCES

[1] Denning D.E., *Database-Security Annual Review Of Computer Science*. 1988; 3: 1-22

[2] Denning, D. E., Denning, P. J., Schwartz, M.D. 1979. *The tracker: a threat to statistical database security*. pp. 76-96

[3] Trueblood, R. P., Hartson, H. R., Martin, J. J. 1983. MULTISAFE: *A Modular Multiprocessing Approach to Secure Database Management*. Pp. 382-409

[4] Mandatory Access Control, Available: http://imsciences.edu.pk/serg/2010/07/mandatory-access-control/

[5] Jain S, Ruchita R, *Database Management System*, Pune, Tech-Max Publications, August 2007, pp. 12-5

[6] R.W. Baldwin. Naming and Grouping Privileges to Simplify

Security Management in Large Databases. In *IEEE Symposium on Computer Security and Privacy*, 1990.

[7] User Identification, Available: http://www.imperva.com/resources/webinars.asp

[8] R. L. Rivest, A. Shamir, and L. M. Adleman. *A method for obtaining digital signatures and public key cryptosystems*. Communications of the ACM, 21(2):120–126, 1978.

[9] D. E. Denning. *Cryptography and Data Security*. Addison-Wesley Publishing Company, Inc., 1982.

[10] Security Plan, Available: http://www.nichd.nih.gov/publications/pubs/upload/data_security.pdf

[11] SQL Injection, Available: http://en.wikipedia.org/wiki/SQL_injection

[12] Prevention From SQL Injection, Available: http://blogs.technet.com/b/srd/archive/2008/05/29/sql-injection-attack.aspx