Database Security – Encryption

Jack Webb

ICTN 6865

Database Security – Encryption

I. Abstract

The paper will cover the importance of databases and why their contents should be secure. Databases are a necessity in today's corporate environment. Even though databases are important to business productivity, their security in most cases are still considered second to other areas. The paper will cover the initial concepts of database. In order to properly place security controls, we must understand the necessary risks involved with improper database security. It is also necessary to cover the types of threats and attacks that can be launched to gain access to or disrupt access to a database. As it would be a long process to cover all possible controls to securing databases, it better to just cover a single concept in database security such as encryption.

II. Introduction

Today's society has a thirst for technology advancements. These advancements make everyday life more manageable and easier. Technology is used for a variety of daily tasks such as entertainment, banking, medical and business endeavors to name a few. The problem with relying all of these technologies is their vulnerabilities. We all want to believe that with every advancement security is a consideration and as advanced as these technologies are security isn't always up to the task. For every advancement, security measures are taken to harden the devices, process or solutions. No matter how through the security measures are there is always someone or a group of individuals looking for ways to break or get around these security measures. As our technologies become more advanced, they start working against us as these technologies make it much easier to break security processes. Information security must be an ever evolving processes as one layer is compromised after another. When looking at what security measure to place most look at securing hardware, networks and systems. Many of today's organizations live and die by the database as they store personal data that is necessary for progress.

Generally, databases are looked at as needing to be secure but more effort is placed in other areas leaving databases exposed to possible attacks (Kirsch). In order to discuss the security of databases, we must first look at the reasons they should be secured such as their vulnerabilities and methods that can be put in place. Every option for database security can't covered; however, database encryption is very important to be discussed.

III. **Why Database Security?**

Todays, databases are essential to everyday business. Databases hold various types of data that is used for day to day business. The data can range from private information, financial data to public information. This data is collected or created by various means such as data entered in web site forums and data entered by business employees. The loss of this data can cause delays to crippling productivity. Data can be lost due to various means such as software errors, hard drive malfunctions and human mistakes. However, there are other means for data loss and that is through database attacks from both outside and inside the trusted network. These threats to database security is ever increasing as databases are becoming more accessible due to the internet and being used with web-based applications (Murray). To understand security, we first must understand the vulnerabilities or threats to databases.

Databases are difficult to secure as there are quite a few vulnerabilities to deal with. Several security issues arise due to design and management situations. One such example occurs during the deployment of the database software. During deployment developers may fail to ensure the database isn't performing actions that it shouldn't. These extra activities could be a vice for attackers to exploit. Data leaks could also be a vulnerability that could lead to attacks or provide information that an attacker could utilize. Just like most software, databases may have a network ability to pass network traffic. Because of this, attackers could intercept this traffic to gain information and initial an attack. If the database has a network feature it should be disabled as well as other features of the database. As with

ports on a router or switch, if the feature isn't going to be used it needs to be disabled. Why provide an avenue for attackers to exploit.

Management inconsistencies can also create vulnerabilities. When databases are deployed it's necessary for administrators to remove default entries such as usernames and passwords (Chickowski). Default, blank or weak usernames and passwords are liabilities to the security of a database. As we know when it comes to security management, the least privileged assigned the better. This is no different when it comes to databases. However, providing higher privileges to users and groups could lead to vulnerabilities from within. Another management flaw is to leave encryption keys on the same disks as the database. This places the encrypted keys in a vulnerable position open to possible attacks. The encryption keys can be separate from the database (Osborne). Storing database data in clear text is also another vulnerability that an attacker can exploit. Databases should be encrypted. This goes as well with the data as it's moved from one location to another. Meaning, as the data moves from let's say as web interface to the database, the data should be encrypted throughout the whole process.

The next three vulnerabilities could be considered the top three when it comes to database vulnerabilities. These are SQL injections, buffer overflow and denial-of-service. If a database isn't able to properly sanitize input data to weed out real data from false, attackers can input SQL data strings that could eventually be processed and grant the attacker advanced privileges or access to other functions (Chickowski). Next, buffer overflows can also create havoc with database access. This can occur when more data is inserted in an application than it is setup to receive. If a name box expects to receive 20 characters and an attacker enters much more than is expect, there could be a corruption of data or over flow to other buffers that could lead to data being over written. The last covered vulnerability is denial-of-service. Denial-of-service is the process of denying service or access in this case to the database server or application. This attack relies on known vulnerabilities associated with the server software or the database software. The denial-of-service doesn't have to be just associated with the database to

cause an effect. As we know operating systems also have vulnerabilities that an attacker can exploit to create a denial-of-service situation (Anuramn). This is in turn will affect the database.

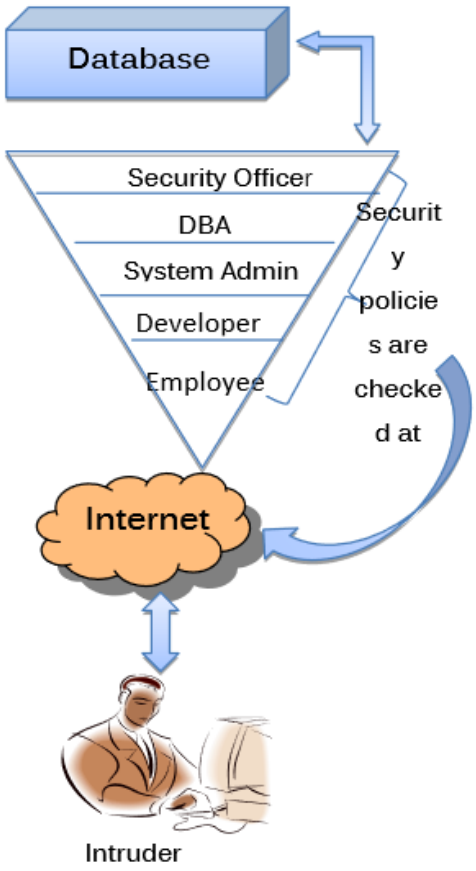


Fig 1: Security layer at organizational level

There are methods that can combat or fix the listed attacks or vulnerabilities. When it comes to database security a well define security policy must cover several critical area such as those found in figure 1 (Igra Basharat). Each level must address or be a player is security of databases and their servers or an attacker will be able to exploit a vulnerability. There are about six critical areas when considering database security. Figure 2 shows the critical areas that must be covered. These area are access control, user identification and authentication, inference policy, accountability and auditing, and encryption.

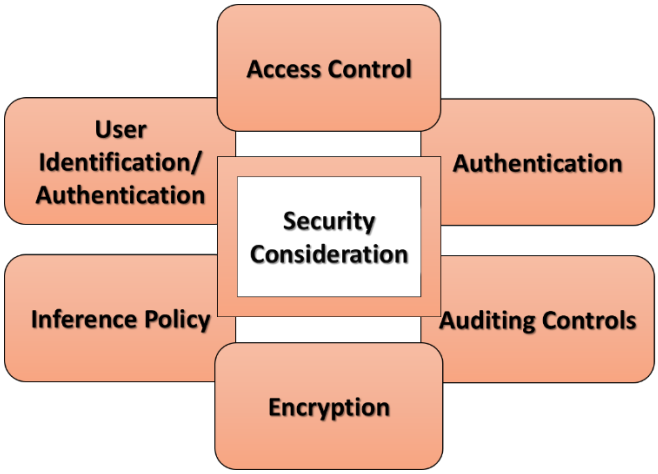


Figure 2. Security Considerations

Access control is probably most important of the six as it determines the accessibility of not only the database but the server as well. With proper access control, attack can be avoided as well as preventing possible error that may occur during daily use (Igra Basharat). Improper access control configuration can lead to vulnerabilities and allow access to those that have no need. Along with access control, user identification and authentication is very important. This is in line with confidentiality, which is a property of information security. Allow access to only those that need access. Identifying and authenticating those that need access is necessary while denying others. There are several levels in regards to access control. When looking at access control one must consider perimeter controls such as those in routers, user access, application system access and privileged user access (Charles Le Grand). Each level implements a certain level of access control that further protects data. Inference refers to a technique that can be used to defeat access control methods. An in place policy dictates how the database data is to be protected. Inference occurs when a user accesses data they can access to infer data they aren't supposed to be able to access (Yip). To assist in database security, a well-defined and utilized accountability and auditing process must be in place. They can be utilized to identify who and when an individual gained access to objects in a database. A collection of this data can be used to detect patterns and used to identify breaches (Murray). Last but not least, encryption of the database and

transmissions must also be used to ensure the safety of security. This is the process of taking the data and making it unreadable to those without proper access and holding encryption keys that are used to encrypt and decrypt the data. Encryption occurs at multiple levels to ensure the data used in these database remain concealed. This must happen at server level, database level and application level (Igra Basharat). Figure 3 provides a view into a multi-level encryption environment. As it is possible to cover each one of these vulnerability solutions, it is easier to just look at one solution in order to cover it in more details. In this case, encryption would be a good area to cover in detail.

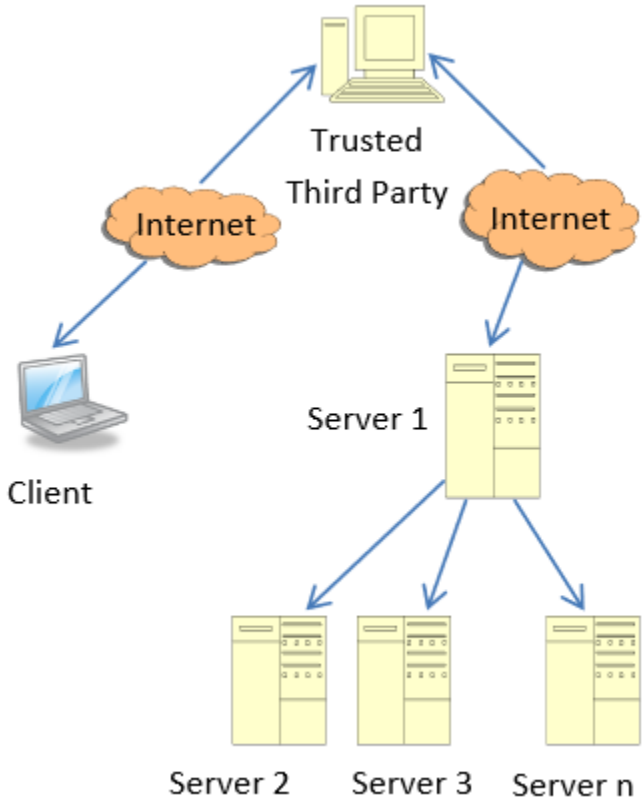


Figure 3. Levels of Encryption

IV. Key Management

Even though encryption is seen as a security positive, it can also create problems that administrators must deal with. As discussed earlier, the utilization of encryption require the use of keys

to encrypt and decrypt data. In business environments where there are many databases to manage, encrypting all of these databases require a multitude of keys. This makes key management a serious issue that must be dealt with (Kirsch). Mismanagement of encryption keys could mean serious problems if the keys aren't available for users to have access to to gain access to the data they need. This is also true with losing keys. Losing keys can also prevent access to data that was encrypted by that key. Key management issues can also arise if keys are stored within the database. If a database if compromised and keys are stored there, then an attacker may also have access to the encryption keys. Encryption keys should be stored separate from the database or at least hide the keys within the database code under a name that isn't obvious (Database Encryption for Application Development). Key management is obviously one of the most important objectives when encryption is implemented. Key management can now be automated through the use of various software packages such as Vormetric Key Management. This software package provides a centralized method of managing keys in a repository (Vormetric Data Security Products Enterprise Key Management). To go hide by hide with key management, keeping track of management activities is a must. Administrators must create audit trails in key management. These audit trails will allow administrators to determine whether keys are in need of destruction. Why is this important? If it is necessary to destroy database data or the database entirely, it may not be possible as some data lingers after destruction. This is where key activities or auditing comes into play. Encrypted data can be destroyed if the encryption key/s is destroyed (Kirsch). Destroying the key saves the drive that have had to be destroyed. However, destroying just a single key isn't the full solution. It is also necessary to destroy all keys that were associated with the select database. This includes the key/s that may have been backed up. Destroying just one key doesn't make for destroyed data.

V. Database Encryption

As mentioned, encryption is the process of making plan text data or readable data unreadable except by those that hold the key to decrypt the data (Igra Basharat). The type of encryption algorithm that is use will decide the type and number of keys that are needed in the encryption and decryption of data. In a symmetric environment a single key or private key is used to encrypt and decrypt the data. In an asymmetric environment, there are two different keys used to complete the same for mentioned process. One key for encryption and a second key for decryption. This is call public key encryption. As attackers are methodically breaching through the different layers of information security meant to protect internal resources, databases and their data are consistently under attack. Many see encryption as the last line in solving or mitigation security issues with databases (Kirsch). Implementing database security is just something some decides on doing. Deciding whether to implement encryption involves a similar process as creating the database itself. There has to be a plan and an understanding on what and how to encrypt. Encryption is a drawn out process that can have drastic implications if done wrong (Reis). So, just deciding one day that encryption must be implemented with a plan could cause access problems and performance issues. Another issue that an administrator must look at is does the data need to be encrypted. If other measure that are in place are adequately getting the job done, then encryption my not be necessary. Of course, there are situations where encryption is not only necessary but the law. One such mandatory requirement is California's SB1386 law, which requires the utmost data protect (SB 1386 and AB 1298 Guideline).

As mentioned encryption is a multi-level process that comprises storage-level, database-level and application-level encryption. Storage level encryption is the process for methodology of encryption the data within the storage system, essentially within the server or workstation. Where ever the database is store. Storage level encryption requires that all aspects of data within the subsystem is encrypted. So, there can't be any unencrypted portions of the system such as a backup. Selectively encrypting single files such as temporary files or log files can be unsafe (Igra Basharat). Database level

encryption is the encryption of data within the database. This encryption can be within specific cells, row, columns, tables or the database as a whole. This encryption covers both saved data and data that is recovered from the database (Igra Basharat). The last level of encryption is within the application itself. This is necessary as the application as access to the database and data.

The encryption of data in transit between servers and clients can be accomplished through several means. Not all databases are stored on the same servers or workstations where the data is created. One example is data created or collected on web servers in an ecommerce environment. The entered or collected data may be routed to a database on a server that is located within the organization's trusted network. Encrypting this traffic keeps prying eyes of the data if an attacker is probing for information. Several of the encryption methods include Secure Socket Layer (SSL) encryption, Secure Shell Layer (SSH), Internet Protocol Security (IPSec) and a transport method from Oracle called Oracle Advanced Security, which is obviously proprietary. All of these methods provide an encryption method of getting the data to the database.

So, it has been decided that encryption is necessary. The question at this point would be, which encryption method to use. In some cases, deciding on which encryption method to use has been taken out of the consumers hands. Some database packages have encryption as part of the package and in some case the only option. As it isn't possible to look at every encryption option available, several will be covered providing a view into database encryption. Table 1 provides data on encryption methods, used algorithms and where the encryption is implemented. In table 2, a comparison is made between the different encryption methods.

Table 1. Database Encryption

Paper	Methods/Techniques	Algorithm	Where encryption can be performed
A Novel Framework for Database Security based on Mixed Cryptography [6]	Mixed Cryptography Technique based on data classification methods	Any symmetric Encryption algorithm can be used	Encryption is done at • Client side • Untrusted database • Server
Database Encryption [7]	Hash Security Module Encryption Strategy	State –of-the art algorithm and mode of operation should be used.	Encryption can be at: • Storage Level • Database Level • Application Level
A Database Encryption Scheme for Enhanced Security and Easy Sharing [9]	Combination of the conventional encryption and public key encryption, utilizing the speed of conventional encryption and convenience of public key encryption.	X	X
Transparent Data Encryption-Solution for Security of Database Contents [10]	Transparent Data Encryption used by Master database key	X	Page level
Fast, Secure Encryption for Indexing in a Column-Oriented DBMS [11]	Fast Comparison Encryption	Symmetric encryption algorithm	DataWare houses

Table 2. Encryption method comparison

Methods/Techniques	Advantages	Disadvantages/ limitations
Mixed Cryptography Technique based on data classification methods	Sensitive data is protected from attacks even at multiple levels because of having many keys to different parties. Secure data storage and data transmission is performed to ensure the maximum protection of sensitive data.	Performance of queries and security analysis is affected because of encryption algorithms. Access control methods are not defined.
Hash Security Module Encryption Strategy	Security server is not tampered Encryption keys are never exposed.	Complex
Transparent Data Encryption used by Master database key	Provides protection to sensitive data on disk drives and backup media from illegal access. Cost of user management is reduced. Provide privacy management.	Encryption across communication channels is not provided. Database could not be opened if the certificate is not available and the backup of certificate and private key is not maintained. Database becomes inaccessible after altering the certificates to be password protected.
Fast Comparison Encryption	Fast indexing operation Low decryption overhead	

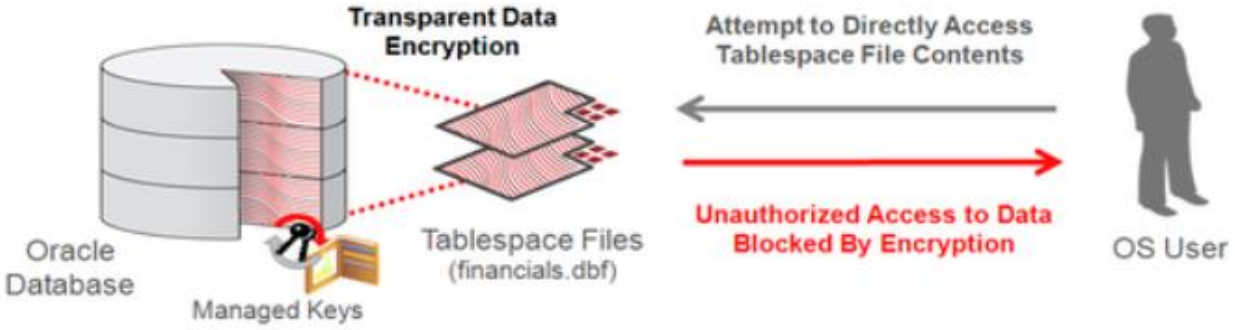
Since there are a few encryption methods several of the more utilized ones will be covered. Oracle provides several database packages that can somewhat be considered an all in one. Meaning, not only does the package come with a database but includes security features such as encryption. Depending on the chosen package, depends on what type of encryption method that is provided. Of course, not all versions of the Oracle packages support the most current encryption methods. For example Oracle 8i only supports the encryption method of DES where more current versions may only support 3DES, Twofish or Blowfish (Database Encryption for Application Development). However, the most current packages supports transparent data encryption with Advanced Encryption Standards (AES) and Triple Data Encryption Standard (3DES) (Huey).

VI. Transparent Data Encryption (TDE)

A common encryption technology used in database security is Transparent Data Encryption (TDE). TDE is used with both Oracle and SQL server products. The primary purpose of TDE is to stop would be attackers from getting pass the database and reading sensitive information by enforcing data at rest encryption (Transparent Data Encryption). This technology provides the strongest encryption method while providing key management and support. Figure 4 shows a brief overview of the TDE process in regards to Oracle database packages. As mentioned earlier, when it comes to the encryption of databases and their data, it is necessary to encrypt backups as well and TDE takes on this process by ensuring that database backups are encrypted to ensure there will not be a loss of data at either storage location (Otey). Depending on the package to be encrypted, each may require a slightly different process. SQL Server is pretty much straight forward. It requires the creation of a master key for the targeted database, a certificate to protect the key, a special key called the database encryption key used to protect the database, which is then secured through the use of a certificate as well, and then finally the encryption of the database (Otey). The Oracle process is just slightly different. Oracle uses what's called a wallet to store the master encryption key. This is not located within the database but at the

operating system. So, Oracles process requires the creation of the wallet and its location. The creation of the data encryption key and then the encryption of the database and data (Jeloka).

Figure 4. Oracle and TDE



VII. Conclusion

Many companies rely heavily of collected data whether it be from within the trusted network or collected data from outside sources such as personal data from customers. Each organization is full responsible for how they protect that data from those that have no need for access. This access could be from within the organization or those that are looking to gain access from outside such as hackers. The loss of such data could be detrimental to not only the operation of the organization but the loss of confidence from the organization's employees, business associates and customer/consumers. In order to prevent such events, organizations that collect and store important data must enforce strict security measures. When it comes down to it encryption is the last line of defense in the protection of data whether it's data in transport or at rest. Several encryption methods were covered but what it comes down to is that the database package may dictate with method as not all are compatible with every method. In the end in this day and age with all the breaches and losses of data, database encryption seems be on the rise and becoming a necessity to combat attackers.

Work Cited

Anuramn. *Threats to Database Security*. Ed. Lamar Stonecypher. 1 Jan. 2010. 22 Nov. 2013. <http://www.brighthub.com/computing/smb-security/articles/61554.aspx>.

Charles Le Grand, Dan Sarel. *Database Security, Compliance and Audit*. 2008. 20 Nov. 2013. <http://www.isaca.org/Journal/Past-Issues/2008/Volume-5/Pages/Database-Security-Compliance-and-Audit1.aspx>.

Chickowski, Ericka. *The 10 Most Common Database Vulnerabilities*. 1 Nov 2010. 22 Nov 2013. <http://www.darkreading.com/vulnerability/the-10-most-common-database-vulnerabilit/228000482>.

Database Encryption for Application Development. n.d. 21 Nov. 2013. <http://www.cisco.com/web/about/security/intelligence/db-encryption.html#6>.

Huey, Patricia. *Oracle Database 2 Day + Security Guide 11g Release 2*. Sept. 2012. 20 Nov. 2013. <http://docs.oracle.com/cd/E11882_01/server.112/e10575.pdf>.

Igra Basharat, Farooque Azam, Abdul Wahab Muzaffar. *Database Security and Encryption: A Survey Study*. June 2012. 22 Nov. 2013. <http://research.ijcaonline.org/volume47/number12/pxc3880218.pdf?vm=r>.

Jeloka, Sumit. *Oracle Database Advanced Security Administrator's Guide 10g Release 2*. June 2012. 23 Nov. 2013. <http://docs.oracle.com/cd/B19306_01/network.102/b14268.pdf>.

Kirsch, Christian. *The role of encryption in database security*. 13 May 2009. 21 Nov. 2013. <http://www.net-security.org/article.php?id=1232&p=2?vm=r>.

Murray, Meg Coffin. *Database Security: What Students Need to Know*. 2010. 29 10 2013. <http://www.jite.org/documents/Vol9/JITEv9IIPp061-077Murray804.pdf>.

Osborne, Charlie. *The top ten most common databse security vulnerabilities*. 26 June 2013. 20 Nov. 2013.

Otey, Michael. *Using Transparent Data Encryption*. 26 June 2012. 23 Nov. 2013. <http://sqlmag.com/database-security/using-transparent-data-encryption?vm=r>.

Reis, Jorge. *What is Database Encryption?* 16 Jan. 2012. 21 Nov. 2013. <http://www.databaseguides.com/what-is-database-encryption>.

SB 1386 and AB 1298 Guideline. Sept. 2002. 24 Nov. 2013. <http://www.oit.ucsb.edu/committees/itpg/sb1386.asp>.

Transparent Data Encryption. n.d. 22 Nov. 2013. <http://www.oracle.com/technetwork/database/options/advanced-security/index-099011.html?vm=r>.

Vormetric Data Security Products Enterprise Key Management. n.d. 22 Nov. 2013. <http://www.vormetric.com/products/enterprise-key-management>.

Yip, Raymond Wai-Man. *A Data Level Database Inference Detection System*. 1998. 22 Nov. 2013. <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.9.4549&rep=rep1&type=pdf>.