|  |
| --- |
|  |
| **Can security be determined for Database Management Systems based on its number and severity of vulnerabilities?** |
|  |
| Economics of Ciber Security |
| **Alma Luz Valencia López [4514882]** |
|  |

|  |
| --- |
|  |

# Abstract

A vulnerability is a weakness in a product that could allow an attacker to compromise the integrity, availability, or confidentiality of that product, that’s why is important to know and understand how security is determined for such products, especially when they store one of the most important assets in the everyday life for both individuals and companies: Data.

Impact and number of vulnerabilities are being compare to have a better insight about “how secure a system is” and to identify if this two variables are needed to determine the less vulnerable database management system among Oracle, IBM and My SQL Server.

# Introduction

Data is the most important asset for individuals and companies since it is used every day in every kind of activities and transactions. This data is commonly stored into databases that need to secure information based on the level of data importance; this data will be structured and generally stored in the form of relational tables to relate data to data.

The data will be managed by a Database Management System (DBMS) defined as a set of applications which help managing data for better performance and faster retrieval by maintaining indices, also preserves logs of transactions which helps recovering data and performs the function of concurrency control.

DBMS can be vulnerable in many ways mainly because such their databases are interfaced with other applications, giving a second entry to reach the data, but it can also happen that the database users leak information to the outside world without knowing.

The main consequences are threats against:

* Confidentiality: Since databases generally hold sensitive data, loss of confidentiality is a frequent problem with database vulnerabilities.
* Authentication: If poor SQL commands are used to check user names and passwords, it may be possible to connect to a system as another user with no previous knowledge of the password.
* Authorization: If authorization information is held in a database, it may be possible to change this information through the successful exploitation of the vulnerability.
* Integrity: Just as it may be possible to read sensitive information, it is also possible to make changes or even delete this information.

DBMS are highly susceptible to vulnerabilities because, most of the exploits and attacks are independent of the platform of the interface the database is connected to, or they are generic for the management system, it does not mean that all of them are equally insecure (or equally secure). Some systems are attacked more because they are more popular among the enterprises or because the information they hold are more sensitive and can generate a bigger gain for the attacker.

# Literature Review

The impact of database attacks may vary from gathering of sensitive data to manipulating database information, and from executing system-level commands to denial of service of the application. The impact also depends on the database (the data it holds), the target machine and the roles, and privileges the attack runs with.

According to Burtescu [1] the focus of attacks on the company’s databases are motivated by the following factors:

* Databases are the mass of information which the company works with;
* Databases can reveal private data by processing public data.

Trinidad [2] points out organizations have long focused their security efforts on the perimeter and endpoints, leaving the datacenter highly vulnerable to anyone gaining access via:

* SQL Injection
* Spear phishing
* Malware / Drive by downloads
* Advanced Persistent Threats
* Insider attacks / Mistakes
* Social engineering

And he identifies the following database vulnerabilities:

* Default and Weak passwords
* SQL injection in the DBMS
* Excessive user and group privileges
* Unnecessary enabled DBMS features
* Broken configuration management
* Buffer Overflows
* Privilege Escalation
* DoS
* Unpatched database
* Unencrypted Data – At rest and In Motion

But if we refer to the DBMS vulnerabilities, Fisch [3] identifies the next four types of vulnerabilities:

* **Inference:** Allows a user to deduce the contents of the inaccessible parts of the database.
* **Aggregation:** Like inference, the damages that results from aggregation are only information dissemination, not database modification. Aggregation is the process of combining multiple database objects into one object with a higher security label than the constituent parts.
* **Data Integrity:** Damage to data integrity can often cause more serious problems than confidentiality breaches: important information may be modified / deleted by the attacker
* **Trojan Horses:** Are the first stage of an attack and their primary purpose is to stay hidden while downloading and installing a stronger threat such as a bot, to perform actions to the DBMS.

The presence of inference or aggregation vulnerabilities leads to the undesired release of normally secure, or hidden, information contained in the database. Data integrity vulnerabilities undermine the correctness of the information within the database. Trojan horse vulnerabilities allow hidden operations to perform unauthorized action in, and to, the DBMS

Rohilla et all [4], identify the following vulnerabilities that partially agree with the ones presented by Trinidad, and actually extend them. These are:

* **Excessive Privilege Abuse:** Users (or applications) are granted database privileges that exceed the requirements of their job function.
* **Legitimate Privilege Abuse:** Authorized user misuses their legitimate database privileges for unauthorized purposes.
* **Privilege Elevation:** Attackers convert their access privileges from an ordinary user to those of an administrator.
* **Platform Vulnerabilities:** Vulnerabilities in operating systems and additional services installed on a database server may lead to unauthorized access, data corruption, or denial of service
* **Inference:** A user can draw inference from a database when the user can guess or conclude more sensitive information from the retrieved information from the database or additionally with some prior knowledge
* **SQL Injection:** The attacker typically inserts (or “injects”) unauthorized SQL statements into a vulnerable SQL data channel. Using SQL injection, attackers may gain unrestricted access to an entire database
* **Unpatched DBMS:** In database, as the vulnerabilities are kept changing that are being exploited by attackers, database vendors release patches so that sensitive information in databases remain protected from threats. Once these patches are released they should be patched immediately. If left unpatched, hackers can reverse engineer the patch, or can often find information online on how to exploit the unpatched vulnerabilities, leaving a DBMS even more vulnerable that before the patch was released
* **Unnecessary DBMS Features Enabled:** Database misconfigurations provide weak access points for hackers to bypass authentication methods and gain access to sensitive information
* **Buffer Overflow:** Since buffers contains only a finite amount of data, the extra data - which has to go somewhere - can overflow into adjacent locations, corrupting or overwriting the valid data held in those locations
* **Weak Audit Trails:** A database audit policy ensures automated, timely and proper recording of database transactions.
* **Denial of Service:** In this type of attack all users (including legitimate users) are denied access to data in the database. Denial of service (DOS) conditions may be created via many techniques - many of which are related to the other mentioned vulnerabilities.
* **Covert Channel:** A covert channel is an indirect means of communication in a computer system which can be used to weaken the system's security policy. A program running at a secret level is prevented from writing directly to unclassified data item.
* **Database Communication Protocol Vulnerabilities:** Large number of security weaknesses is being identified in the database communication protocols of all database retailers. Fraudulent activities directing these vulnerabilities can vary from illegal data access to data exploitation and denial of service.
* **Advanced Persistent Threats:** This type threat happens whenever large, well-funded organizations makes highly focused assaults on large stores of critical data. These attacks are relentless, defined, and perpetrated by skilled, motivated, organized, and well-funded groups
* **Insider Mistakes:** Some attacks are not intentional, they just happen unknowingly, by mistake. This type of attack can be called as “unintentional authorized user attack” or insider mistake. It can occur in two situations. The first one is when an authorized user inadvertently accesses sensitive data and mistakenly modifies or deletes the information. The latter can occur accidentally when a user makes an unauthorized copy of sensitive information for the purpose of backup or “taking work home.” Although it is not a malicious act, but the organizational security policies are being violated and results in data residing on a storage device which, if compromised, could lead to an unintentional security breach.
* **Social Engineering:** In this, users unknowingly provide information to an attacker via a web interface like a compromised website or through an email response to what appears to be a legitimate request
* **Weak Authentication:** Weak authentication schemes allow attackers to assume the identity of legitimate database users by stealing or otherwise obtaining login credentials.

Although DBMS vulnerabilities does not belong to the OWASP, SANS, CWE or Cenzic ranking directly as a whole [5], some of them are contained into other categories such as SQLi, Buffer Overflow, Authorization and Authentication vulnerabilities and Server and Application vulnerabilities.

# Research question, Objective and Hypothesis

With so many DBMS vulnerabilities identified that seems to affect all the databases out there, it is normal to ask:

* Which Database Management System has fewer vulnerabilities being exploited?
* How these vulnerabilities are ranked according to the CVSS metric?
* Having more vulnerabilities being exploited means that the DB management system is more insecure?

These questions will help us understand the correlation between the severity and the number of vulnerabilities being exploited.

A system must be call “Insecure” when it has a higher number of exploits with a high severity related. The hypothesis is that most of the vulnerabilities in the Open Source Vulnerability Database related to the Database Management Systems are ranked as low severity and the DBMS is ranked as insecure because of the high number of vulnerabilities being exploited regardless the severity.

# Methodology

Three Database Management Systems will be compared against each other based on the number of vulnerabilities being exploited and its severity according to the CVSSv2 base score. The data will be taken from the Open Source Vulnerability Database (OSVDB) for Oracle, IBM DB2 and MySQL Server management systems since two suppliers have more than the 50% of the European market (Oracle, with a 40.8% market share, IBM with 29.4% [6]) and MySQL is ranked as second position in popularity [7] (just behind Oracle) and all of them have revenue growth over the past years. Microsoft SQL Server was not considered due to the lack of vulnerability information in the OSV database, the Symantec Vulnerability Database [8] or the CVE database.

The vulnerability categories chosen for the analysis try to gather most of the vulnerable aspects in the management systems, these are:

* DoS
* Overflow
* Privilege Escalation
* Memory Corruption, and
* InfoDisclosure

The data extracted covers three years from 2012 to 2014, records without CVSS score were omitted. To be able to determine the severity of the score, the following conversion had to be made, according to the National Vulnerability Database [9] recommendation:

1. Vulnerabilities are labeled "Low" severity if they have a CVSS base score of 0.0-3.9.
2. Vulnerabilities will be labeled "Medium" severity if they have a base CVSS score of 4.0-6.9.
3. Vulnerabilities will be labeled "High" severity if they have a CVSS base score of 7.0-10.0.

The data was counting per year and severity per each one of the categories, resulting in two types of total results, the first one is the total amount of vulnerabilities per year and the second is the amount of vulnerabilities per severity within the three years.

The final outcome will be a descriptive analysis about the records to evaluate the security per each one of the mentioned categories.

# Results

The total amount of vulnerabilities gathered for the three years and the three DBMS are shown in Figure 1, we can observe that Oracle has the fewer amount of records whilst MySQL Server has the higher. We can also see the distribution of the vulnerabilities categories, IBM DB2 is more likely to suffer Memory corruption attacks, in the other hand, Oracle is more prone to InfoDisclosure and MySQL server is more susceptible to DoS.

Figure 1DBMS total vulnerabilities

However, this graph is still raw data since it does not tell anything about the severity of these higher amount of exploits. To measure this aspect the three higher categories (DoS, Memory Corruption and InfoDisclosure) are analyzed.

First, the DoS attack is shown in Figure 2, it can be assumed that because MySQL Server is more attaked is also more insecure, the data reveals that IBM DB2 has a greater amount of vulnerabilities ranked as “High”, and almost the same amount ranked as “Medium” but the difference is significant in the ones ranked as “Low” for MySQL Server.

Figure 2DoS Vulnerabilities

For Memory Corruption, we can observe in Figure 3, that indeed IBM DB2 is susceptible to it and also the severity is “High” when the attack is done. While MySQL Server does not present any of this kind of attacks, Oracle has only one ranked as “Medium”

Figure 3 Memory corruption vulnerabilities

And finally, InfoDisclosure is depicted in Figure 4, where it is observed that the three systems are strong to “High” ranked vulnerabilities but have significantly less exploits than Oracle (making It the less secure in this category). And IBM is slightly better than MySQL Server by having a smaller amount of “Medium” exploits and the same amount of “Low” exploits.

Figure 4InfoDisclosure vulnerabilities

# Limitations

The limitations that come with this kind of topic and that were presented in this assignment are mainly:

* The availability of the information: Most of this information is not made available to the general public as a whole, because it also represents vulnerability if it has not been patched already. Also, most of the databases are not up-to-date and they use different schemas to show information, some of them are very descriptive and provide a lot of information about the records, but they don’t have a high amount of data, whereas other have lots of records but not enough information about them, making the data gathering difficult and incomplete.
* The reliability of it: Most of the data contained in the public vulnerability databases is “White market” data, meaning that they have not being exploited in the wild and they don’t represent actual exploits, they represent vulnerabilities created by companies in a lab with the purpose of warn vendors about them.

# Conclusions

By following the thought that the number of vulnerabilities stablish also the security of a system, MySQL will rank as less secure, IBM will be next and finally Oracle will be declared as the most secure, however the tendency shown in the descripted categories is that security cannot (and should not) be only determined by the number of vulnerabilities found because the severity is not represented, and this needs to be look closely. In this case IBM ranked as less secure as a whole followed by Oracle and finally MySQL, despite the dataset was not equal size, Oracle had 35 records, IBM 80 and MySQL 81. But it needs to be noted that the categories also matter when deciding which system is best, since DBMS have to be chosen according to enterprises’ needs.

# References

1. Burtescu, E. (2009). Database Security-Attacks and Control Methods. *Journal of Applied Quantitative Methods*, *4*(4), 449-454.
2. Mark Trinidad - Top 10 Database Vulnerabilities and Vulnerabilities and Misconfigurations. Application Security Inc. <http://www.sifma.org/uploadedfiles/societies/sifma_internal_auditors_society/top10-database-vulnerabilities-and-misconfigurations.pdf>
3. Fisch, E. A., & White, G. B. (1999). *Secure computers and networks: analysis, design, and implementation*. CRC Press.
4. Rohilla, S., & Mittal, P. K. (2013). Database Security: Threats and Challenges.*International Journal of Advanced Research in Computer Science and Software Engineering*, *3*(5).
5. Kaur, D., & Kaur, P. Ranking and Impact of Web Applications‟ Vulnerabilities.
6. ComputerWeekly – Choosing the right management system. <http://www.computerweekly.com/feature/Choosing-the-right-database-management-system>
7. DB-Engines Ranking - popularity ranking of relational DBMS  
   <http://db-engines.com/en/ranking>
8. Security Focus - <http://www.securityfocus.com/>
9. The National Vulnerability Database - <https://nvd.nist.gov/cvss.cfm>