Afghanistan Research Journal - Natural Science
ISSN (Online): 2789-8601
Volume 4, Issue 1, pp. 1-7, June 2022

Database Security Mechanisms in MySQL

Abdullah Hamidi (MSc)
Database / Information Systems
Computer Science Faculty, Herat University
Herat, Afghanistan

Abdul Razzaq Hamraz (MSc)
Database / Information Systems
Computer Science Faculty, Herat University
Herat, Afghanistan

Khadija Rahmani (MSc)
Software Engineering
Computer Science Faculty, Herat University
Herat, Afghanistan

***Abstract*—** MySQL security is a concept that originates from database security and mainly comprises attacks that exploit database systems vulnerabilities. SQL injection, inference attack, passive attack, active attack, and other database side attacks are general security issues in many modern database systems. Those methods are used by hackers to retrieve, manipulate, misuse, make or delete information in organizations' relational databases through application layer or backend layer. Different techniques to prevent MySQL against these attacks investigated and discussed in this article. Besides, different ways to secure or database were introduced. In this article, different ways to protect the data in relational databases including database backups, database and table locking, database encryption, user control, MySQL Enterprise Firewall, and use of views are discussed. Furthermore, each protection method explained with their usages and advantages. Database designers have to be aware of these methods to increase data protection on their designed information management systems. The goals of this research are to cover all security problems that occur in MySQL backend, declaring the security vulnerabilities and providing suggestions to improve MySQL security and preventing an attacker from attacking these systems.

***Keywords*—** relational, MySQL, database, security, backup, locking, views, firewall

I. Introduction

In the world of information technology, data security is one of the vital subjects, which these data define the worth of organizations, so by the growth of technology organizations store their data in so-called databases. "Database technologies are a core component of many computing systems. They allow data to be retained and shared electronically and the amount of data contained in these systems continues to grow at a rate" [1]. So, we need to ensure the integrity, Confidentially, Availability and Authenticity of the data and secure the data from unintended access which is called database security, Database security is the protection of an invaluable organizational resource against unauthorized reading, changing or erasing of the data. "Database security strives to ensure that only authenticated users perform authorized activities at authorized times" [1]. "It includes the system, processes, and procedures that protect a database from unintended activity" [14]. Database security in MySQL is one of the most important topics that have been discussed among security personnel. "The growing number of incidents proves that it's something that should be taken care of immediately. Database security should provide controlled and protected access to the members and also should preserve the overall quality of the data" [15]. It's very important to understand the structure of the database and identify potential threats at the beginning stages. These points by Oracle should be considered when securing a database (2016):

- Protecting data from unauthorized access.
- Preventing unauthorized disclosure.
- Recovering from hardware or software errors.

II. Goals and Importance

Nowadays, people trying to use technology according to their needs to gain the advantage of this unlimited weapon. One of these valuable components of technology is database systems which are changing the work outcomes and producing effective results.

This technology has been tested in most of developed and developing countries, and the expected result has been achieved. Moreover, from understanding of its effectiveness. Nowadays, we know one of the most challenging issues is data security.

The purpose of this research is declaring the security vulnerabilities to improve Database systems security in MySQL and preventing attacker from attacking to these systems, In order to know public and private sectors use where and in which level database security and knowing about security if they apply in Herat development companies.

III. Database Security

It is important to protect the data from unauthorized access, disclosure, modification or destruction if we are using DBMSs to store and maintain our data. Ensuring that users have the proper authority to view the data, insert new data, or update existing data is an important aspect of application development. Database security involves protecting a database from unauthorized access, malicious destruction, and even any accidental loss or misuse. Due to the high value of data incorporate databases, there is a strong motivation for unauthorized users to gain access to them.

It may happen that competitors have strong motivation to access sensitive information about product development plans, cost-saving initiatives and customer profiles of other companies. Sometimes they may want to access information regarding unannounced financial results, business transactions and even customer e-credit card numbers. They may not only steal valuable information, in fact, if they have access to the database, but they may also destroy it and great havoc may occur [2].

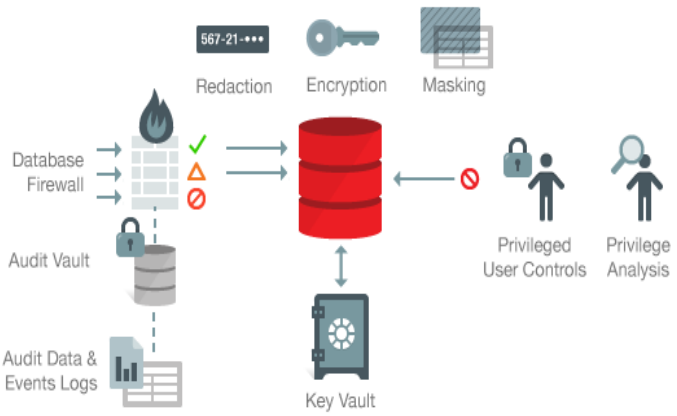

Fig. 1. Database security. (Oracle, database security, 2016)

Furthermore, the database environment is getting more complex where access to data has become more open through the internet and mobile computing. Thus, you can imagine the importance of having database security.

IV. Threats to a Database

Any situation or event that may affect a system and organization is called a threat. This threat may be intentional or unintentional and could be caused by a situation or event that involves a person, action or circumstance which results in harm to someone or to an organization. The harm may be loss of hardware, software or data (tangible) or could be loss of credibility or client confidence and trust (intangible). "However, focusing on database security alone will not ensure a secure database. This is because all parts of the systems must be secure. This includes the buildings in which the database is stored physically, the network, the operating system, DBMS and the personnel who have authorized access to the system" [3]. In this current article our focus is on data security in relational database management systems.

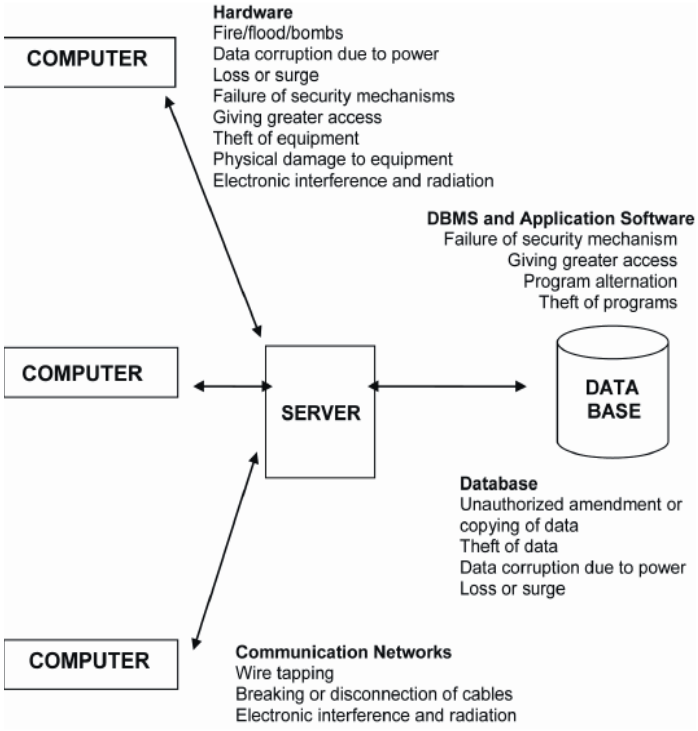

Fig. 2. Threats of Database (Connolly, 2005)

V. Types of Attacks on Database

Databases today are facing different kind of attacks. It is preferable to describe the attacks which can be performed on the databases. The major attacks on databases can be categorized as shown in inference, active and passive attacks and SQL Injection Attacks SQLIA on relational DBMSs.

Inference

Inference is a major attack on database systems. Inference is a way to derive sensitive data from non-sensitive data. The query fired is very much specific in this type of attack and matches exactly one data item which is called a direct attack. Attacks may be Indirect in the inference category which includes the use of statistical data to get sensitive information.

Passive Attacks on Databases

In a passive attack, the attacker only observes data present in the database.

Active Attacks on Databases

In an active attack, actual database values are modified. This is a serious kind of attack.

Volume 4, Issue 1, pp. 1-7, June 2022
ISSN (Online): 2789-8601

SQLIA (SQL Injection Attack)
Today, databases serve as a backend for many applications including web applications. One of the major attacks with such applications is SQL Injection Attack. This type of attack is called SQLIA or SIA. One of the basic reasons for SQLIA attacks is that most of the web applications use on the fly SQL queries without applying proper user input validation. Attackers can make the server run malicious SQL queries and can manipulate the database. Therefore, SQLIA is considered as a dangerous type of attack on databases.

VI. Data Protection in RDBMSs

Database protection means unlawful users have no access to the database and its sensitive information, whether intentional or accidental. Therefore, most companies consider the possibility of threats as actions for their database systems. This article addresses threats, countermeasures, and database security approaches to relational database threats and considerations of security techniques. [6]
RDBMS threats can be summarized as [7]:

- Required user privileges can be granted by the administrator. Abusing these privileges may lead to the creation of trapdoors from the program.
- The user has legal access to the database. He/she may have a malicious intention to abuse the tool.
- One of the threats is the software or operating system vulnerability. This helps the perpetrator violate sensitive information as they are back home.

Techniques data protection in RDBMS

This type ranges from physical controls to administrative procedures. It can be categorized into various forms of control as [8]:

1. Access Control of database or DCL
2. Backup and Restore
3. Enabling DBMS Firewall
4. Use of Views
5. Locking

VII. Implementing Security in MySQL

1. Access control using SQL Data Control Language
According to [9] The DCL (Data Control Language) commands are used for assigning the different authorizations to the user, these types of authorizations are known as privilege. Grant and Revoke commands are the DCL commands. The grant command is used for conferring the authorization to the users whereas the revoke command is used for withdrawing the authorization. Select, insert, update and delete are some of the privileges that are included in SQL standards. This language has commands such as grant and revoke which mainly deals with the rights, permissions and other controls of the database system.

"The MySQL privilege system ensures that all users may perform only the operations permitted to them. As a user, when you connect to a MySQL server, your identity is determined by the host from which you connect and the user's name you specify. When you issue requests after connecting, the system grants privileges according to your identity and what you want to do" [11].

Grant
"The database administrator defines the GRANT command in SQL for giving the access or privileges to the users of the database. Three major components which are involved in the authorization are the users, privilege/s (operations) and a database object" [9]. The user is the one who triggers the execution of the application program. "Operations are the component which is embedded in an application program. The operations are performed on database objects such as relation or view name"[9].

Syntax of GRANT Command [10]

```
grant <privilege record>
on <relation title or view title>
to <user/role record>;
```

```
mysql> create user 'hamid1'@'localhost' IDENTIFIED BY 'ssss';
Query OK, 0 rows affected (0.00 sec)
```

Fig. 3. Creating user account on local host.

```
mysql> GRANT SELECT ON library.* TO 'hamid1'@'localhost'
IDENTIFIED BY 'ssss';
Query OK, 0 rows affected (0.00 sec)
```

Fig. 4. Grant select privilege.

```
mysql> GRANT SELECT,UPDATE,INSERT ON library.books TO
'hamis'@'localhost' IDENTIFIED BY 'ssss';
Query OK, 0 rows affected (0.00 sec)
```

Fig. 5. Grant multiple privileges on one table.

According to the figure 3-3 and figure 3-4, we create user and give it only select privilege on library database. This user can't perform another operation on library database. In figure 3-5, we give multiple privileges just on books table to the user "Hamid" and lid have privileges only on books table.

Revoke

Another commends of DCL is revoke commend. This command in SQL is defined to take away the granted privileges (authorizations) from the user of the database. The one who has the authority to withdraw the privileges is the database administrator [9]

Syntax Revoke Command:

```
revoke <privilege list>
on <relation name or view name>
from <user/role list>;
```

The revoke command is similar to grant command except for the revoke keyword and 'from'. In given command, the operations included in the privilege are taken from the particular user or role list. Revoking becomes complex when privileges are propagated from one user to other.

We can revoke the given privileges by revoke statement.

```
mysql> revoke select ON library.* from 'hamid1'@'localhost';
Query OK, 0 rows affected (0.00 sec)
```

Fig. 6. revoking select privilege

```
mysql> REVOKE ALL PRIVILEGES, GRANT OPTION FROM
'hamid1'@'localhost';
Query OK, 0 rows affected (0.00 sec)
```

Fig. 7. revoking all privileges.

According figure 3-6, for security purpose we can revoke granted privilege from specific user and also can revoke all granted privileges according figure 3-7 and limiting his privileges.

2. MySQL Enterprise Firewall

Database Firewalls are a type of Web Application Firewall that monitors databases to identify and protect against database specific attacks which mostly seek to access sensitive information stored in the databases. "MySQL Enterprise Firewall is an application-level firewall that enables database administrators to permit or deny SQL statement executions based on matching against whitelists of accepted statement patterns".

This helps harden MySQL Server against attacks such as SQL injection or attempts to exploit applications by using them outside of their legitimate query workload characteristics. MySQL Enterprise Firewall has stored procedures that perform tasks such as registering MySQL accounts with the firewall, establishing their operational mode, and managing transfer of firewall data between the cache and the underlying system tables. In the firewall procedure we create the firewall tables, functions, stored procedures and install the necessary plugins. After running the script, the firewall is enabled. to test if it is enabled or not the following scripts could be used:

```
mysql> SHOW GLOBAL VARIABLES LIKE 'mysql_firewall_mode';
+------------------------------+-------+
| Variable_name                | Value |
+------------------------------+-------+
| mysql_firewall_max_query_size |  4096 |
| mysql_firewall_mode          |    ON |
| mysql_firewall_trace         |   OFF |
```

Fig. 8. shows MySQL firewall mode

To test the firewall, a current MySQL user is used, as an example – hamid@localhost. The user probably doesn't need all privileges, but for this example, everything is granted to this user according to figure 3-4. Accordingly, library schema is selected to test the MySQL procedures and functionalities. Afterward, the firewall is set to record those queries which are granted to be executed:

```
mysql> CALL `mysql`.`sp_set_firewall_mode`("hamid@localhost","RECORDING")
    +-------------------------------------------+
    | read_firewall_whitelist(arg_userhost,FW.rule) |
    +-------------------------------------------+
    | Imported users: 0  Imported rules: 0          |
    +-------------------------------------------+
    1 row in set (0.14 sec)

    +----------------------------------------+
    | set_firewall_mode(arg_userhost, arg_mode) |
    +----------------------------------------+
    | OK                                     |
    +----------------------------------------+
    1 row in set (0.22 sec)
    Query OK, 5 rows affected (0.28 sec)
```

Fig. 9. Call and set firewall to record allowed queries to allowed user.

Using the following script it is possible to check if the firewall is running and recording for our user or not:

```
mysql> SELECT * FROM MYSQL.FIREWALL_USERS;
+-------------------+-------------+
| USERHOST          | MODE        |
+-------------------+-------------+
| hamid@localhost |  RECORDING |
+-------------------+-------------+
1 row in set (0.02 sec)
```

Fig. 10. Display Firewall mode

It is possible to run multiple queries on library to check for recording, and then turning off the recording by turning on the protection mode:

```
mysql> CALL `mysql`.`sp_set_firewall_mode`("hamid@localhost","PROTECTING");
+----------------------------------------+
| set_firewall_mode(arg_userhost, arg_mode) |
+----------------------------------------+
| OK                                     |
+----------------------------------------+
1 row in set (0.00 sec)
```

Fig. 11. Set firewall mode to protecting.

The firewall is now protecting against non-whitelisted queries. We can execute a couple of the queries we previously ran, which should be allowed by the firewall. Now we run two new queries, which should be blocked by the firewall.

```
mysql> select * from books;
ERROR 1045 (42000): Firewall prevents statement

mysql> select * from copies;
ERROR 1045 (42000): Firewall prevents statement
```

Fig. 12. Queries prevented by firewall

Multiple queries are performed on the database in multiple mode, to see how much firewall activity you have, you may look at the status variables:

```
mysql> SHOW GLOBAL STATUS LIKE 'Firewall%';
+--------------------------+-------+
| Variable_name            | Value |
+--------------------------+-------+
| Firewall_access_denied   | 42    |
| Firewall_access_granted  | 55    |
| Firewall_cached_entries  | 78    |
+--------------------------+-------+
```

Fig. 13. Status of firewall variable

3. View

Views are virtual tables that are created through some operations on database objects. By removing certain columns or rows and combining certain tables, such views can be used to limit the scope of objects that users can manipulate or retrieve information from.

```
CREATE VIEW scrambling AS SELECT b.title, b.ISBN, SUM(c.id) copies, SUM(ba.id) authors FROM books b INNER JOIN copies c ON b.id = c.books_id INNER JOIN books_authors ba ON b.id = ba.books_id GROUP BY b.id
```

Fig. 14. View on library schema.

4. Backup and Recovery

Backup is one of the best tools to secure our data in case of loss. Backup means taking a copy of the sample process logs plus a copy of the relational database periodically and storing it in memory for later retrieval [8].

A complete backup of your database will include all the details of your database: tables, stored procedures, functions, views, authorization information, indexes and of course the data stored in those tables. There is also just enough information from the transaction log to ensure that the database can be restored in a continuous state and restore the database online if it fails during a recovery operation. In general, it can be assumed that by restoring a complete backup of the database, the database will return to the status quo in the country when the backup process begins. However, it is possible that the effect of the transaction that was created when the backup was started would still be included in the backup [10].

Backing up data is likely to be the only type of backup you need depending on the data recovery need. For example, let's say that you rely solely on full backups, you do it once in the middle of the night, and the server experiences a deadly crash at 11 pm. one night. In this case, you will only be able to restore the full database backup at midnight the previous day and so you may lose 23 hours of data. [10].

“A full database backup is probably the most common type of backup in the SQL Server world. This is actually a backup of the data file (s) associated with a database”. [12].

Backing up a database is essentially "archiving" your database because it existed at the time of the backup operation [12]. It is useful to know exactly what is in this "archive" and a full backup includes [12]:

- Copy of database when creating backup
- All user objects and data
- information Database system information
- User information
- License information
- Enough transaction reporting can return the database online if it fails

Full back up is a disaster recovery strategy for any database user, in the event of data corruption or the loss of a disk drive or even catastrophic hardware failure, where all the physical media of a server is lost or damaged. In such cases, the availability of a complete backup file, stored securely separately, maybe the only way to get back up and work on a new server with at least most of its data intact. If there is also differential backup, there is a strong possibility that we will be able to restore the database to a state very close to where it was shortly before the disaster. [12].

Implementation of Full Backup

The MySQL dump program can make backups. It can back up all kinds of tables.

```
C:\Program Files\MySQL\MySQL Server 5.5\bin>mysqldump –uroot –p library > libarary.sql
Enter password: ****
C:\Program Files\MySQL\MySQL Server 5.5\bin>
```

Fig. 16. Database recovery

```
C:\Program Files\MySQL\MySQL Server 5.5\bin>mysql –uroot –p library < libarary.sql
Enter password: ****
```

Fig. 16. Database recovery

5. Locking Mechanism

One of the most important characteristics of transactions is that they are separated. Technically, this means that executing transactions has the same effect as serial transactions, one after the other, with no overlap in both cases, respectively. These executions are called serialized executions, meaning they "have the same effect on serial execution" [13].

The most famous mechanism used to achieve serial locking. The concept is simple [13]:

- Each transaction retains access to the data it uses. The reservation is called a lock.
- There is a read lock and a write lock.
- Specifies the read lock transaction before reading a piece of data. Sets the write lock before writing data.
- Read the lock conflict with the write lock, and write the lock conflict with the write lock.
- The transaction can only get a lock if no other transaction has a contradictory lock in the same case. Therefore, it can obtain a read lock at x if it has no write lock transaction at x. It can only get write locks on x if there is no deal to read locks or write locks on x. [13].

Although the concept of lock is simple, its effects on performance and accuracy can be complex, counter-intuitive and difficult to predict. Making robust TP applications requires a complete understanding of the lock. [13].

Locking affects performance. When setting a lock transaction, it delays the other transactions needed to adjust the contradictory lock. All else being equal, the more transactions are made simultaneously, the greater the likelihood of such delays. The frequency and length of such delays can also be affected by transaction design, database layout, and transaction distribution and database distribution. To understand how to minimize this impact on performance, one must understand the locking mechanisms and how they are used and how these mechanisms and usage scenarios influence performance [13].

Locking also affects accuracy. Although locking usually strikes people correctly, not all users of the lock result in the correct results. “For example, it seems that reserving access to data before access eliminates the possibility of interfering with each other” [13]. But if the ability to serialize the target is simply not enough to lock the data before accessing it. The timing of the lock operation is also important [13].

Implementation of Transaction Locking

Although an application developer never directly confronts locks, he knows how to lock them for two reasons. First, locking can have a significant impact on the performance of a TP system. Most systems offer to tune mechanisms to optimize performance. To use these mechanisms, it is valuable to understand their effect on system internal behaviors. Second, some of these optimizations can violate correctly. Understanding lock execution helps to understand when such optimizations are acceptable and what other options are possible [13].

Lock Manager is a component that provides operations [13]:

- Set the lock with the mode of lock mode on behalf of the transaction ID for the transaction.
- Release the transaction ID lock on the data.
- Unlock all transaction IDs.

The locks on the lock table are low-level data structure in the main memory, just like a control table in an operating system (ie, like an SQL table). The locking and unlocking operation allow the locks to be inserted and removed from the locking table respectively [13]. The first and foremost thing is that the first use of a lock is to solve synchronization problems. If scripts are running that write to the database, but the multi-step operations are not susceptible to the problems described in the last section, locks are not required. Simple scripts that insert a row, delete a row, or update a row, and do not use previous SELECT results or user-entered data as input, require locking They don't. [6].

Locks are special property variables. By default, each MySQL table has a lock variable. If the user sets the lock variable for a particular table, no other user can perform specific actions on that table. The user who sets the lock variable holds the lock on the table. In practice, there are two types of locks for each table: READ LOCKs, when the user reads only one table, and WRITE LOCKs when the user is reading and writing a table. MySQL is not allowed to lock only one of the two tables used in the above transaction. The following rules apply to locks [13]:

Volume 4, Issue 1, pp. 1-7, June 2022
ISSN (Online): 2789-8601

If you keep the lock, all other tables used should also be locked. Failure to do so will result in MySQL error.

If aliases are used in queries-for example [13]:

```
SELECT * from customer c where c.custid=1
```

The alias must be locked with:

```
LOCK TABLES customer c READ
or:
LOCK TABLES customer c WRITE
```

VIII. Conclusion

Nowadays data security has become an important topic, a database is a repository for our data which is faced with different security threats such as SQL injection, inference attacks, passive attacks, and other active attacks. The most important aspects of securing database systems are the ability to bind an incoming request, determine where and how the attacker can attack, and find out measures to protect the database from attacks. This article focuses on verifying those key aspects of a secure database system from the backend aspect. Therefore, the best ways to make a database system safer, especially against SQL injection or other backend attacks are MySQL enterprise firewall to permit or deny SQL statement execution based on matching against whitelists of accepted statement patterns, changing the port of MySQL which attacker exploit default values, removing anonymous user, taking backup, use of views and applying access privileges on the user account that every user can access to granted databases and can perform the granted operation on specific database and table.

Every organization and company in the business environment needs to collect data from their users. Users who provide their information to the companies have to trust them and be confident about their data security. Organizations should use various standard and reliable data protection methods in their databases to protect their data from unknown and unauthorized access. There are many ways to protect databases and make them more secure we discussed some of them in this article.

X. References

[1] Murray, M. C. (2010). Database Security: What Students Need to Know
[2] Database security issues. (n. d.). Retrieved December 29, 2009, from data bases.about.com/ od/security/database_security_issues.htm Mannino, M. V. (2001).
[3] Oracle. (2016). database security. Retrieved from http://www.oracle.com/technetwork/database/security/overview/index.html
[4] Jeffrey A. Hoffer, J. F. (2011). Essentials of sysetems analysis and Design . pearson.
[5] Connolly, B. (2005). Database systems. Pearson.
[6] Madden, A. D. (2003). DATA PROTECTION. European: With financial support from the EU's Fundamental Rights and Citizenship Programme.
[7] Derclaye, E. (2005). What is a Database? The Journal of World Intellectual Propert, 4.
[8] Mustafa, A. A. (2016). See discussions, stats, and author profiles for this pubSecurity Of Database Management Systems. The Center of Judicial Expertise and Research 15 PUBLICATIONS 16 CITATIONS , 2.
[9] Ji-Won Byun, N. (2009). Privacy Protection in Relational Database Systems. The user has requested enhancement of the downloaded file.
[10] Martti Laiho, D. A. (2010). SQL Transaction. DBTech VET Teachers project.
[11] MySQL. (NA). Chapter 4 Access Control and Account. Retrieved April 30 2022 from https://dev.mysql.com/doc/mysql-security-excerpt/8.0/en/access-control.html
[12] McGehee, S. (2012). SQL Server Backup and Restore . The Red Gate Guide.
[13] Newcomer, B. a. (2001). Locking. Chicago: A. Bernstein and Eric Newcomer.
[14] Abraham, A., Mauri, J. L., Buford, J., Suzuki, J., & Thampi, S. M. (Eds.). (2011). Advances in Computing and Communications, Part I: First International Conference, ACC 2011, Kochi, India, July 22-24, 2011. Proceedings (Vol. 190). Springer Science & Business Media.
[15] Lynch, Steve (2015). Database Security. Retrieved May 15 2022 from https://resources.infosecinstitute.com/topic/database-security/