Junaid Girkar

60004190057

TE COMPS A4

**Experiment No 9**

**Aim:** A case study on Database security issues and measures taken to handle those issues.

**Theory:**

The breach could be caused by a variety of software vulnerabilities, misconfigurations, or habits of misuse or carelessness. Here are some of the most well-known causes and forms of cyber threats to database security.

**Insider Threats**

An insider threat is a security risk posed by one of the three sources listed below, each of which has privileged access to the database:

* A nefarious insider with nefarious intentions
* An irresponsible employee who exposes the database to attack as a result of his or her conduct.
* Any unauthorized person who obtains credentials by social engineering or other means, or gains access to the database's credentials.

One of the most common causes of database security breaches is an insider threat, which generally happens when a large number of employees have been given privileged user access.

**Human Error**

Nearly 50% of all data breaches are still caused by weak passwords, password sharing, data erasure or damage by mistake, and other undesired user behaviors.

### **The exploitation of Database Software Vulnerabilities**

Attackers constantly attempt to isolate and target vulnerabilities in software, and database management software is a highly valuable target. New vulnerabilities are discovered daily, and all open source database management platforms and commercial database software vendors issue security patches regularly. However, if you don’t use these patches quickly, your database might be exposed to attack.

Even if you do apply patches on time, there is always the risk of zero-day attacks, when attackers discover a vulnerability, but it has not yet been discovered and patched by the database vendor.

### **SQL/NoSQL Injection Attacks**

The use of arbitrary non-SQL and SQL attack strings in database queries is a database-specific danger. These are usually queried that are developed as extensions of web application forms or that are received via HTTP requests. If developers do not follow secure coding practises and the organization does not do regular vulnerability testing, every database system is vulnerable to these attacks.

**Buffer Overflow Attacks**

A buffer overflow occurs when a process attempts to write more data to a fixed-length block of memory than it is capable of holding. Attackers could utilize the surplus data stored at nearby memory addresses as a jumping-off point for their attacks.

**Denial of Service (DoS/DDoS) Attacks**

In a denial of service (DoS) assault, the cybercriminal uses a huge number of bogus requests to overwhelm the target service—in this case, the database server. As a result, the server is unable to handle genuine user requests and frequently crashes or becomes unstable.

Fake traffic is generated by a large number of computers in a botnet controlled by the attacker in a distributed denial of service (DDoS) assault. Without a sufficiently scalable defensive architecture, this results in extremely high traffic volumes, which are impossible to halt. Cloud-based DDoS prevention services can dynamically scale up to handle massive DDoS attacks.

**Malware**

Malware is computer programme designed to exploit security flaws or harm a database. Malware could infiltrate the database's network through any endpoint device. Because of their high value and sensitivity, malware security is critical on any endpoint, but more so on database servers.

**Distributed Systems**

Because there are many users, diverse data, multiple sites, and distributed control, a distributed system requires more security than a centralised system. We'll look at the many aspects of distributed database security in this chapter.

There are two categories of intruders in distributed communication systems:

* They are passive eavesdroppers who monitor messages and obtain private information.
* Active attackers not only monitor messages, but actively tamper with data by adding new data or changing current data.

Security measures include communications security, data security, and data auditing.

**Communications Security**

In a distributed database, a lot of data communication takes place owing to the diversified location of data, users and transactions. So, it demands secure communication between users and databases and between the different database environments.

Security in communication encompasses the following −

* Data should not be corrupt during transfer.
* The communication channel should be protected against both passive eavesdroppers and active attackers.
* In order to achieve the above stated requirements, well-defined security algorithms and protocols should be adopted.

Two popular, consistent technologies for achieving end-to-end secure communications are −

* Secure Socket Layer Protocol or Transport Layer Security Protocol.
* Virtual Private Networks (VPN).

**Data Security**

In distributed systems, it is imperative to adopt measures to secure data apart from communications. The data security measures are −

* Authentication and authorization − These are the access control measures adopted to ensure that only authentic users can use the database. To provide authentication digital certificates are used. Besides, login is restricted through username/password combination.
* Data encryption − The two approaches for data encryption in distributed systems are −
  + Internal to distributed database approach: The user applications encrypt the data and then store the encrypted data in the database. For using the stored data, the applications fetch the encrypted data from the database and then decrypt it.
  + External to distributed database: The distributed database system has its own encryption capabilities. The user applications store data and retrieve them without realizing that the data is stored in an encrypted form in the database.
* Validated input − In this security measure, the user application checks for each input before it can be used for updating the database. An unvalidated input can cause a wide range of exploits like buffer overrun, command injection, cross-site scripting and corruption in data.

**Data Auditing**

In order to determine the security measures that should be implemented, a database security system must detect and monitor security infractions. It's often difficult to discover security breaches right after they happen. Examining audit logs is one way to find security issues. Information like this can be found in audit logs.

* Date, time, and location of unsuccessful access attempts.
* Attempts to gain access that were successful.
* Changes to the database system that are critical.
* Access to massive amounts of data, particularly from various databases.

All of the foregoing data provides insight into the database's activity. Periodic log analysis aids in identifying any unusual activity, as well as its location and time of occurrence.

Document Databases(NoSQL)

Authentication and encryption are practically non-existent in NoSQL databases, or are very poor when implemented. The following are some of the security concerns with NoSQL databases:

* By default, administrative user or authentication is disabled.
* It features a password storing system that is extremely vulnerable.
* The client sends plaintext messages to the server (MongoDB)
* External encryption techniques such as LDAP, Kerberos, and others are not supported.
* Data files are not encrypted due to a lack of encryption capabilities.
* Weak authentication on both the client and server sides
* SQL injection vulnerability
* Denial-of-service (DoS) attacks are a type of cyber-attack
* At rest, data is unencrypted.
* The available encryption solution isn't ready for use in production.
* For client communication, encryption is not accessible.

With all these security problems, it is best to understand that NoSQL databases are still new technologies and more security enhancements will be added to newer versions. Enterprise package Cassandra tools provided by companies like Datastax do have more security enhancements and hence are more secure and provide companies with all the security needed.

**Solution**

**1. Pseudonyms-based Communication Network:**

In the context of this system, users can have access to multiple services by inserting their credentials only once, that is when they are initially connected to the system. Such a system is called anonymous because users can be known only through their pseudonyms, and the transactions demonstrated by the same user cannot be linked, as their identity is disclosed.

For this reason, it is considered the best means in terms of user protection.

Additionally, the Credential Authority CA prevents the sharing of credentials or pseudonyms and guarantees that users who enter the system have a public and secret key that makes them unique to the system. Another entity in the system is the Veriﬁer V, whose role is to certify the validity of the user credentials and to communicate with either the Issuing Authority or the Credential Authority and inform that the user is not the owner of the credential that is presenting.

Users, in terms of a digital credential, transmit the public key and the CAs digital signature derived from a Proof of Knowledge, through which they prove that they know the secret key and the attributes in the digital credential that satisﬁes the particular attribute property they are revealing.

**2. Monitoring, Filtering, Blocking:**

Malicious tasks and queries cannot be detected and disabled in NoSQL databases. Advanced scripts can readily evade the Kerberos central authentication mechanism, and in general, monitoring is limited to data processing primarily in the API.

No information about cluster communication, user connection details, or data changing information (even editing or deleting) is recorded in a cloud environment. Because there are no log files, identifying incidents of data breach or malicious data loss in the cluster is a difficult task.

Big data technologies provide real-time security systems, resulting in high-speed data analysis. As a result, anomaly detection is applied in real time, and security analytics can be recorded and updated on a regular basis.

**Mobile Database**

1. **Security**

The data which is left at the fixed location is more secure as compared to mobile data. That is mobile data is less secure. Data is also becoming more volatile and techniques must be able to compensate for its loss. The most important thing needed in this environment is the authorizing access to critical data and proper techniques.

1. **Data distribution and Replication**

Uneven distribution of data among the mobile units and the base stations take place here. Higher data availability and low cost of remote access is there in data distribution and replication. The problem of Cache management is compounded by the consistency constraints. The most updated data and frequently accessed data is provided by the Caches to the mobile units. It processes their own transactions. There is more efficient access to data and higher security is available.

1. **Replication issues**

There is an increase in costs for updates and signaling due to an increase in the number of replicas. Mobile hosts can move anywhere and anytime.

1. **Division of labor**

There is a certain change in the division of labor in query processing because of certain characteristics of the mobile environment. There are some cases in which the client must function independently of the server.

1. **Recovery and fault tolerance –**

Fault tolerance is the ability of a system to perform its function correctly even in the presence of internal faults. Faults can be classified in two types: transient and permanent. Without any apparent intervention, a transient fault will eventually disappear but a permanent fault will remain unless it is removed by some external agency.

The mobile database environment must deal with site, transaction, media, and communication failures. Due to limited battery power there is a site failure at MU. If a voluntary shutdown occurs in MU, then it should not be treated as a failure. Whenever Mu crosses the cells, most frequently there will be transaction failures during handoff. Due to failure of MU, there is a big cause of network partitioning and affection of the routing algorithms. The characterization of mobile computing is done by:

* Limiting resource availability
* Frequent disconnection
* High mobility
* Low bandwidth

**Solution**

For mobile operators, the first step in defeating attacks on their networks is to recognize their newfound role as an ISP. This means implementing a layered defense for their network that:

1. Changes security policies and practices to better reflect the new threats.

2. Concentrates, whenever possible, wireless data services into a smaller number of data centers. Many mobile operators in Europe have already taken these types of steps to protect their core networks.

3. Protects end users by implementing technology on their devices and in the network – e.g., anti-virus, firewalls, content scanning – that provides file-level security.

4. Take an architecture approach to implementing security solutions in their network; point solutions are not sufficient.

5. Make client-side anti-virus and firewall software readily available to their subscribers who use data devices (e.g., feature phones with data capabilities, Smartphone, notebook computers)

**Object Oriented Database**

**1. Aggregation:**

The aggregation problem occurs when a user can form aggregates of related items, all of which are classified at some level, that deduce classified data [26]. The higher level information (which may be thought to be subject to a higher level of security clearance) may be inferred from a large number of lower level data items. A collection of information items may be required to be classified at a higher security level than any of the individual items that comprise it.

The aggregation problem occurs when a subject's right to individual pieces of information results in knowledge to which it does not have a right. The aggregation problem prevents the subject from gaining access to information of higher sensitivity by aggregating lower sensitivity data. This is usually addressed by restricting combinations of accesses in certain ways.

**2. Inference problem:**

The word “inference” means “forming a conclusion from premises”. Users of any database can draw inferences from the information they have obtained from the database and prior additional information (called supplementary knowledge) they have. The inference can lead to information disclosure if the user is able to access information they are not authorized to read. This is the inference problem in database security. Inference problem occurs when a user can deduce (or infer) information from a collection of individual accesses against a database summarized different approaches to handle the inference problem:

(1) place restrictions on the set of allowable queries that can be issued by a user;

(2) add noise to the data; and

(3) augment a database with a logic-based inference engine to modify queries before the database processes them.

**3. Polyinstantiation:**

This problem arises when users with different security levels attempt to use the same information. The variety of clearances and sensitivities in a secure database system result in conflicts between the objects that can be accessed and modified by the users. Through the use of polyinstantiation, information is located in more than one location, usually with different security levels.

Obviously, the more sensitive information is omitted from the instances with lower security levels. Although polyinstantiation solves the multiparty update conflict problem, it raises a potentially greater problem in the form of ensuring the integrity of the data within the database. Without some method of simultaneously updating all occurrences of the data in the database, the integrity of the information quickly disappears. In essence, the system becomes a collection of several distinct database systems, each with its own data.

**Solution**

**Multilevel Security:**

Multilevel secure computers protect objects classified at more than one level and allow sharing between users of different clearance levels. Objects are labeled with their sensitivity levels. Subject arc associated with clearance levels.

A multilevel secure computer arbitrates all access of objects by subjects. The arbitration is carried out by the rcfcrcncc monitor according to a security policy. MLS/DRMSs must deal with large numbers of objects, intcrrclatcd in complex ways which have semantic meaning. This causes several problems.

The first is efficiency. Large numbers of objects can cause a large burden on the access monitor. Secondly, all of these objects must be classified in a complete and consistent way. The third problem is representing and manipulating objects containing data of multiple sensitivity levels. The interrelations of the data and their semantics lead to an inference problem. Inference occurs when information which can be rctricvcd from the database allows other data to be deduced. Inference provides a flow of data which is not arbitrated by the reference monitor.

**Conclusion:** We studied the different types of security issues in various types of database systems and the solution to these issues.