**IE2042- Database Management Systems for Security**

**Online Auction Database System**

**Group Assignment**

**Database Design, Implementation and Security**

A picture containing icon

Description automatically generated

BSc (Hons) in Information Technology

Sri Lanka Institute of Information Technology

Malabe

Sri Lanka

**Part 1 - ERD**

Diagram, schematic

Description automatically generated**(1)**

A picture containing diagram

Description automatically generated**(2)** **Logical Model**

Graphical user interface

Description automatically generated(3) **Normalized logical model**

**SQL Parts**

(4) **Creating Tables**

![Graphical user interface, text, application, email

Description automatically generated]()

![Graphical user interface, text, application, email

Description automatically generated]()

![Graphical user interface, text, application

Description automatically generated]()

![Text

Description automatically generated]()

**Inserting values**

![Text

Description automatically generated]()

![Text, letter

Description automatically generated]()

![Table

Description automatically generated]()

![Table

Description automatically generated]()

![Text

Description automatically generated with medium confidence]()

**Assumptions –** 1. In our database we didn’t use seller name, so that we use seller id.

2. We have to pass the item number to the procedure 4 then it will increment the bid price by 15% of the current bid of the saman.

**Trigger 1**

Explaining – when a buyer bid for an item this will check whether it is higher than the last bid price.

![Text

Description automatically generated]()

![Graphical user interface, text, application, email

Description automatically generated]()

**Trigger 2**

Explaining – Database is only inserting members who have account balance more than Rs 300000.

![Graphical user interface, text, application, email

Description automatically generated]()

**Views**

1.View for seller

![A picture containing text

Description automatically generated]()

![Text

Description automatically generated]()2.View for buyer

**Indexes**

1. Index no 1 getting full name of the member.

![Graphical user interface, text, application

Description automatically generated]()

1. Index no 2 getting buyer full address.

![Text

Description automatically generated]()

**Procedures**

![Graphical user interface, text, application, email

Description automatically generated]()Procedure no 1

Procedure no 2

![Graphical user interface, text, application, email

Description automatically generated]()

Procedure no 3

**Assumption** – In our database we didn’t use seller name, so that we use seller id.

![Graphical user interface, text, application, email

Description automatically generated]()

Procedure no 4

![Graphical user interface, text, application, email

Description automatically generated]()**Assumption –** When we pass the itemno it will increment the bid price by 15% of the current bid of the saman.

**Part 2**

**Database Vulnerability Report**

**SQL INJECTION**

**what is SQL Injection?**

SQL Injection is a major online database vulnerability. Structured Query Language is a computer language used to store, alter, and retrieve data from relational databases. Structured Query Language An adversary can access a database and sensitive information through code injection by changing SQL logic. SQL injection attacks are a widespread internet vulnerability.

SQL Injection can damage any website or online application that uses SQL. MySQL, Oracle, SQL Server are SQL databases.

SQL injection gives cybercriminals unrestricted database access. SQL and NoSQL injections are two types of computer assaults. SQL injections target traditional databases, NoSQL injections massive data databases. SQL injection can remove, alter, or steal private data. It compromises the data's trustworthiness, safety, and security. It also jeopardizes the company's reputation and operations. SQL injection attacks aim to acquire sensitive data such as credit card numbers, SSNs, and passwords from a database table. Malicious code can be inserted into the table's database query to retrieve this information.

**Types of SQL Injection**

**In-band SQLI**

SQL injection, commonly referred to as classic SQL injection, is a specific sort of SQL injection. To say something is "in-band," we mean that the attacker receives a direct response via the same communication channel that they used to attack. This is the case, for example, if an attacker performs the attack manually using a web browser.

In-band SQLI is 2 types as below,

1. Error based SQLI

Error-based SQL injection modifies the database's data via error output. In in-band injection, the attacker gathers database data while attacking using the same channel. A common attack method. By exploiting a programming weakness, the server can report a SQL error instead of the requested data. grep's extract function automates this process. The database error is often enough for an attacker to fully grasp the database. [1]

1. Union based SQLI

Utilizes the union SQL operator to aggregate the outcomes of two or more select statements into a single result, which is subsequently returned as part of the http response.The outcomes of the combined select statements are then returned as a single result.

In order for a UNION query to be successful, there are two primary requirements that need to be satisfied:

1. The number of columns returned by each query must be the same.
2. There must be backward compatibility between each query and each of the data types in each respective column.

**Inferential / blind SQLI**

Using data payloads, the attacker sends them to the server and watches how the server responds and behaves. An attacker cannot see in-band information about an attack using this method, which is known as "blind SQLi," because data is not sent to the hacker from a website database.

Blind SQL injections rely on the server's response and behavior patterns; therefore they are often slower to execute but may be just as dangerous. In order to classify blind SQL injections, we can use the following [2]

1. Boolean-based Blind SQLI

A SQL query is sent to the database by the attacker, which prompts the application to produce a value. According to whether or not the query is true, the outcome will be different. Depending on the outcome, the data in the HTTP response may be modified or remain unchanged. The attacker can then determine if the message's outcome was real or false. [2]

1. Time-based Blind SQLI

Database is forced to wait before responding to a SQL query sent by the attacker. The attacker can tell if a query is true or false based on how long it takes the database to answer. Responses to HTTP requests might be instantaneous or require some amount of time to be created. As a result, the attacker does not need to rely on the database to determine if the message they used returned true or false. [2]

**Out-of-band SQLI**

This type of attack is only possible if the database server used by the web application has certain functionalities enabled. SQL injection attacks are typically employed as an alternative to the in-band and inferential SQLi methods.

Out-of-band an attacker may resort to SQLi if they are unable to launch and gather information via the same channel, or if the server is too slow or unstable for them to do so. Both DNS and HTTP rely on a server's processing power to send data to an attacker. [2]

Tools Associated With SQLI

* SQLmap
* SQLSus
* Mole
* Haviji
* BSQL Hacker

[3]

**How SQL Injection Works**

For example, login pages, search pages, and feedback forms are all targets of the attacker. The app provides the user a form to complete in order to log their activity. the form is submitted by a cybercriminal using a SQL attack.

* Common Vulnerable Login Query

**SELECT \* FROM users**

**WHERE login = ‘abc’**

**AND password = ‘123’**

If it returns something, then login.

* MS SQL Server Login Syntax

**VAR sql = “SELECT \* FROM users**

**WHERE login = +formusr+**

**AND password = +formpwd+.**

**formusr = ‘or 1=1**

**formpws = anything**

* Injecting through Strings

**SELECT \* FROM users**

**WHERE username = ‘ ‘ or 1=1**

**AND password = ‘anything’**

**-** The string parameter is closed.

-Afterwards, the SQL command is regarded as complete.

-Other than string fields, there are also numeric and date fields.

* Injecting Numeric Fields

**SELECT \* FROM student**

**WHERE studentid = 34546**

**AND age = 15**

* PHP/MySQL login Syntax

**$sql = “SELECT \* FROM student WHERE “studentid = $formstudentid AND age = $formage”;**

**$formstudentid = 1 or 1=1#**

**$formage = 15**

**SELECT \* FROM student**

**WHERE studentid = 1 or 1=1**

**# AND age = 15**

-Exploit data is included in the string. SQL queries are made up of this string of characters.

-The database receives a SQL query from the application.

-Sends data back to application after database runs query including exploit

-The user receives data from the application.

**Impact of SQL Injection**

* Unauthorized access to private information.
* Negative reputation.
* The alteration of confidential data.
* The database server has been taken over by a third party.
* There was a loss of data.

**Countermeasures and Mitigation**

* Verifying the accuracy of input

Simple input checks prevent many attacks. Before using user input, check type, size, length, format, and range. String variables should only allow expected values. Reject submissions with binary, escape, or non-ASCII data. This prevents script injection and buffer overrun. Validate XML document data against schema.

* Validate Input Sources

Because there are so many different ways to breach a database's security, the developer needs to examine and authenticate all of the database's input sources and prohibit access from persons or websites that cannot be identified or trusted.

* Input Checking Functions

SQL injection attacks use characters like SELECT and INSERT. Remove these characters and sequences to reduce injection attack risk. Look for unwanted words like "INSERT," "UPDATE," "DELETE," "DROP," etc. Check a statement or user input. Function should handle this.

* User Permissions & Access Rights

Create accounts with "low privileges" that can be accessed by programs. Never provide database accounts instance-level privileges. The database owner or schema owner should never be allowed to provide database accounts ownership rights. Make sure you're familiar with your database's permissions.

* Keep database backups

**DENIAL OF SERVICE ATTACK**

**What is DOS Attack?**

Database management systems are vulnerable to denial-of-service attacks as well. Network-based attacks are known as Denial of Service (DoS) attacks. As a form of attack, DOS is used to prevent legitimate users from accessing or slowing down a resource. An attempt to make a networked system unusable at some point but not permanently harm it is the goal of this malicious effort A barrage of data packets is sent to the target machine by the attacker, who uses specially coded software to put a strain on the system's limited resources. Dos assaults aim to damage a company's reputation, injure customers, and cause financial losses; in extreme situations, they can even result in the loss of human life. Currently, the most costly cybercrime for victims' organizations is denial of service attacks.

**Types of DOS Attacks and How they Work**

1. **Bandwidth Attacks**

Attacks will eat up all of the available bandwidth on the network. Every website is allotted a certain amount of bandwidth for its hosting, and if more people visit the site, the hosting will be forced to throttle the site. The attacker follows suit. The attacker will repeatedly refresh and open many pages on a website, causing it to go offline.

1. **Protocol Attacks**

When an attack is launched, it will eat up the system's resources such as the CPU and RAM. To transport data over a network, protocols are necessary. As a result of these assaults, the victim's system consumes an excessive amount of resources because of a specific feature or implementation flaw of a protocol installed there.

1. **ICMP Flood Attacks**

A "ping flood" occurs when an attacker floods a victim's computer with repeated ICMP echo queries (DoS). For each ICMP request, the attacker hopes the victim would send an ICMP "echo reply" packet, consuming both outgoing and incoming bandwidth. Most effective when attacker has greater bandwidth than target. If the target machine is too slow, user performance can suffer.

1. **UDP Flood**

The goal of a UDP flood is to overload a targeted server with User Datagram Protocol (UDP) packets, which is a sort of denial-of-service attack. Additionally, a denial-of-service attack can occur if the firewall protecting the targeted server is overwhelmed by UDP flooding.

1. **TCP SYN Flood Attacks**

By exploiting TCP's three-way handshaking, an attacker can transmit connection requests from unreachable source addresses. TCP-SYN flooding prevents the sender from receiving the final handshake message. This causes the server to allocate memory and wait for a pending connection. The target system's congested buffer prevents other systems from communicating with it.

**Impact of DOS Attack**

* Performance issues with the database/server
* Database or server unavailability
* Inability to access any database/server causes a rise in spam emails.
* A loss of wireless or wired internet service.

**Countermeasures and Mitigation**

One approach can't block all Database DoS attacks. Because every feature can be exploited, no single solution can stop every attack. DoS protection takes time and tools. Ensure your databases are patched and configured. Understanding vulnerability is difficult. Uninstall network software. High-profile databases need preventative and investigative processes.

By making the appropriate judgments, we may minimize our exposure to a DOS attack.

* **Build redundancy into infrastructure**

Disperse your servers across various data centers and use a load balancing system to distribute traffic among them if you want to make it difficult for an attacker to execute a DDoS attack against them. Data centers should be located in separate nations or areas of the same country if possible.

* **Practice Basic Network Security**

Networks can be protected from intrusion by implementing stringent security measures. Anti-phishing techniques and secure firewalls that allow only limited traffic from the outside are all examples of safe practices.

* **Deploy anti-DDoS hardware and software modules**

Servers need network and web application firewalls. Some volumetric attacks can be stopped from the outside by configuring the firewall or router.

* **Check for security patches and keep updated.**

# References

|  |  |
| --- | --- |
| [1] | [Online]. Available: https://beaglesecurity.com/blog/vulnerability/error-based-sql-injection.html. |
| [2] | [Online]. Available: https://www.imperva.com/learn/application-security/sql-injection-sqli/#:~:text=SQL%20injections%20typically%20fall%20under,data%20and%20their%20damage%20potential.. |
| [3] | "https://www.serverwatch.com/reviews/sql-injection-detection-tools/," [Online]. |