SQL Injection scanner

Preliminary Investigation:

Introduction:
A widespread problem for programmers is avoiding injection attacks. In other words, writing secure code that doesn't allow for the injection of malicious code into an application. Injection attacks stem from a lack of strict separation between program instructions (i.e., code) and user-provided (or external) input. This allows an attacker to inject malicious code into a data snippet. The lack of separation means that an application can execute malicious code as crafted by the attacker.

Injection attacks are some of the most successful and widespread types of attack. Additionally, SQL injection is one of the most common types of injection attack. To carry it out, an attacker provides malicious SQL statements through the application. These control the backend database server. SQL injection is independent of the technology used for the underlying application. Thus, this attack technique is common.

SQL injection continues to be a significant threat to application security, but the right SQL injection scanner can protect your software from malicious attacks.

In SQL attacks, hackers trick an application into sending unexpected SQL commands. Web form fields are a typical point of attack. Hackers enter a command in the form field, and if the application adds it to an SQL query without properly sanitizing it, attackers can include their own SQL commands to be executed by the database.

SQL injection scanner technology can easily protect your organization against these type of attacks, but choosing the right technology is critical. Your SQL injection scanner solution must be easy to use, and it must not create obstacles for development teams working to meet aggressive development timelines.

Aim :- To find and aware the modern applications from sql injection vulnerability.

What is sql injection scanner?

The SQL Injection Scanner (Light Scan) performs a quick and fast scan of a target URL that allows it to identify vulnerabilities in web applications. It does that by searching if the parameters of the target URLs are vulnerable to SQL Injection attack and reports the malicious pages that could affect your website.

The online scanner includes two steps:

1. **Spidering the target:** In this first step, the scanner tries to identify all the pages within the target web application, including injectable parameters in login forms, URLs, headers, etc.
2. **Accurate SQL injection testing:** During this phase, for each page discovered in the previous step, the online tool will try to detect if the parameters are vulnerable to SQL Injection and report them in the results page.

How does the SQL Scanner work?

To better secure your web applications from SQL injection attack is to identify and fix security vulnerabilities before hackers do.

Our SQL injection scanner was created to easily perform SQL injection testing and find web applications flaws in a timely manner.

The SQL Injection Scanner using OWASP ZAP (Full Scan) is our comprehensive online security solution that allows you to do a complete SQL injection assessment of the target web applications and find critical vulnerabilities with a significant impact for any business.

The online tool offers an intuitive and simple interface using OWASP ZAP , the most popular open-source web application security scanner.

The SQL Injection Scanner (Light Scan) performs a quick and fast scan of a target URL that allows it to identify vulnerabilities in web applications. It does that by searching if the parameters of the target URLs are vulnerable to SQL Injection attack and reports the malicious pages that could affect your website.

The online scanner includes two steps:

1. **Spidering the target:** In this first step, the scanner tries to identify all the pages within the target web application, including injectable parameters in login forms, URLs, headers, etc.
2. **Accurate SQL injection testing:** During this phase, for each page discovered in the previous step, the online tool will try to detect if the parameters are vulnerable to SQL Injection and report them in the results page.

How and Why Is an SQL Injection Attack Performed?

To make an SQL Injection attack, an attacker must first find vulnerable user inputs within the web page or web application. A web page or web application that has an SQL Injection vulnerability uses such user input directly in an SQL query. The attacker can create input content. Such content

is often called a malicious payload and is the key part of the attack. After the attacker sends this content, malicious SQL commands are executed in the database.

SQL is a query language that was designed to manage data stored in relational databases. You can use it to access, modify, and delete data. Many web applications and websites store all the data in SQL databases. In some cases, you can also use SQL commands to run operating system commands. Therefore, a successful SQL Injection attack can have very serious consequences.

- Attackers can use SQL Injections to find the credentials of other users in the database. They can then impersonate these users. The impersonated user may be a database administrator with all database privileges.
- SQL lets you select and output data from the database. An SQL Injection vulnerability could allow the attacker to gain complete access to all data in a database server.
- SQL also lets you alter data in a database and add new data. For example, in a financial application, an attacker could use SQL Injection to alter balances, void transactions, or transfer money to their account.
- You can use SQL to delete records from a database, even drop tables. Even if the administrator makes database backups, deletion of data could affect application availability until the database is restored. Also, backups may not cover the most recent data.
- In some database servers, you can access the operating system using the database server. This may be intentional or accidental. In such case, an attacker could use an SQL Injection as the initial vector and then attack the internal network behind a firewall.

How to Prevent an SQL Injection

The only sure way to prevent SQL Injection attacks is input validation and parametrized queries including prepared statements. The application code should never use the input directly. The developer must sanitize all input, not only web form inputs such as login forms. They must remove potential malicious code elements such as single quotes. It is also a good idea to turn off the visibility of database errors on your production sites. Database errors can be used with SQL Injection to gain information about your database.

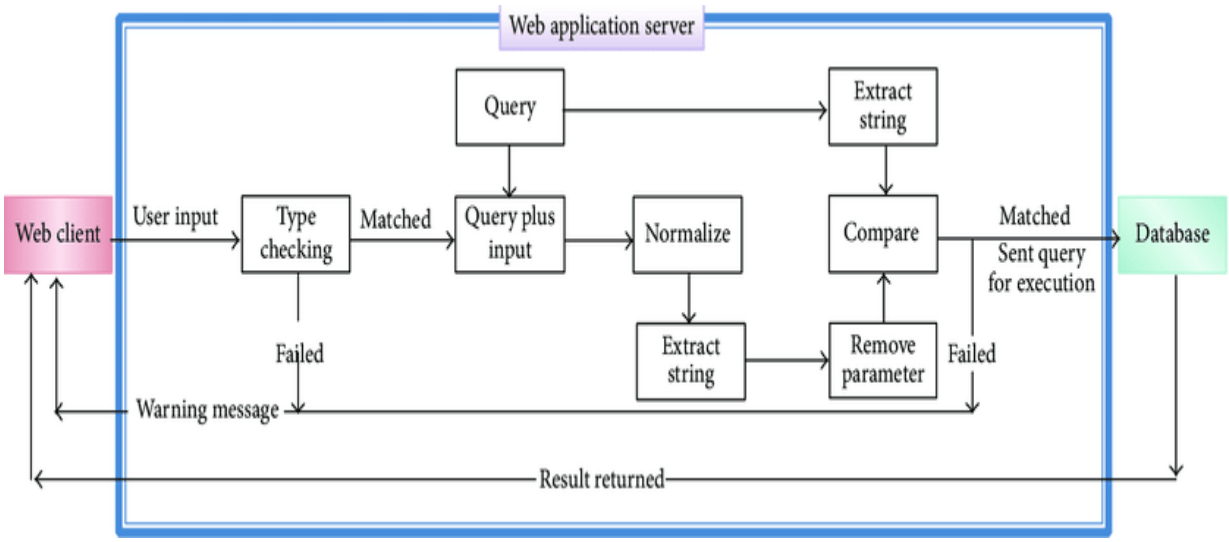
If you discover an SQL Injection vulnerability, for example using an Acunetix scan, you may be unable to fix it immediately. For example, the vulnerability may be in open source code. In such cases, you can use a web application firewall to sanitize your input temporarily.

Prepared statements :

The statements that have been pre-complied with the SQL query is called as prepared statement. SQL query is nothing but the plain text representation of the statement written by programmer while developing database access programmed. Prepared statements in SQL query binds variables that allow you to put inputs into subsequent queries. In Java input set method is used to set bind variable such as setString(index, output) call for a String type output variable. Set methods render the additional security to confirm each input variable with respect to its declared

type. The primary purpose of prepared 15 statement is to increase security and efficiency. Prepared statements are built to execute same statement number of times while compiling the statement. This property is not available in plain text SQL statement. The functionality of the prepared function is same as the plain text SQL statements, but the prepared statements have more structured way than the plain text SQL statements. Manipulation of the structure of the pre-complied query can prevent using structure handling of the prepared statement, hence preclude SQL injection vulnerability. The limitation of the prepared statement is that they can only be created if the structure of the statement is known before the creation of the statement. Thus the dynamically created statements can be created with knowing the structure of the statement which is not possible in prepared statement. Prepared statements are precompiled, once the statements are built by the Connection object in Java. When all of the inputs are set into the statement and the statement is executed, it sent to the database.

Proposed System block diagram:



Requirements And Specification:

Software Requirements:
For developing the application the following are the Software Requirements:
Operating system : Linux
Coding Language : php, python
Server : apache/ngenix

Hardware Requirements:
For developing the application the following are the Hardware Requirements:
Processor : Pentium IV or higher
RAM : 1GB
Space on Hard Disk : minimum 10GB

Problem Statement:

To identify the vulnerabilities of the website for the purpose of improving the security features and creating a blockchain based website. Website is made for the registration of the passport which contains the personal details of the individual. Attack is performed on the website which is created of our own using block
chain and denied its service.

Future scope:

The above tools will test and let you know if your website has SQL injection vulnerability. If you are wondering how to protect your site against SQL injection, then the following will give you an idea.

The poorly coded web application is often responsible for SQL injection, so you got to fix the vulnerable code. However, another thing you can do is to implement the WAF (web application firewall) in front of the application.

There are two possible ways to integrate WAF with your application.

- Integrate WAF in Web Server – you can use WAF like ModSecurity with Nginx, Apache, or WebKnight with IIS. This would be possible when you are hosting your website on your own, like in Cloud/VPS or dedicated. However, if you are on shared hosting, then you can't install it there.
- Use cloud-based WAF – probably, the easiest way to add site protection is by implementing the website firewall. The good thing is it will work for any website, and you can get it started in less than 10 minutes.