A

MINI PROJECT
REPORT ON

INTRUSION PROTECTION AGAINST SQL-INJECTION AND CROSS-SITE SCRIPTING USING REVERSE PROXY BASED AGENT

SUBMITTED TO THE SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE.

FOR

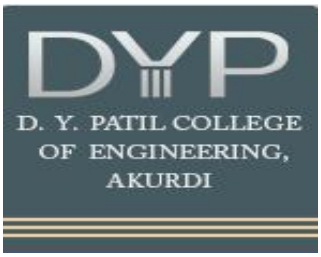
LAB PRACTICE III

Information and Cyber Security

BACHELOR OF ENGINEERING (COMPUTER ENGINEERING)

SUBMITTED BY

Name: Rutuja Meshram	**Exam Seat No: B150084281**
Name: Rutuja Vetal	**Exam Seat No: B150084348**
Name: Mohit Khedkar	**Exam Seat No: B150084268**



DEPARTMENT OF COMPUTER ENGINEERING
D.Y.PATIL COLLEGE OF ENGINEERING AKURDI, PUNE-44.

SAVITRIBAI PHULE PUNE UNIVERSITY, 2021-22 SEM-II

INDEX

1. ABSTRACT

The main objective is to protect a web application against SQL injection attack and Cross-Site Scripting attacks. Internet has eased the life of human in numerous ways, but the drawbacks like the intrusions that are attached with the internet applications sustains the growth of these applications. We have to develop a new policy based Proxy Agent, which classifies the request as a scripted request or query based request, and then, detects the respective type of attack..

2. INTRODUCTION

Problem Statement

SQL Injection attacks and Cross -Site Scripting attacks are the two most common attacks on web application. Develop a new policy based Proxy Agent, which classifies the request as a scripted request or query based request, and then, detects the respective type of attack, if any in the request. It should detect both SQL injection attack as well as the Cross-Site Scripting attacks.

Objective

- Confidentiality refers to access control of information to ensure that those who should not have access are kept out. This can be done with passwords, usernames, and other access control components.
- Integrity ensures that the information end-users receive is accurate and unaltered by anyone other than the site owner. This is often done with encryption, such as Secure Socket Layer (SSL) certificates which ensure that data in transit is encrypted.
- Availability rounds out the triad and ensures information can be accessed when needed. The most common threat to website availability is a Distributed Denial of Service attack or DDoS attack.

Scope and Limitation

This project demonstrates the need of a secure web application. It makes us understand Injection attacks. It also focussed on detection of SQL injection and Cross-site scripting attacks.

3. Software/Hardware Requirements

- **Software Requirement Specifications**
  Machine with Php and xammp server installed

- **Hardware Requirement Specification**

  Operating System:

  Windows OS,Mac OS,Linux mint,Ubuntu etc.

  Browser:

  Product runs and supports all browser and their versions :

  Google chrome,Firefox etc.

  Memory Requirement:

  Memory, 2 GB minimum, 4 GB recommended Processor:

  No gpu required and minimal processor works

THEORY CONCEPTS

What is Code Injection?

Code Injection is a type of attack in a web application, in which the attackers inject or provide some malicious code in the input data field to gain unauthorized and unlimited access, or to steal credentials from the users account. The injected malicious code executes as a part of the application. This results in either damage to the database, or an undesirable operation on the internet. Attacks can be performed within software, web application etc, which is vulnerable to such type of injection attacks. Vulnerability is a kind of lacuna or weakness in the application which can be easily exploited by attackers to gain unintended access to the data. Some common code injection attacks are HTTP Request Splitting Attacks, SQL Injection Attacks, HTML Injection Attacks, Cross-Site Scripting, Spoofing, DNS Poisoning etc.

What is SQL Injection Attack?

This attack occurs from malicious code being inserted into a string which is sent to the SQL Server for execution. A SQL Injection Attack usually starts with identifying weaknesses in the applications where unchecked users' input is transformed into database queries. This attack allows the attacker to access data from the database, which can be stolen or manipulated.

What is Cross-Site Scripting?
Cross-Site Scripting, or XSS, is another prevailing security flaw that Web applications are vulnerable to. In an XSS attack, the attacker is able to insert malicious code into a website. When this code is executed in a visitor's browser it can manipulate the browser to do whatever it wants. Typical attacks include installing malware, hijacking a user's session, or redirecting users to another site.
What is Reverse Proxy?
Reverse Proxy is a technique which is used to sanitize the user's inputs that may transform into a database attack. In this technique a filter program redirects the user's input to the proxy server before it is sent to the application server. At the proxy server, data cleaning algorithm is triggered using a sanitizing application.
Why Reverse Proxy?
A reverse proxy is used to sanitize the request from the user. When the request becomes high, more reverse proxys can be used to handle the request. This enables the system to maintain a low response time, even at high load.

5. Results

- **Code :**

```php
<?php
$hostname = "localhost";
$username = "root";
$password = "";
$dbname = "test";
$conn = mysqli_connect($hostname, $username,
$password, $dbname);
if(!$conn) {
    die("Unable to connect");
}
if($_POST) {
    $uname = $_POST["username"];
    $pass = $_POST["password"];
    //Making sure that SQL Injection doesn't
work
    // $uname =
mysqli_real_escape_string($conn,
$uname);//test or 1=1
    // $pass =
mysqli_real_escape_string($conn, $pass);
    // ---

    $sql = "SELECT * FROM logininfo WHERE
username = '$uname' AND password = '$pass'";
    $result = mysqli_query($conn, $sql);
    if(mysqli_num_rows($result) == 1) {
        echo "Welcome, user!";
    } else {
```

```
        echo "Incorrect Username/Password";
    }
}
?>
<!DOCTYPE html>
<html>
<head>
    <title>Login Portal</title>
    <style type="text/css">
        input[type=text],input[type=password]
{
            padding: 16px;
            margin: 8px;
            border: 1px solid #f1f1f1;
            letter-spacing: 1px;
            border-radius: 3px;
            width: 240px;
        }
        input[type=submit] {
            margin-left: 8px;
            width: 274px;
            border-radius: 3px;
            border: 1px solid #4285f4;
            background-color: #4285f4;
            padding: 16px;
            color: white;
            font-weight: 600;
            cursor: pointer;
```

```
        }
    </style>
</head>
<body>
    <form action method="POST"
autocomplete="off">
        <input type="text" name="username"
placeholder="Username" /><br />
        <input type="password"
name="password" placeholder="********" /><br
/>
        <input type="submit" name="login"
value="LOGIN" />
    </form>
</body>
</html>
```

6. CONCLUSION

Thus we have successfully understood the importance of security in a web application against injection attacks. We have learnt how reverse proxy based agent classifies the request as a scripted request or query based request, and then, detects the respective type of attack