Conducting a Cross Site Scripting (XSS) attack

Lab Objective:

Testing a website for an XSS vulnerability – Cross site scripting.

Lab Purpose:

XXS is a common vulnerability in web applications and is frequently listed as a top vulnerability in the OWASP top ten. XXS occurs when web applications execute JavaScript, which is input into the form sections of a web application. The applications perform no security checks on the entered data. It simply passes it straight to the server, causing inputted JavaScript to execute.

Lab Tool:

Web browser

Lab Topology:

Any web browser of your choosing for this lab.

STEP ONE:

Let's begin by navigating to the following URL:

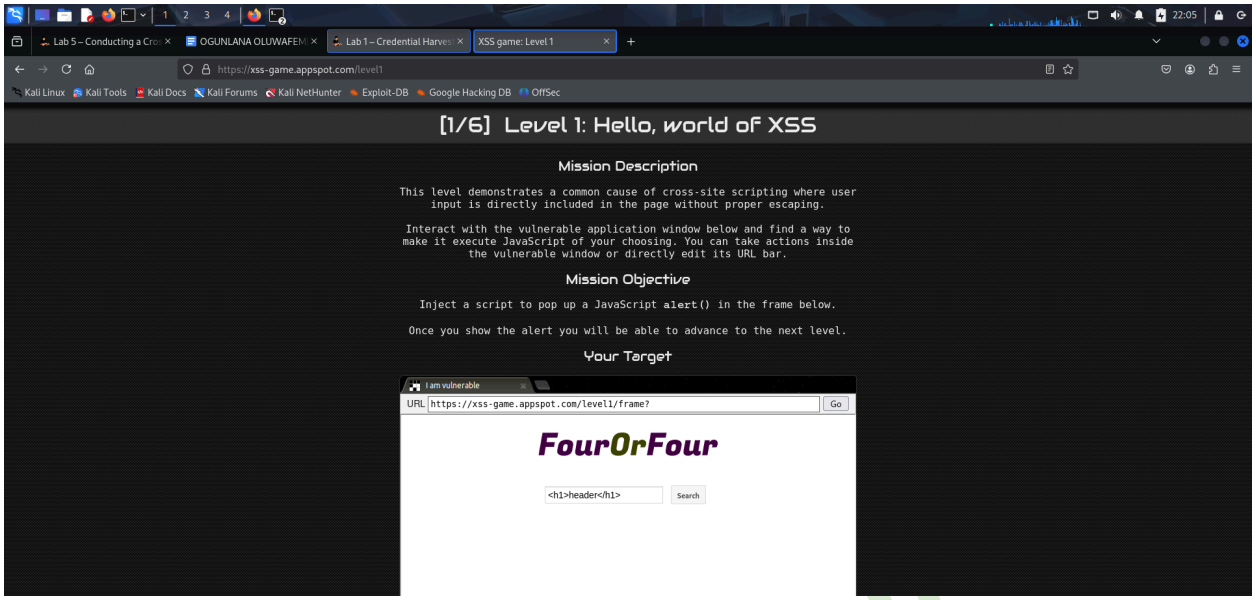
https://xss-game.appspot.com/level1

This is the first level. We are presented with a simple search box for a web page.

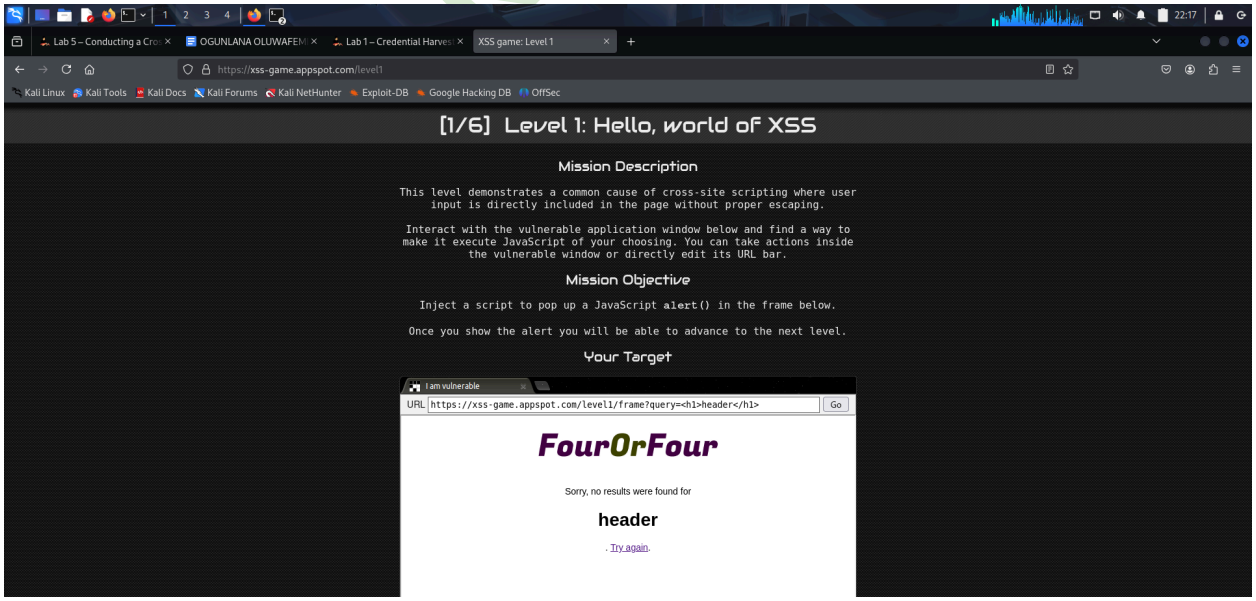
To be able to execute JavaScript in a web application like this one, a basic understanding of the syntax for JavaScript and HTML is required.

For example,

Input "<h1>header</h1>" to ascertain if the site is vulnerable to XSS attack.

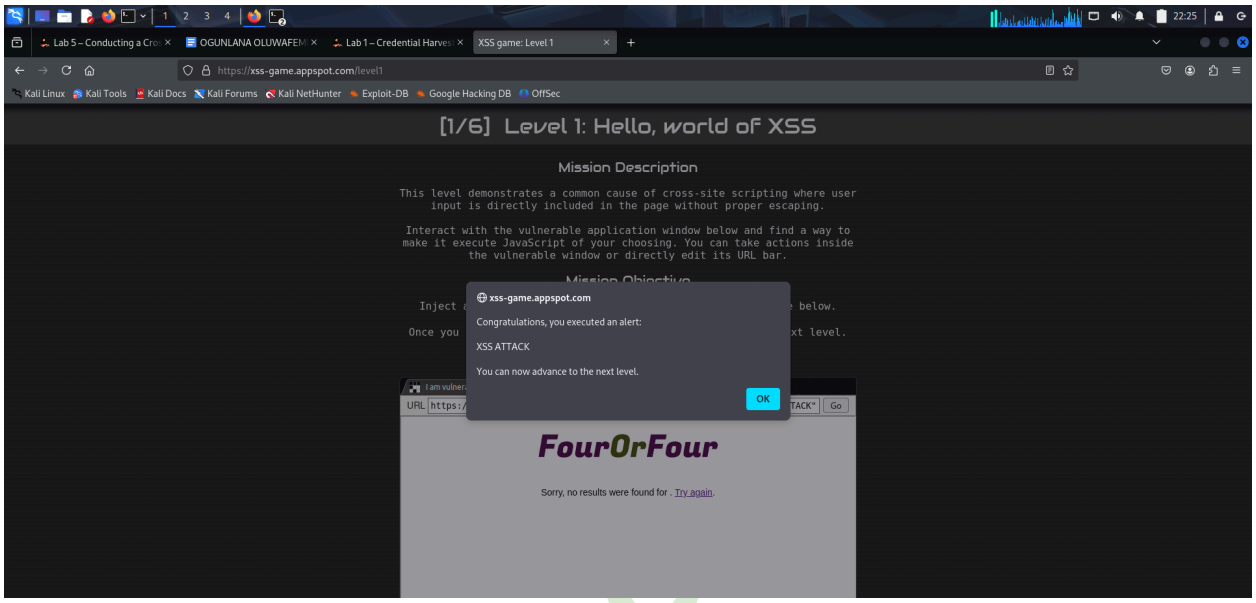


- If the website displays "header" as a large heading, it means the website is reflecting your input directly onto the page without checking if it's safe.
- This is a sign that the website might be vulnerable to XSS.



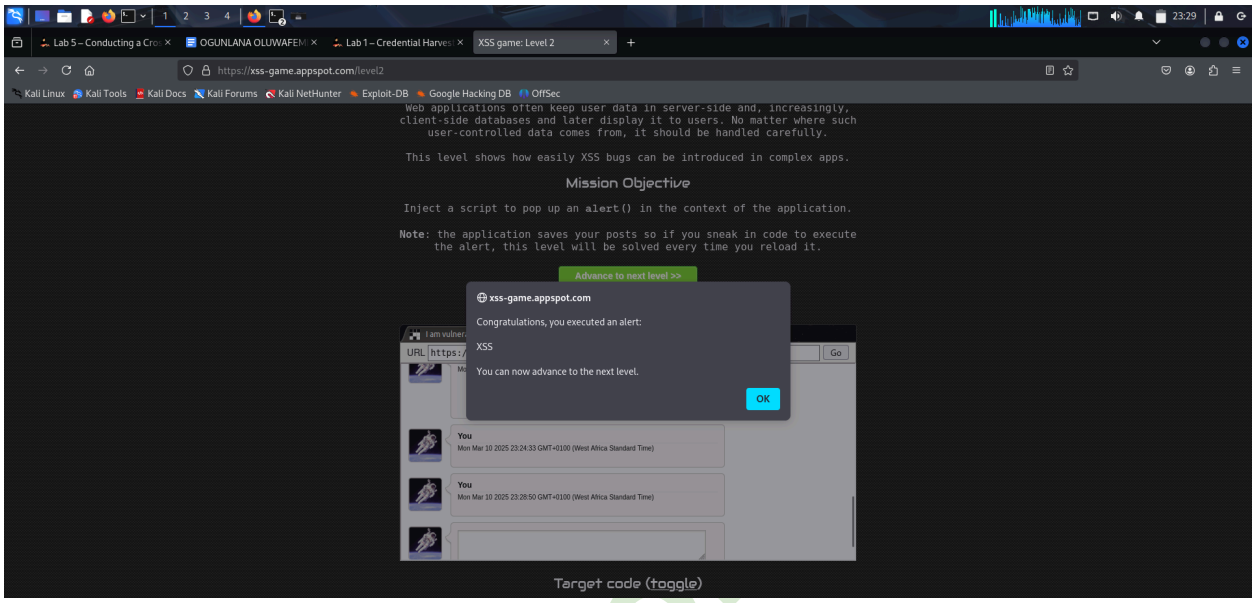
STEP TWO:

To execute the XSS attack. Now, let’s try to inject JavaScript.
Enter “<script>alert(”XSS ATTACK”)</script>” into the search box. This is a simple JavaScript code that creates a pop-up alert with the message "XSS ATTACK".



STEP THREE:

- Go to **Level 2**: https://xss-game.appspot.com/level2.
- You'll see a forum where you can post messages.
- Enter the following into the message box: <script>alert("XSS");</script>



Every time someone views the page, the script will run, and they'll see a pop-up alert with the message "XSS".

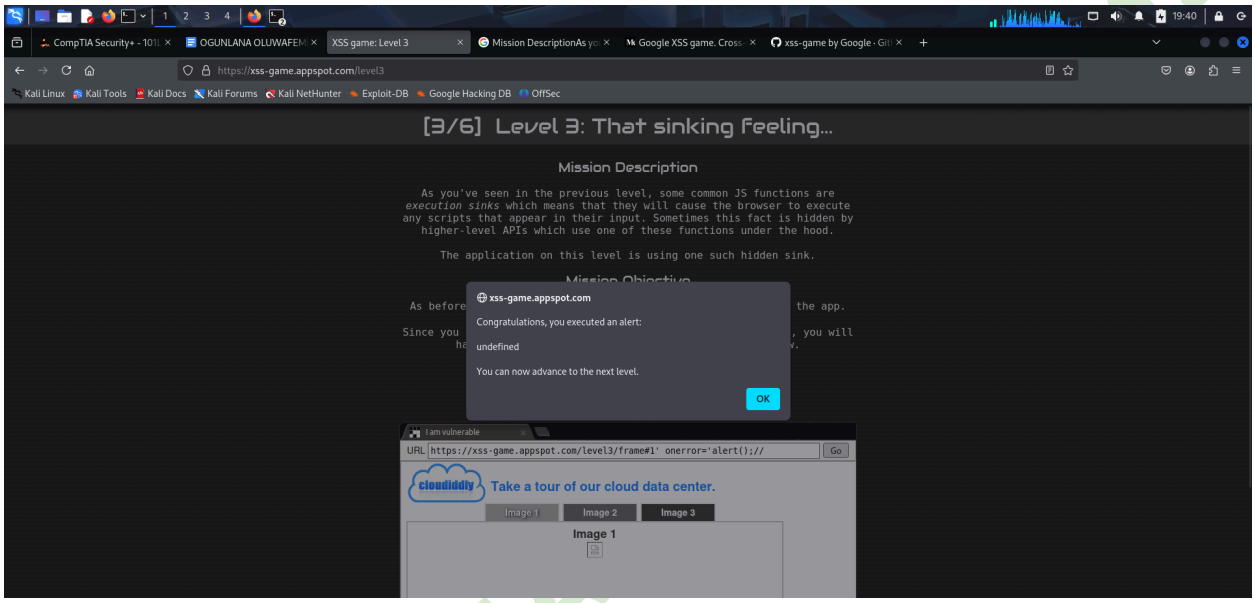
STEP FOUR:

Go to **level 3:**https://xss-game.appspot.com/level3

In this mission, the website is vulnerable to XSS, but there's no input box where one can inject the script. Instead, the injection is done by modifying the **URL** in the browser's address bar.

Modify the URL by typing;

https://xss-game.appspot.com/level3/frame#1' onerror='alert();//



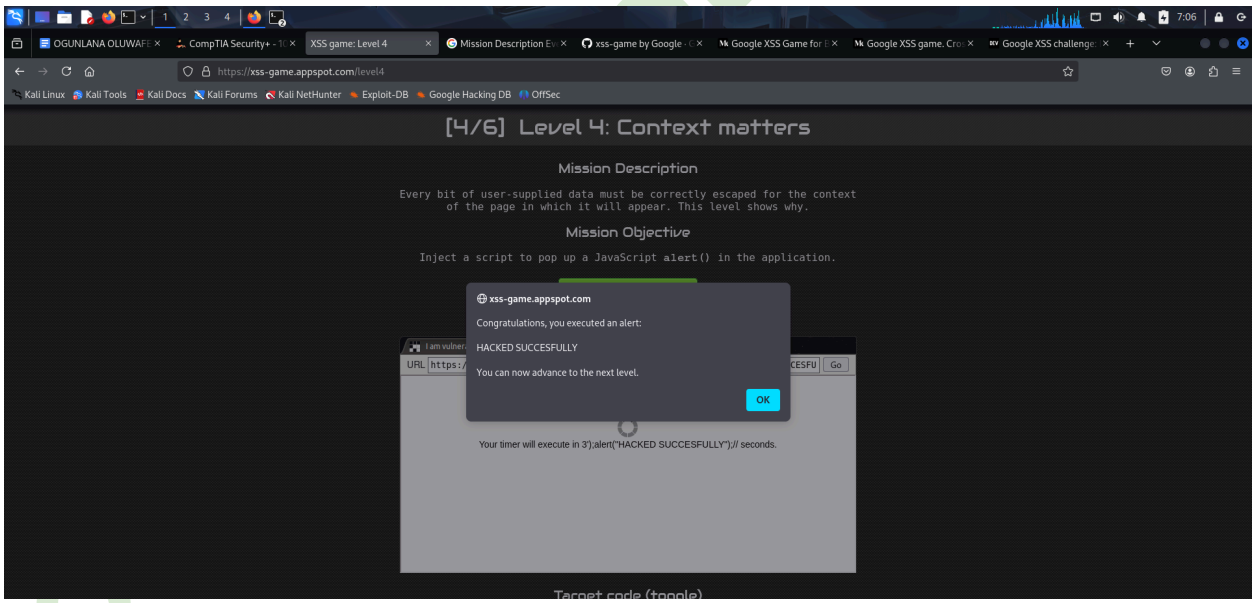
STEP FIVE:

Go to **level 4: https://xss-game.appspot.com/level4**

Inject code ***3');alert("HACKED SUCCESFULLY");//*** into the comment box

The code `3');alert("HACKED SUCCESSFULLY");//` is designed to **break out of a JavaScript string** and execute your own code. Here's what each part does:

1. `3');`:
    - Closes the existing string and ends the current JavaScript statement.
2. `alert("HACKED SUCCESSFULLY")`:
    - This is the malicious code you want to run. It will display a pop-up with the message "HACKED SUCCESSFULLY."
3. `//`:
    - This is a comment in JavaScript. It tells the browser to ignore everything after it, preventing errors.



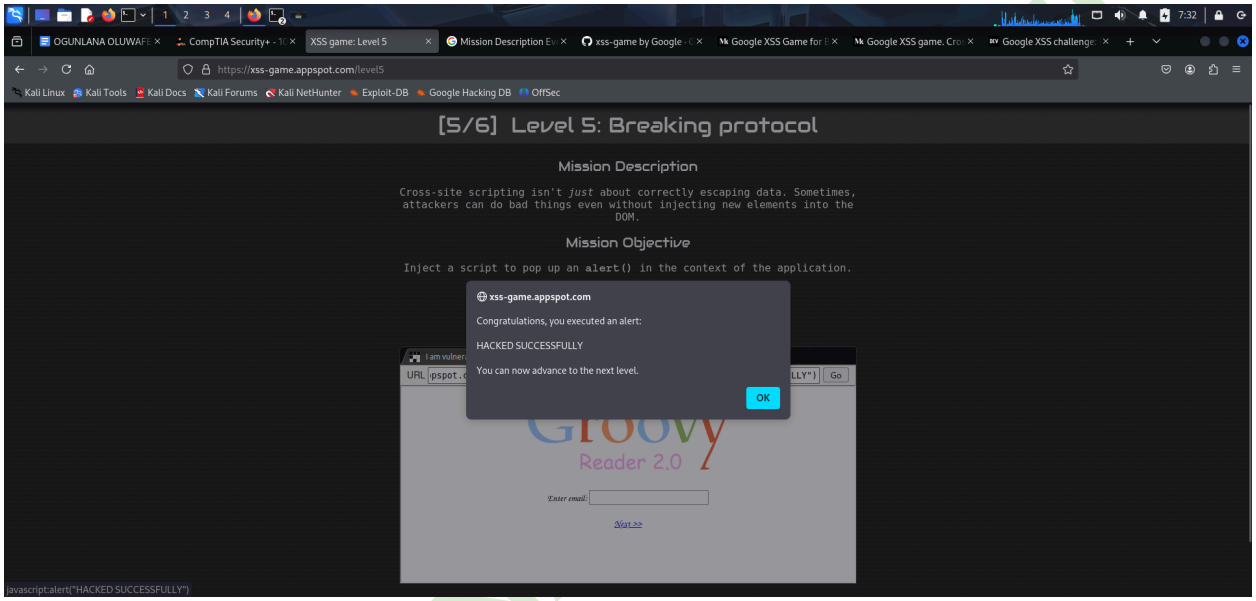
STEP SIX:

The objective is to Inject a script to pop up an `alert()` in the context of the application.

Go to: **https://xss-game.appspot.com/level5**

Tweak the URL to **https://xss-game.appspot.com/level5/frame/signup?next=javascript:alert("HACKED SUCCESSFULLY")**

Then click on “next” icon



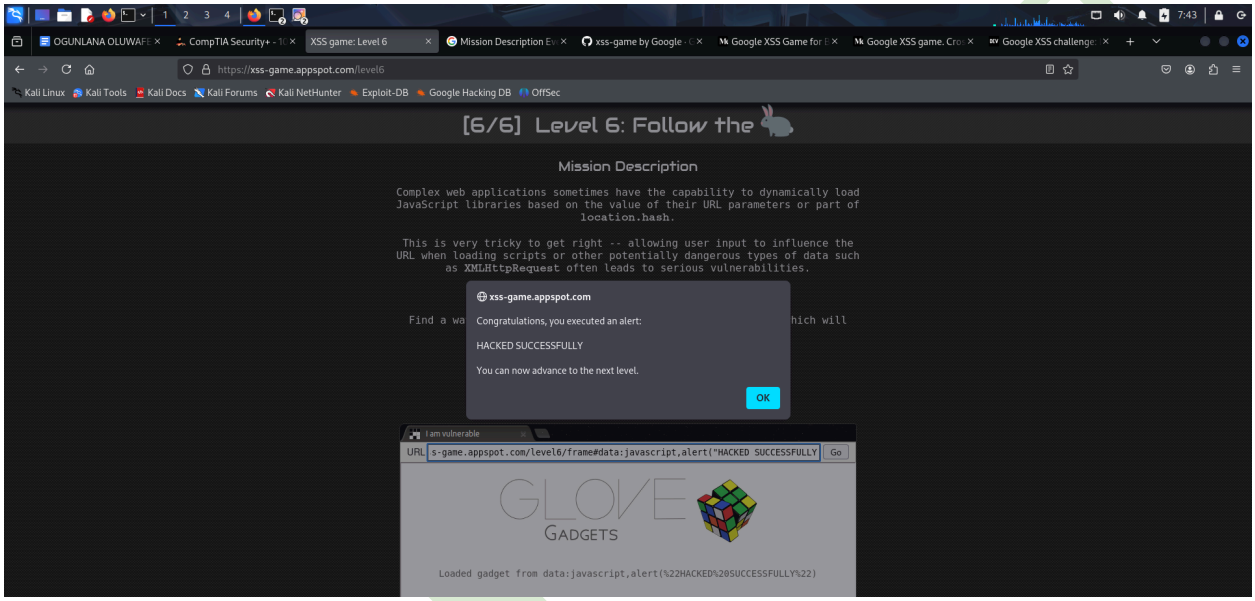
STEP SEVEN:

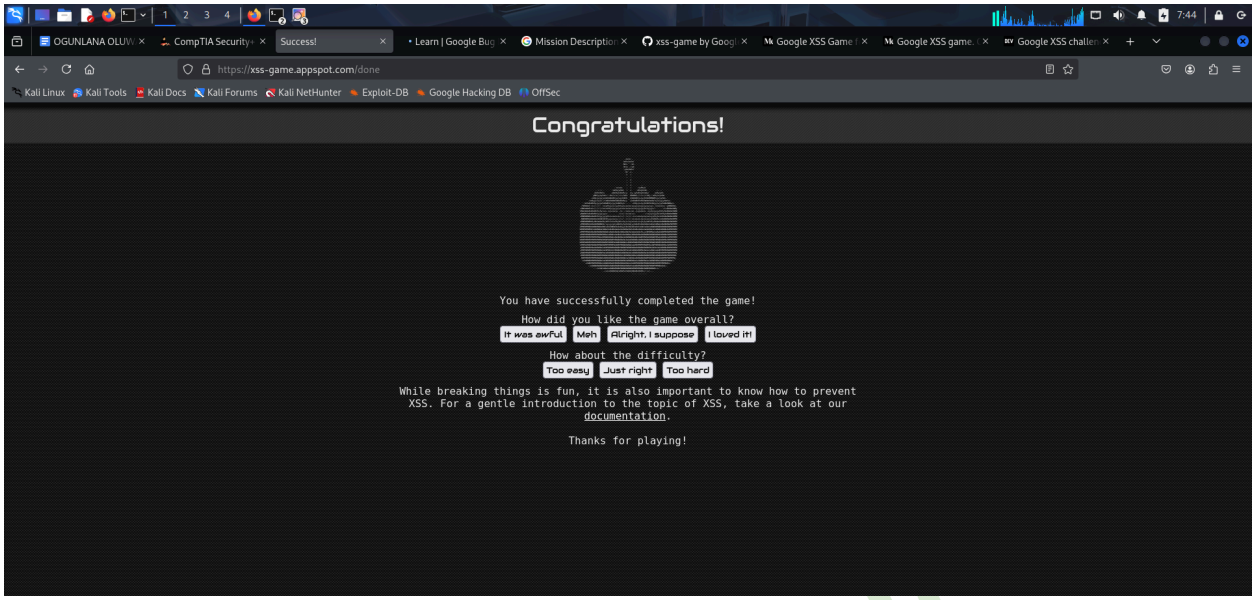
The objective is to Find a way to make the application request an external file which will cause it to execute an `alert()`.

Go to: **https://xss-game.appspot.com/level6**

Tweak the URL to;

https://xss-game.appspot.com/level6/frame#data:javascript,alert("HACKED SUCCESSFULLY")





END