CONDUCTING A CROSS-SITE SCRIPTING (XXS) ATTACK

TOOL: WEB BROWSER

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

Cross-Site Scripting (XSS) is a type of security vulnerability typically found in web applications that allows an attacker to inject malicious scripts into content delivered to users. This can happen when an application includes untrusted data in a webpage without proper validation or escaping.

Types of XSS:

1. **Stored XSS**: The malicious script is stored on the server (e.g., in a database) and is served to users when they request a page that includes this data. It can affect many users over time.
2. **Reflected XSS**: The malicious script is reflected off a web server, typically through a URL or request parameter. This type is often delivered via phishing emails or links.
3. **DOM-based XSS**: The vulnerability is in the client-side code (JavaScript), where the modification of the Document Object Model (DOM) can lead to the execution of malicious scripts.

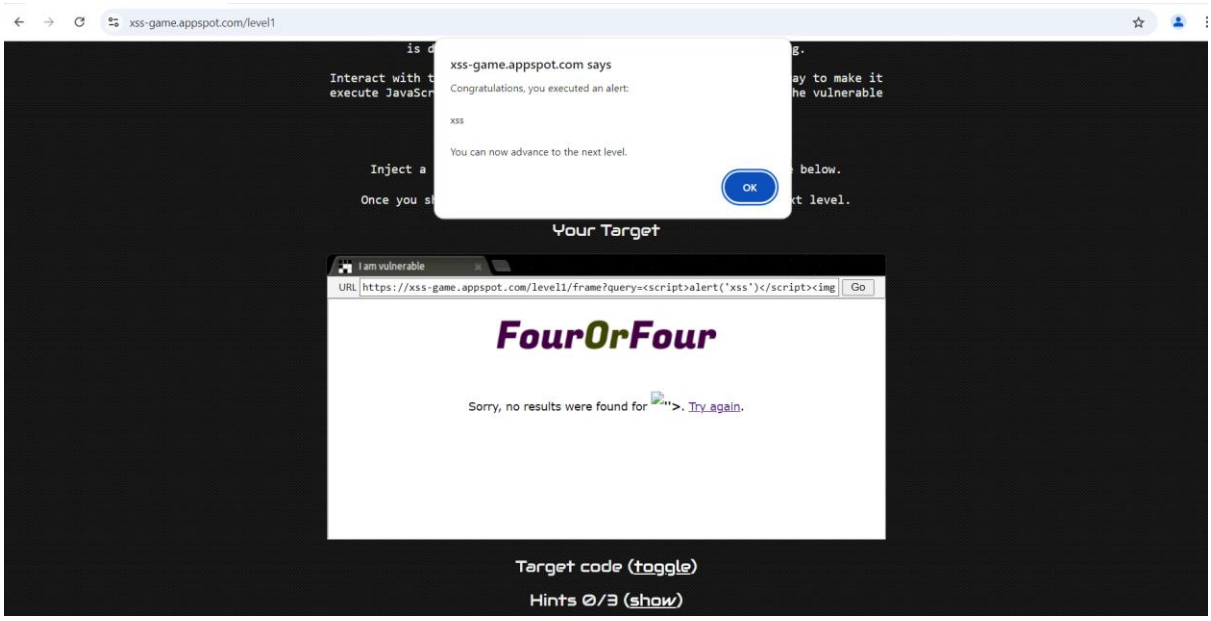
Vulnerabilities and Risks:

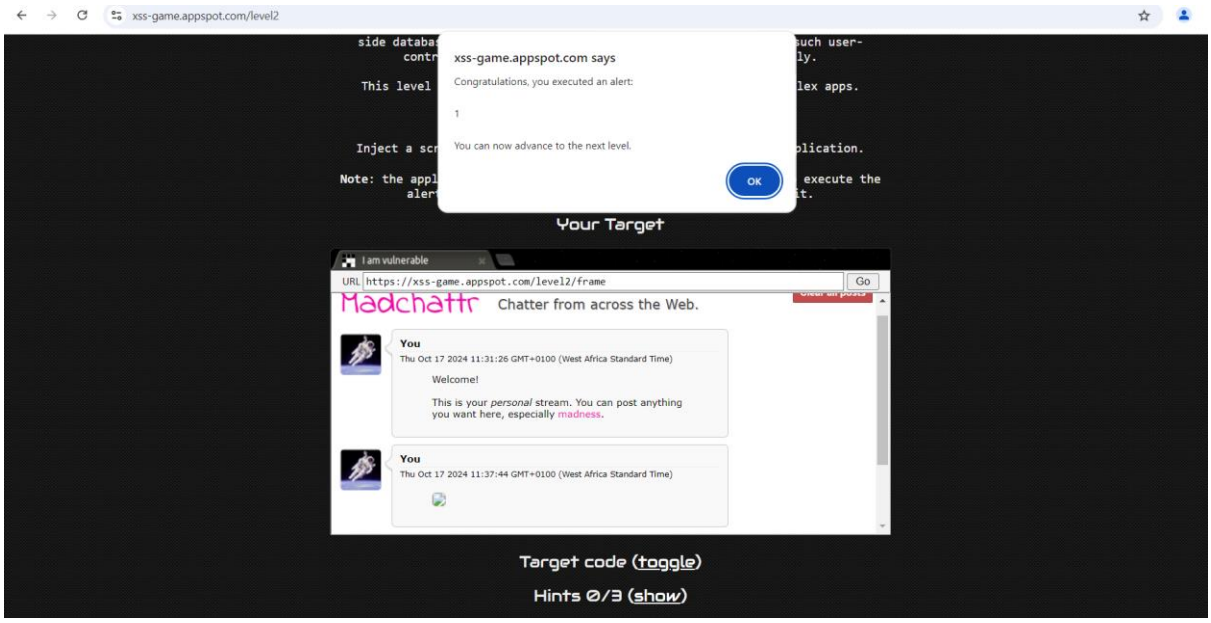
- **Data Theft**: Attackers can steal sensitive information like cookies, session tokens, or other private data.
- **Account Takeover**: By stealing authentication tokens, attackers can impersonate users.
- **Malware Distribution**: XSS can be used to distribute malware by redirecting users to malicious sites.
- **Defacement**: Attackers can modify the content displayed to users, damaging the reputation of the site.
- **Phishing**: Attackers can create fake forms to collect sensitive user information.

Prevention Measures:

1. **Input Validation**: Validate and sanitize all user inputs to ensure they don't contain harmful scripts.
2. **Output Encoding**: Encode data before rendering it in the browser to prevent execution (e.g., using HTML entity encoding).
3. **Content Security Policy (CSP)**: Implement CSP to restrict sources of executable scripts.
4. **HTTPOnly Cookies**: Use the HTTPOnly flag on cookies to prevent access via JavaScript.
5. **Regular Security Audits**: Conduct regular security assessments to identify and fix vulnerabilities.

By implementing these measures, developers can significantly reduce the risk of XSS attacks in their applications.





xss-game.appspot.com/level3

Mission Description

As you've seen in the previous level, some common JS functions are *execution sinks* which means that they will cause the browser to execute any scripts that appear in their input. Sometimes this fact is hidden by higher-level APIs which use one of these functions under the hood.

The application on this level is using one such hidden sink.

Mission Objective

As before, inject a script to pop up a JavaScript `alert()` in the app.

Since you can't enter your payload anywhere in the application, you will have to manually edit the address in the URL bar below.

Your Target