

Esercizio extra attacchi XSS (Cross-Site Scripting)

Livello 1

[1/6] Level 1: Hello, world of XSS

Mission Description

This level demonstrates a common cause of cross-site scripting where user input is directly included in the page without proper escaping.

Interact with the vulnerable application window below and find a way to make it execute JavaScript of your choosing. You can take actions inside the vulnerable window or directly edit its URL bar.

Mission Objective

Inject a script to pop up a JavaScript `alert()` in the frame below.

Once you show the alert you will be able to advance to the next level.

Your Target



Target code ([toggle](#))

Hints 0/3 ([show](#))

Per superare questo livello basta semplicemente scrivere un piccolo script come:

```
<script> alert('attacco in corso')</script>
```

Nella barra di ricerca e l'attacco xss andrà a buon fine

Livello 2

[2/6] Level 2: Persistence is key

Mission Description

Web applications often keep user data in server-side and, increasingly, client-side databases and later display it to users. No matter where such user-controlled data comes from, it should be handled carefully.

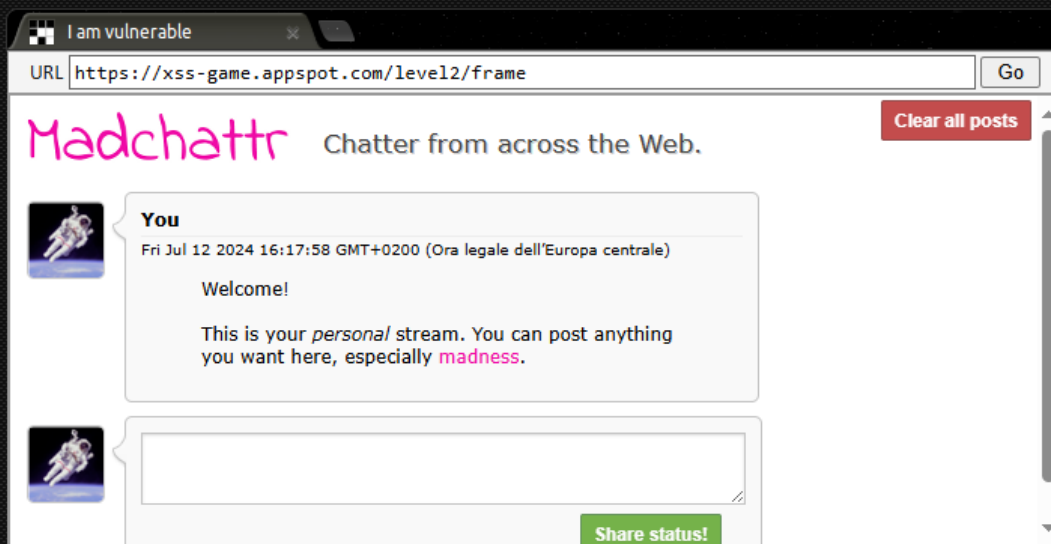
This level shows how easily XSS bugs can be introduced in complex apps.

Mission Objective

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

Note: the application saves your posts so if you sneak in code to execute the alert, this level will be solved every time you reload it.

Your Target



Target code ([toggle](#))

Hints 0/3 ([show](#))

Per superare il secondo livello non basta un semplice script come prima ma bisognerà cambiare tattica useremo "img" questa tecnica sfrutta il fatto che un browser eseguirà il codice JavaScript associato all'attributo "onerror" quando non riesce a caricare l'immagine specificata. Possiamo infatti superare il livello usando una stringa del tipo:

```
<img src='immagine' onerror="alert('attacco in corso')">
```

[3/6] Level 3: That sinking feeling...

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.

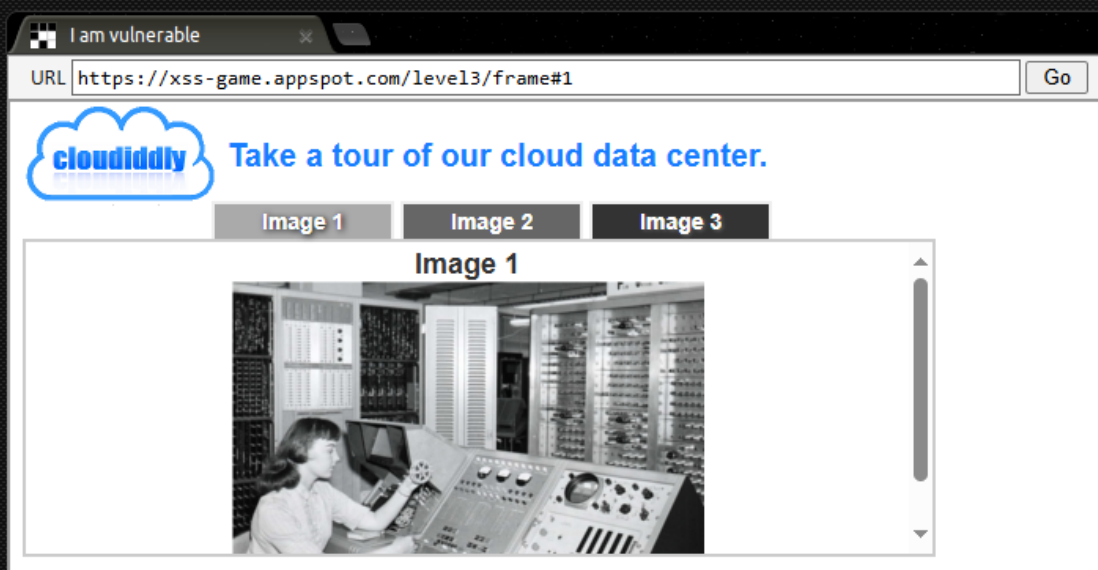
The application on this level is using one such hidden sink.

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



Target code (toggle)

Hints 0/4 (show)

Per superare questo livello dobbiamo procedere in modo simile al precedente. Come possiamo vedere, ci sono già delle immagini caricate sul sito e dall'URL possiamo vedere che sono divise da un #. Possiamo sfruttare questa debolezza per far caricare al sito il nostro script attraverso l'URL inserendolo dopo il # al posto del numero dell'immagine con un codice come:

`https://xss-game.appspot.com/level3/frame#`

Livello 4

[4/6] Level 4: Context matters

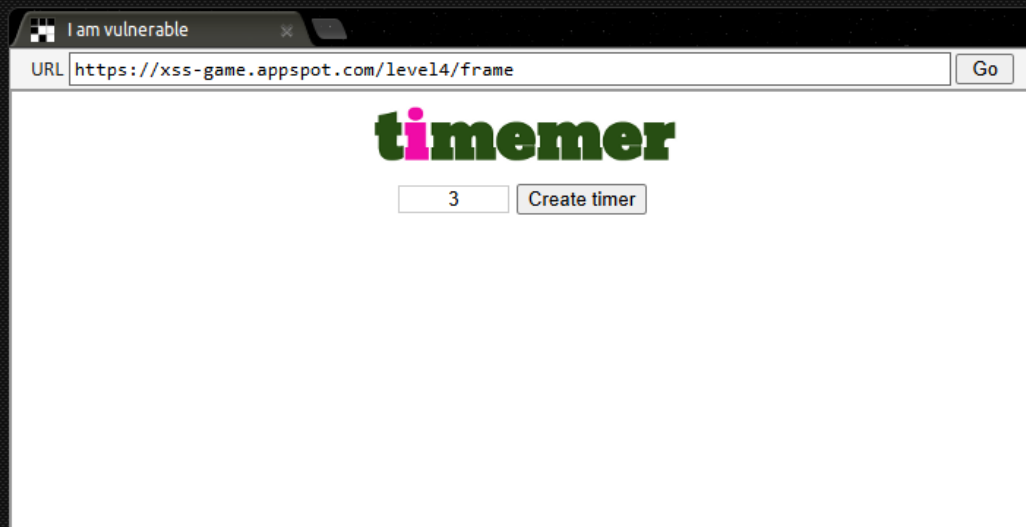
Mission Description

Every bit of user-supplied data must be correctly escaped for the context of the page in which it will appear. This level shows why.

Mission Objective

Inject a script to pop up a JavaScript `alert()` in the application.

Your Target



Target code ([toggle](#))

Hints 0/3 ([show](#))