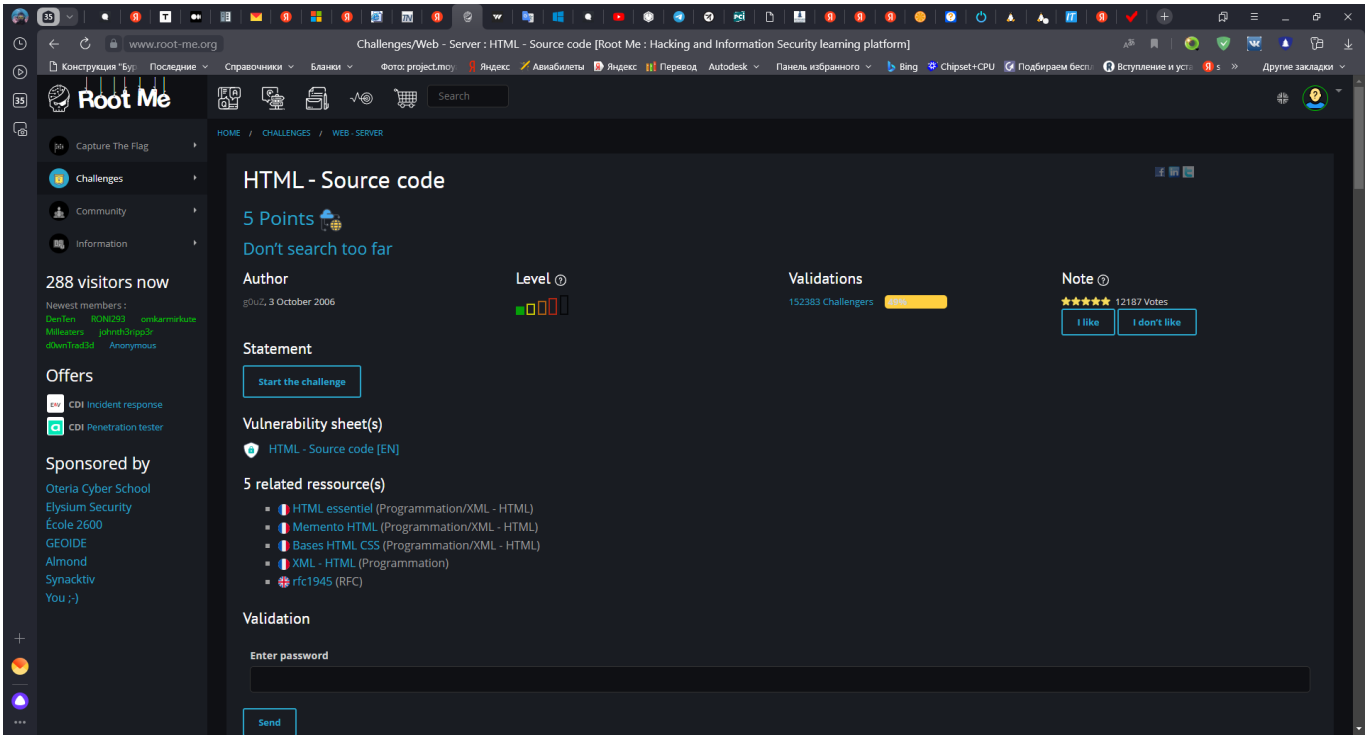
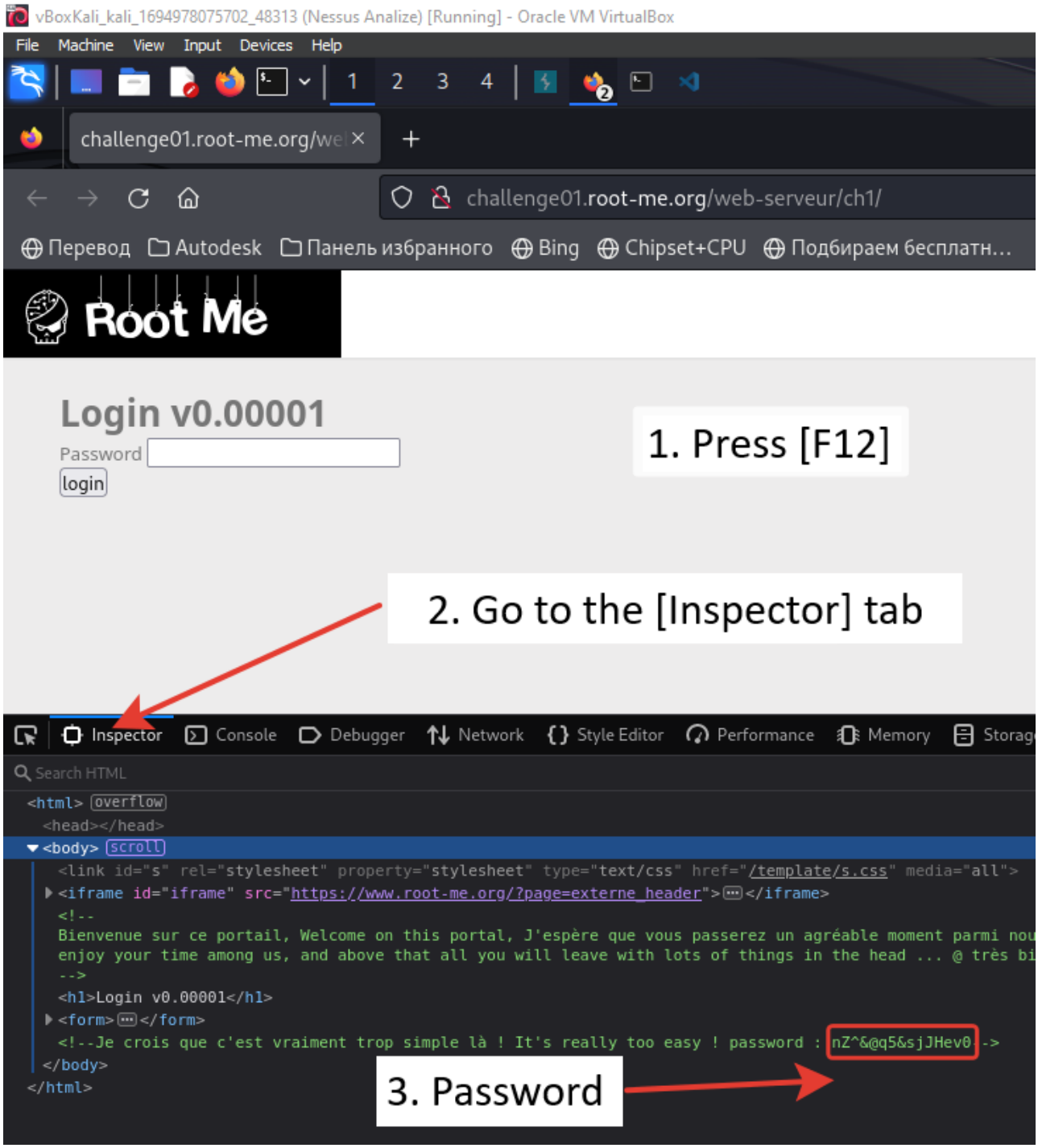
Web Application Security Testing -> **RootMe (HTML_SourceCode, HTML_UserAgent, WeakPassword, HTTP_Headers, HTTP_POST, HTTP_VerbTampering)**

Root Me (HTML - Source code)

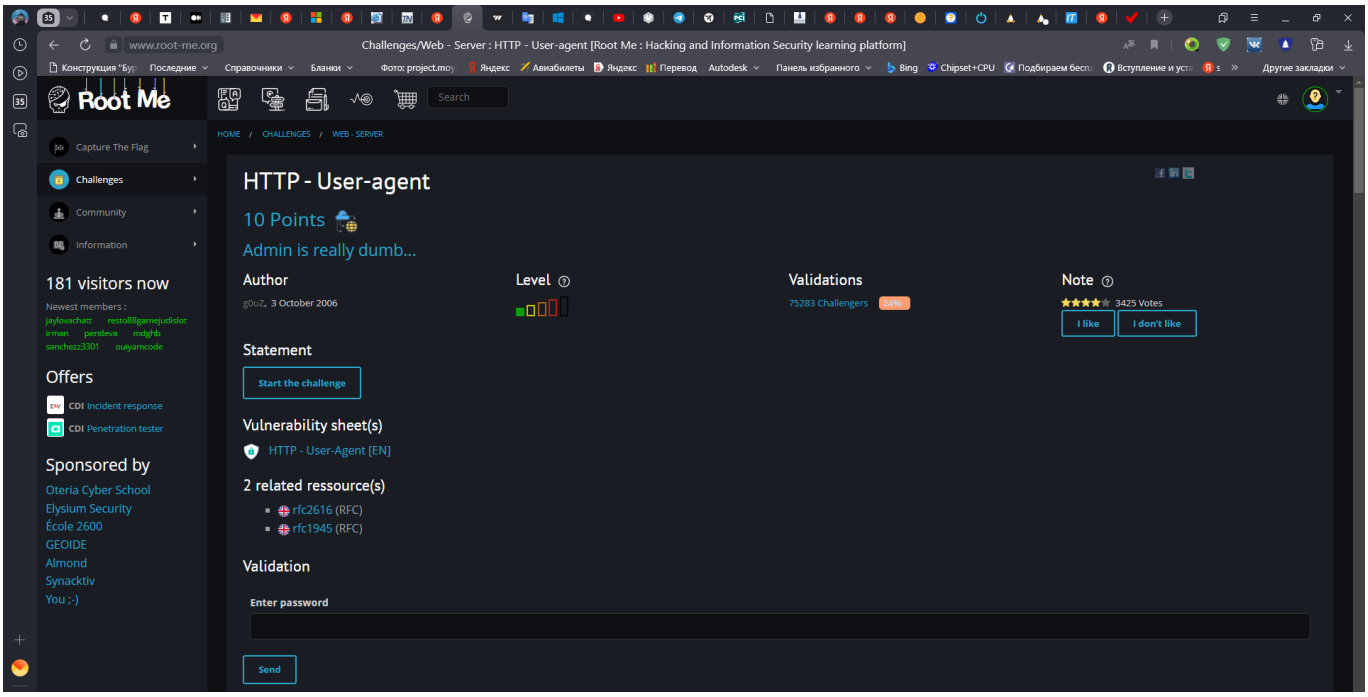


Solution:

1. Press the `[F12]` key to open the developer tools.
2. Go to the `[Inspector]` tab to research the source code.
3. The html page contains a commented out password.

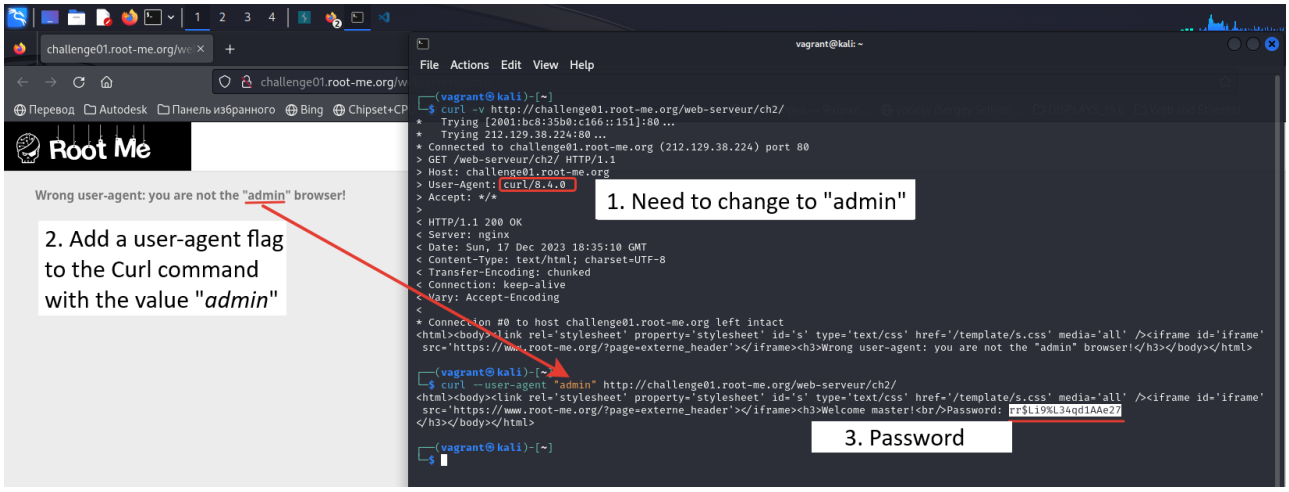


Root Me (HTML - User-agent)

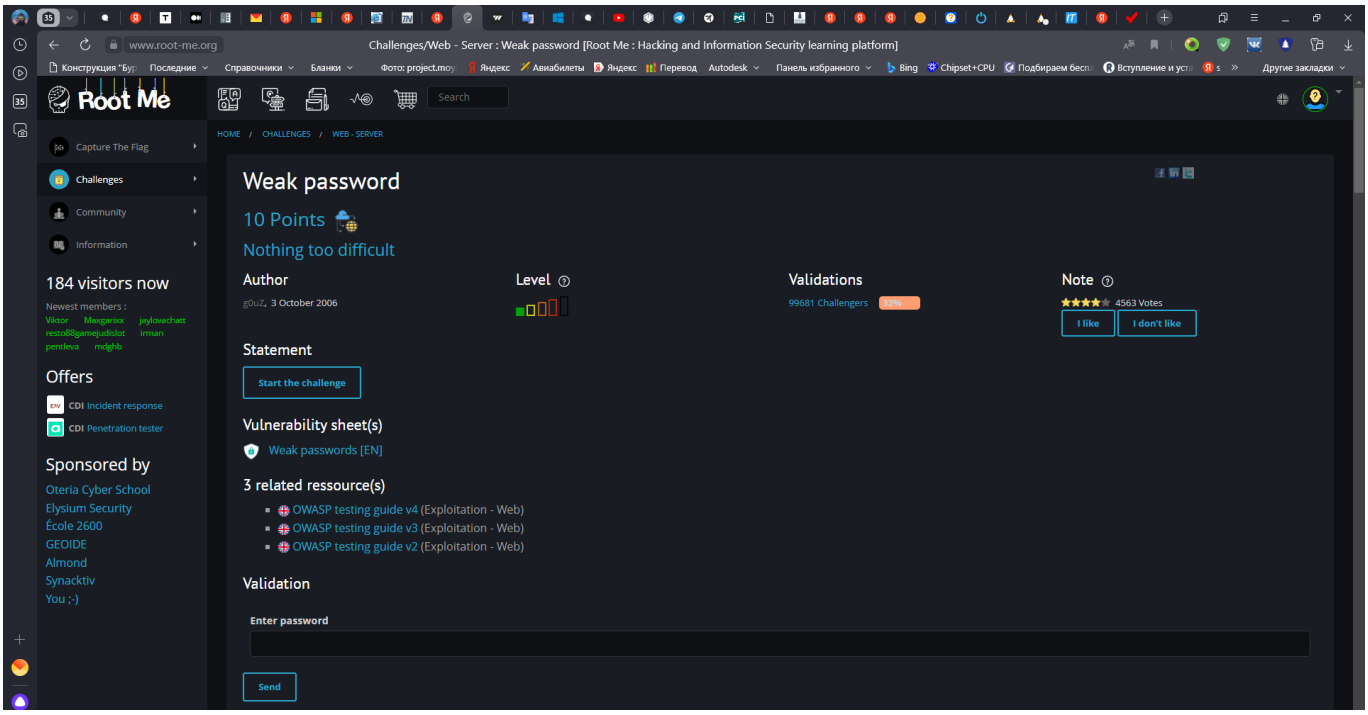


Solution

1. On the challenge page there is a hint that we are using the wrong browser.
2. If we look at the server response, we can verify that the User-agent contains `curl/8.4.0`, but we must use the browser "*admin*". Of course, to solve this problem, we should change the User-Agent request header.
3. Add the **--user-agent** flag with the modified header "*admin*".
4. Collect the flag

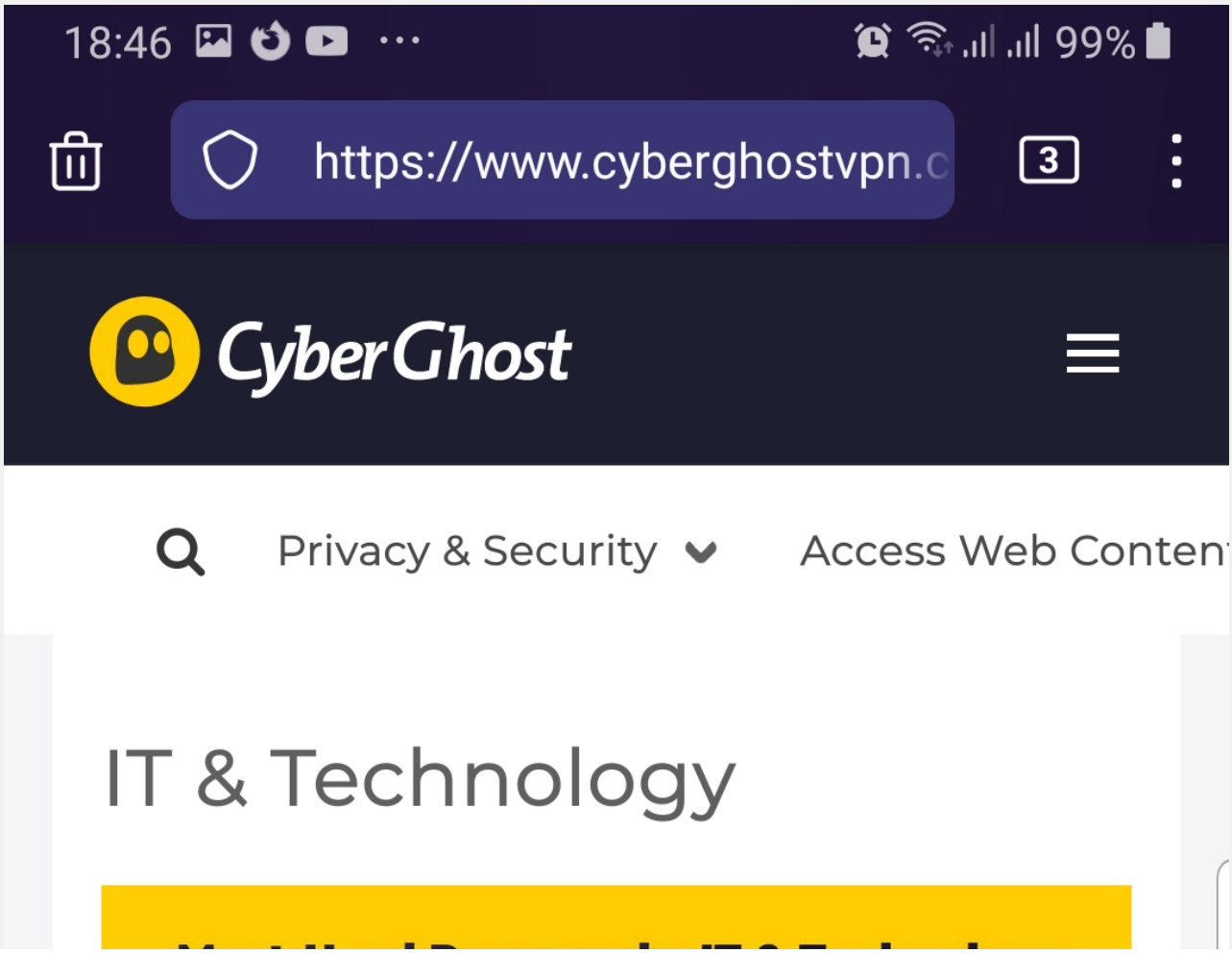


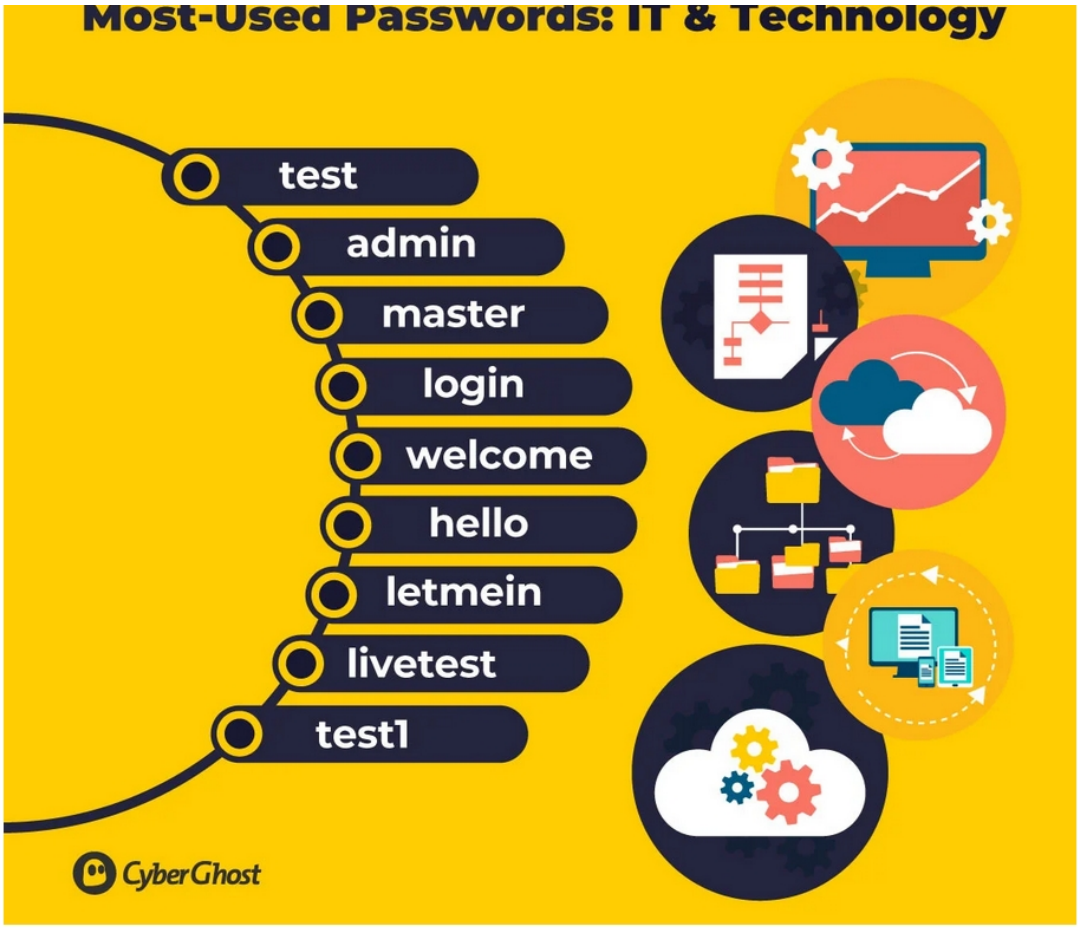
Root Me (Weak password)



This challenge can be solved in 2 ways:

1. Brute force method
2. Use the lists of "*Most-used Passwords*"

> **NOTE:**
>
> For example, I used 10 lists from CyberGhost
>
> 
>



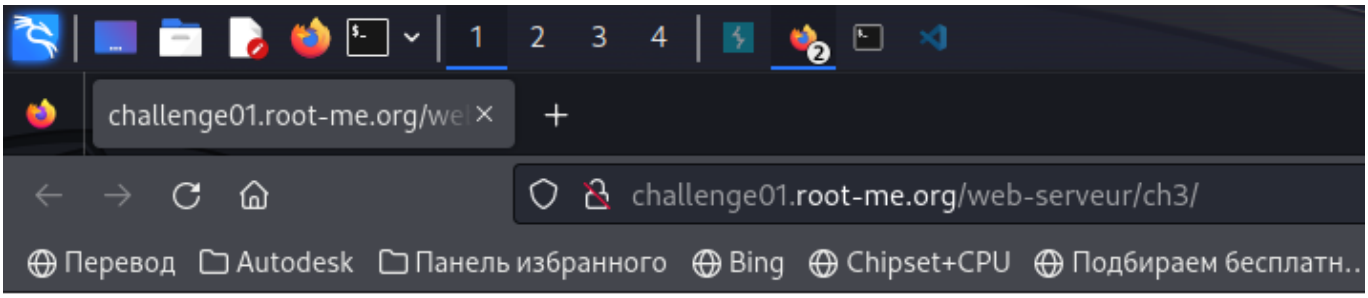
The constant evolution of technology coincides quite naturally with the rise in tech-related passwords. This is likely linked to the ever-growing necessity for testing by IT departments.

Honestly, every time the first thing I do is try the following pairs:

- `admin`/`''`,
- `admin`/`admin`, <- valid
- `admin`/`password`,
- `admin`/`password1`,
- `admin`/`password123`,
- `admin`/`passw0rd`,
- `admin`/`passwd`,

- `root/''`,
- `root/root`,
- `root/toor`,
- `root/password`,
- `root/passw0rd`,
- etc.

After manually entering the username - `admin` and password - `admin` I logged in.

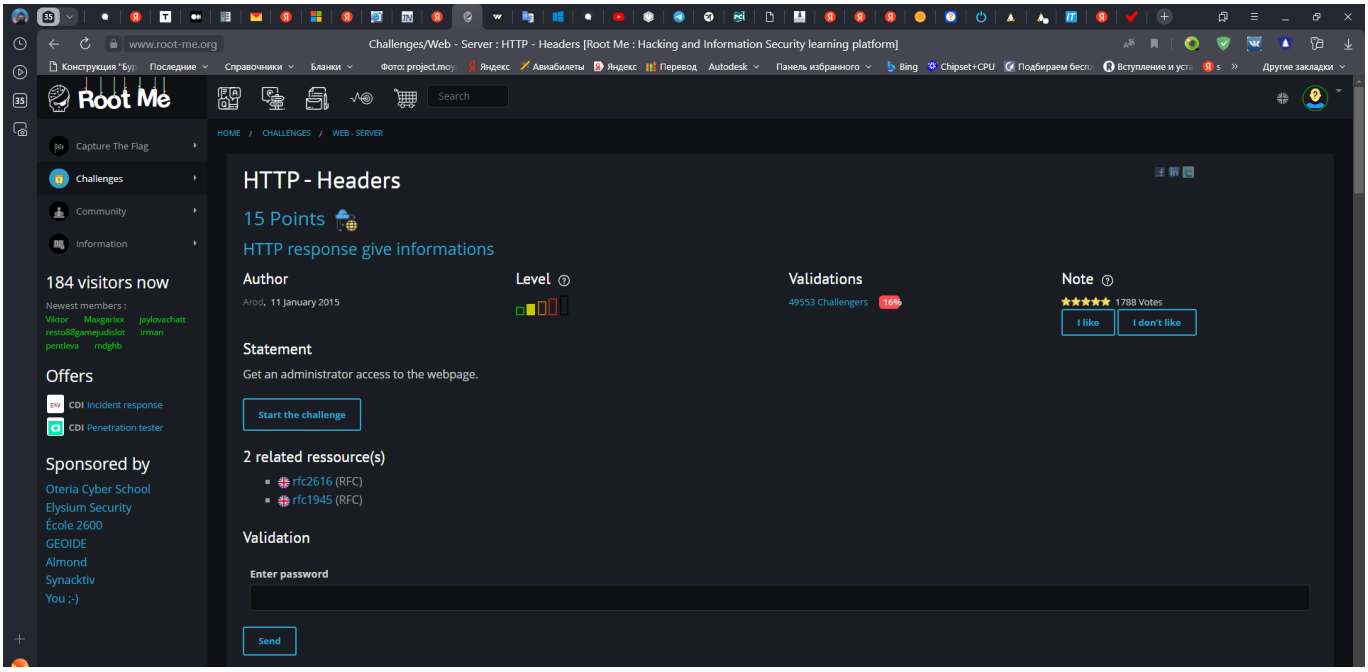


Bien joué, vous pouvez utiliser ce mot de passe pour valider le challenge

User: *admin* **password:** *admin*

Well done, you can use this password to validate the challenge

Root Me (HTTP - Headers)



Solution

1. On the challenge page there is a hint that the content isn't only part of an HTTP response.

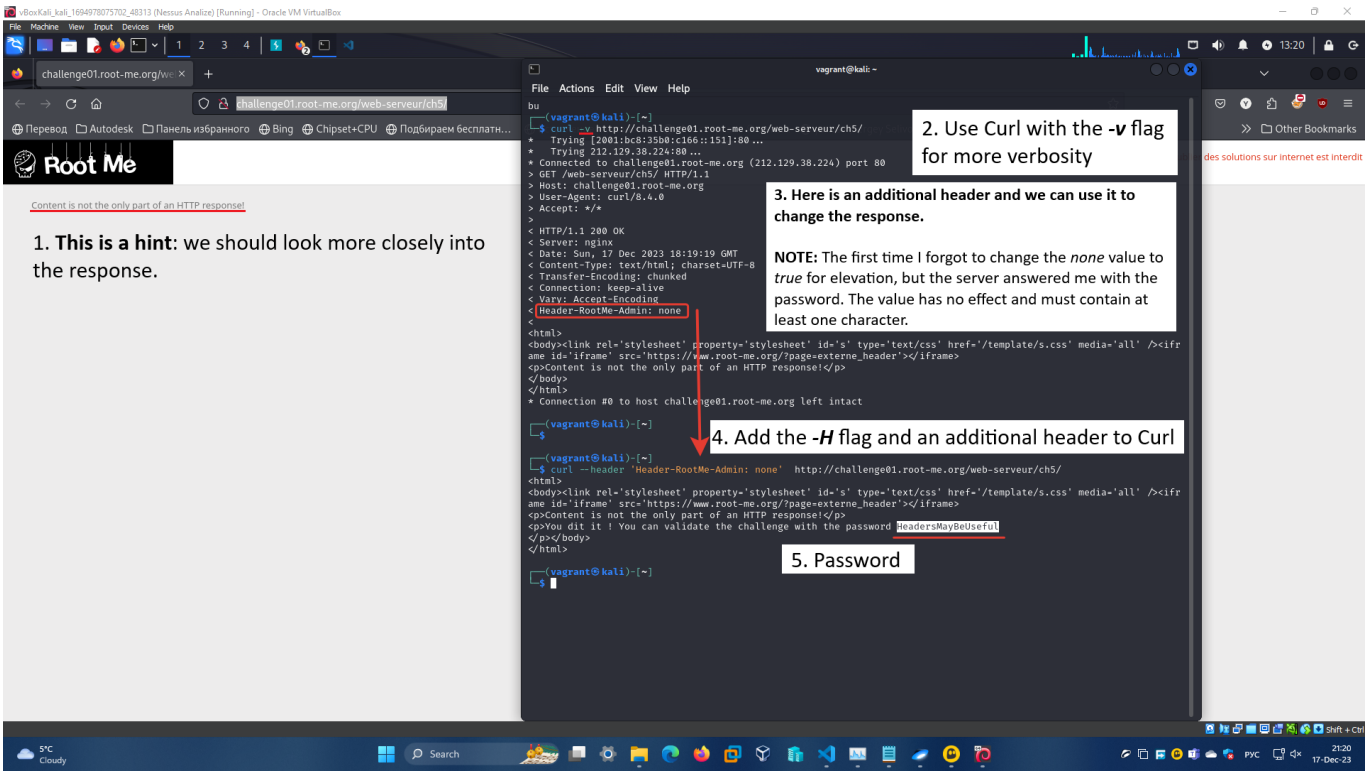
2. Use Curl with the **-v** flag for more verbosity.

3. Here is an additional header and we can use it to change the response.

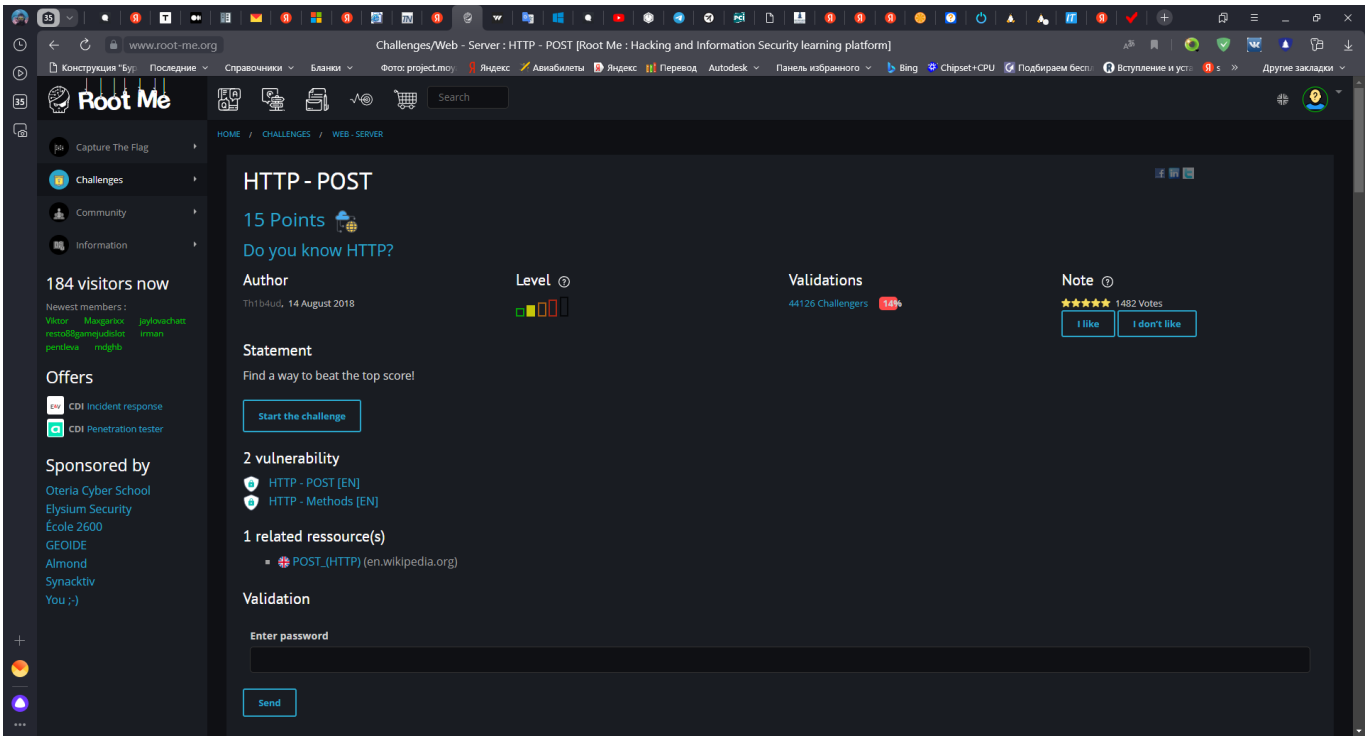
4. Add the **-H** or **--header** flag to modify the additional header with the value of `Header-RootMe-Admin: true` to Curl.

> **NOTE:**
>
> The first time I was in a hurry and copied a line from the answer, I forgot to change the `none` value to `true` (*for elevation*) and press `[Enter]`. This is amazing, the server responded to me with a password page. I sent a few additional requests and found that the value had no effect and must contain at least one character.

5. The response contains the password.



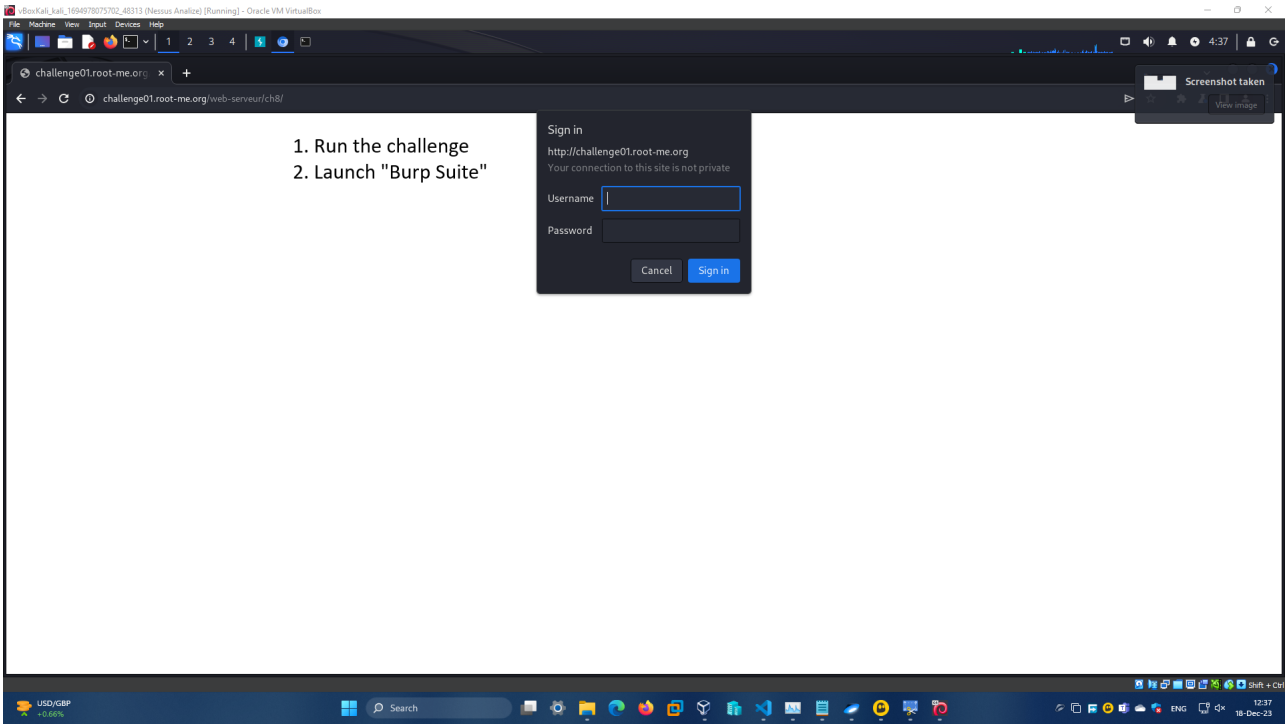
Root Me (HTTP - POST)



Solution

> **NOTE:**
>
> To solve this problem, Curl is not enough, you need to use a proxy (*e.g.: Burp Suite*).

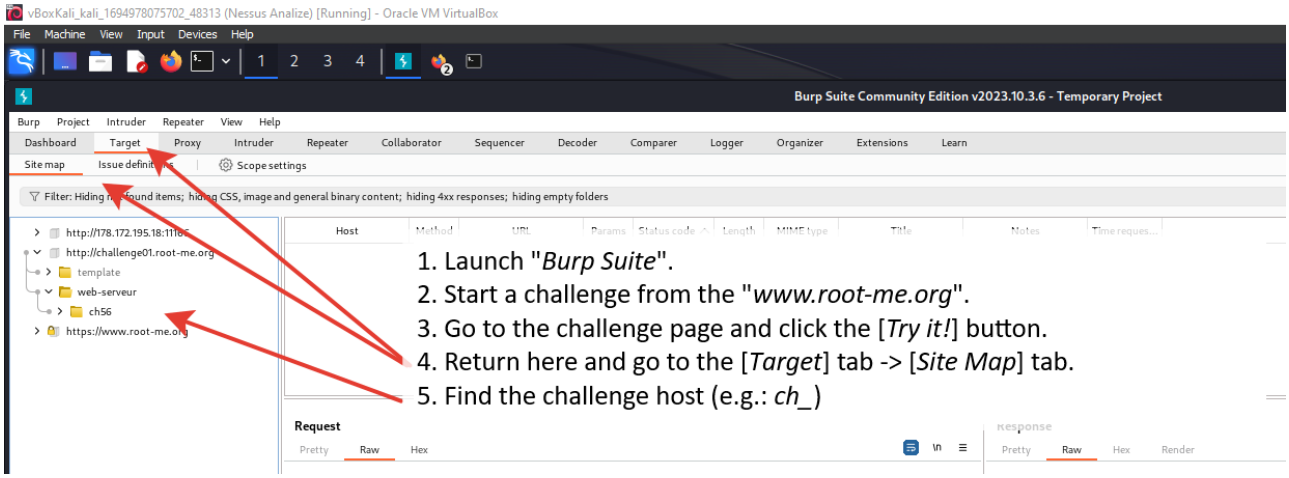
1. Run the challenge
2. Launch "*Burp Suite*"



3. Start a challenge from the "<*www.root-me.org*>".
4. Go to the challenge page and click the `[Try it!]` button.

5. Return here and go to the `[Terget]` tab -> `[Site Map]` tab.

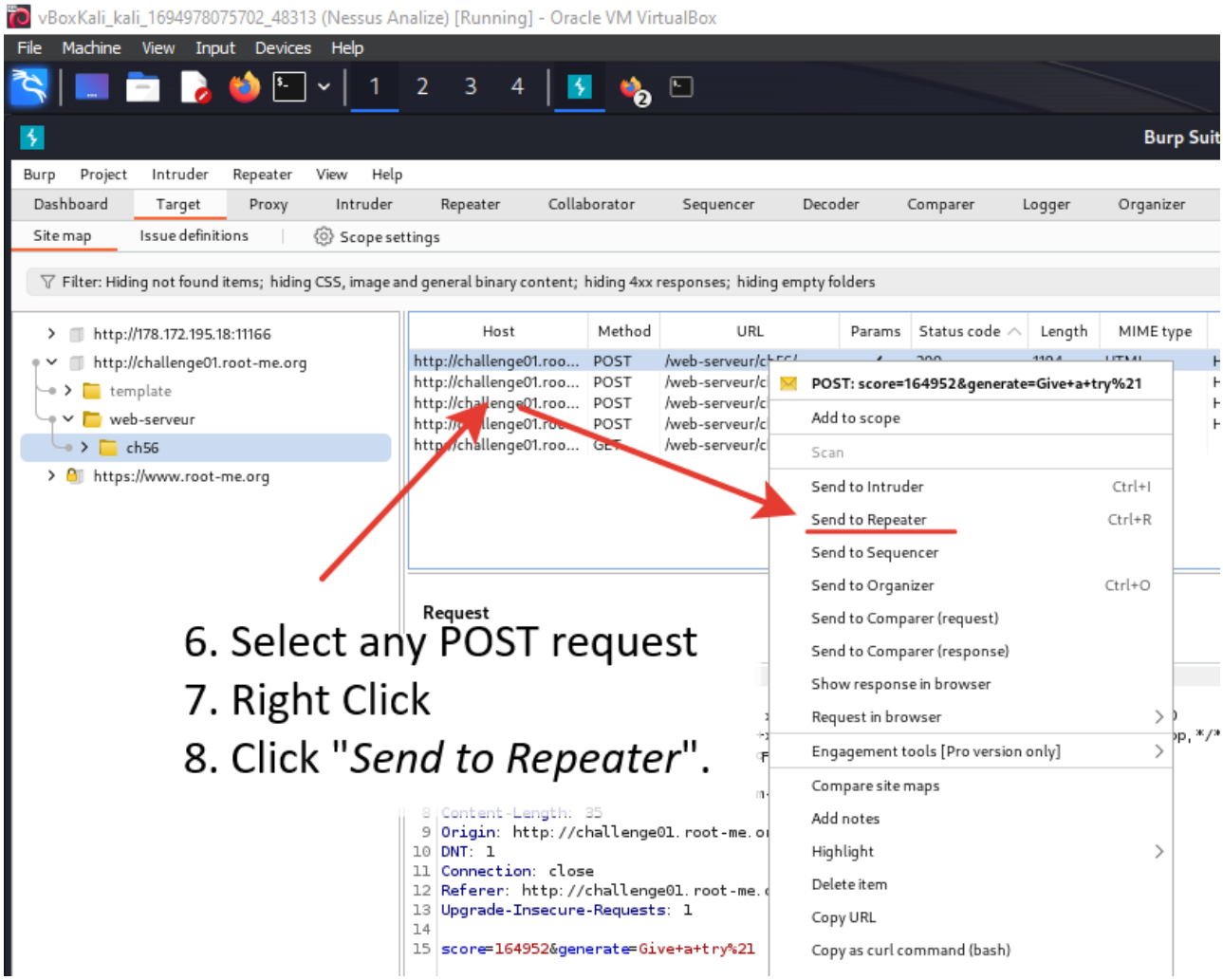
6. Find the challenge host (*e.g.: ch_*)



7. Select any POST request

8. Right click

9. Click "*Send to Repeater*"

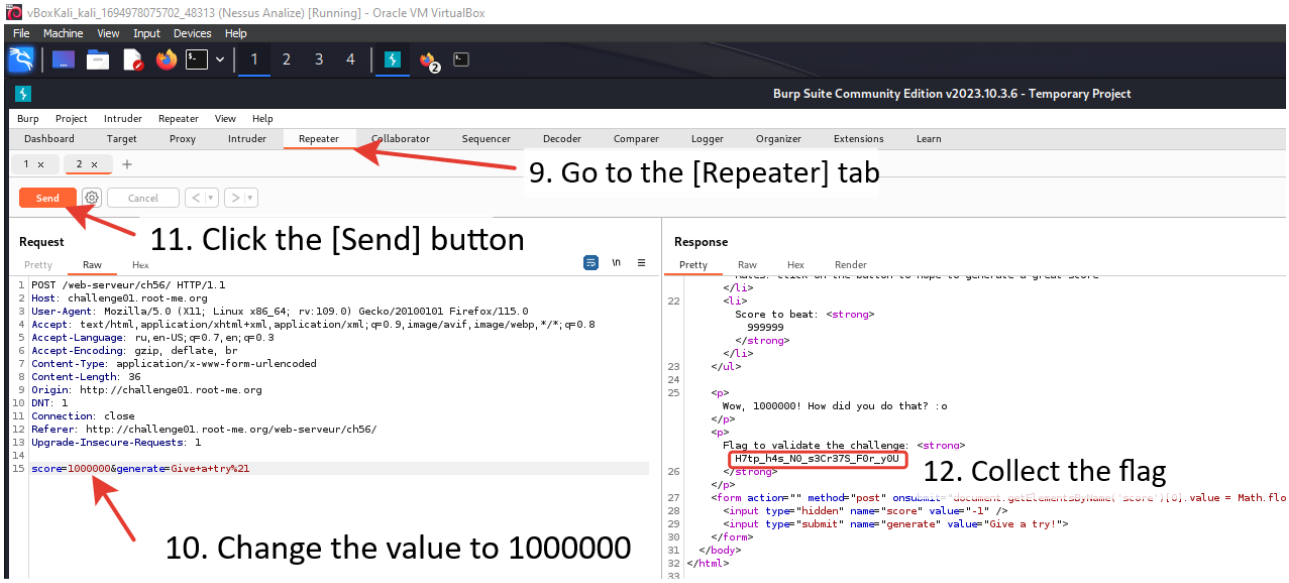


10. Go to the `[Repeater]` tab

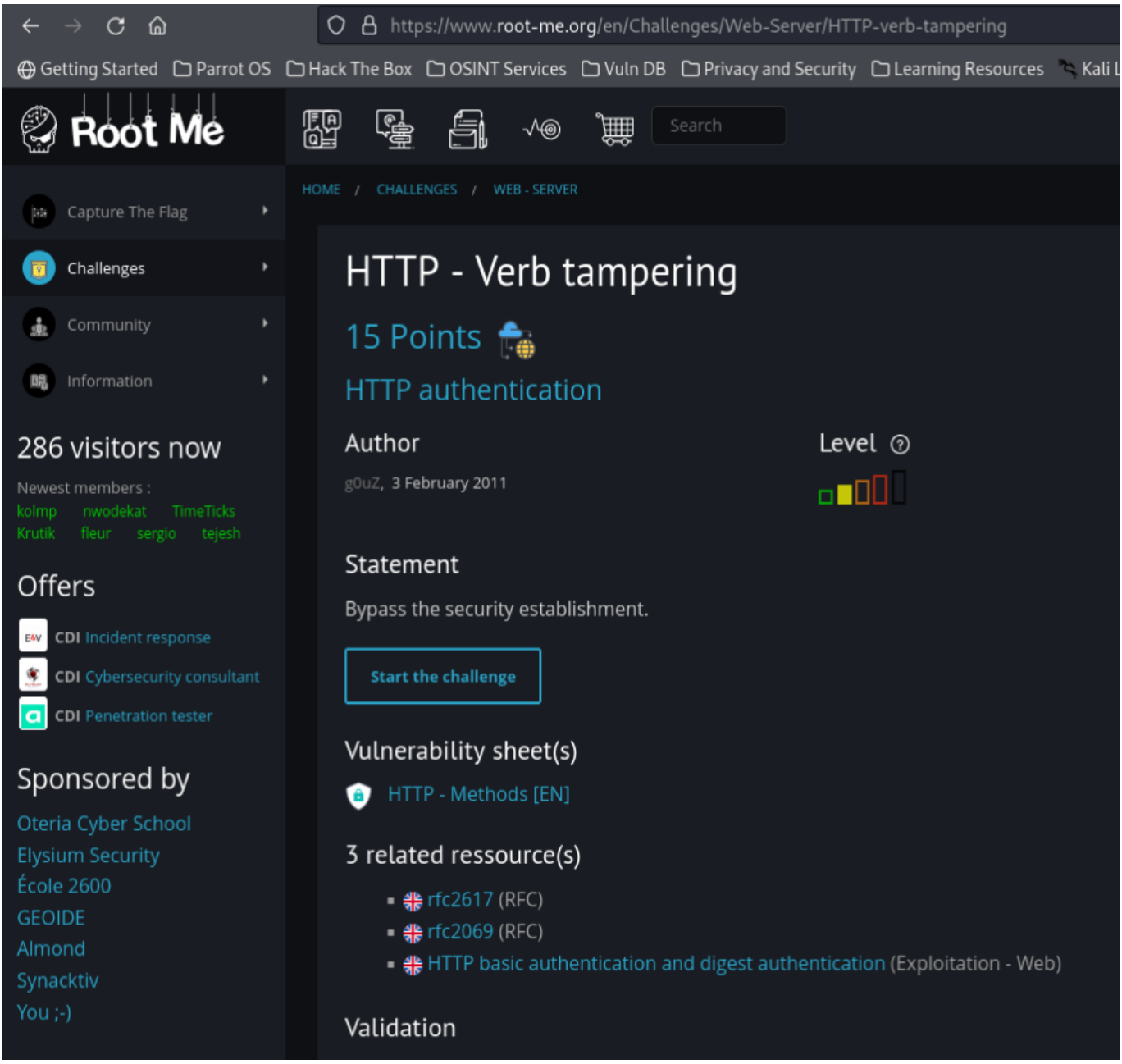
11. Change the value to *1000000*

12. Click the [Send] button

13. Collect the flag

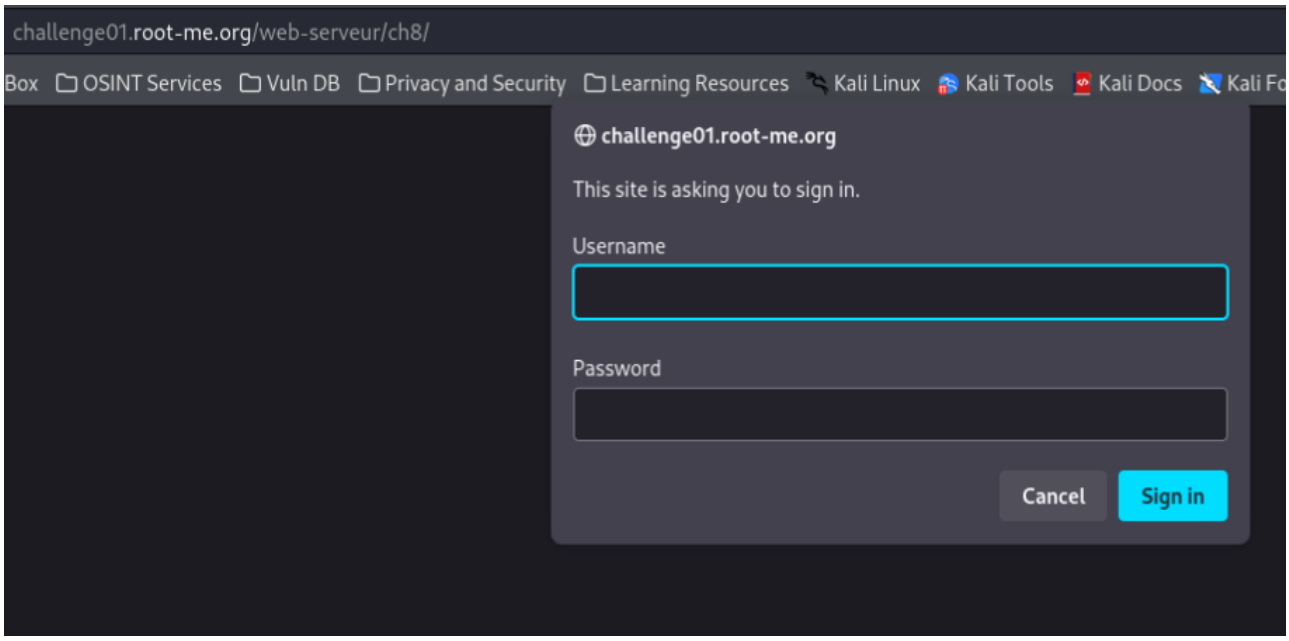


RootMe (HTTP - Verb tampering)



Solution

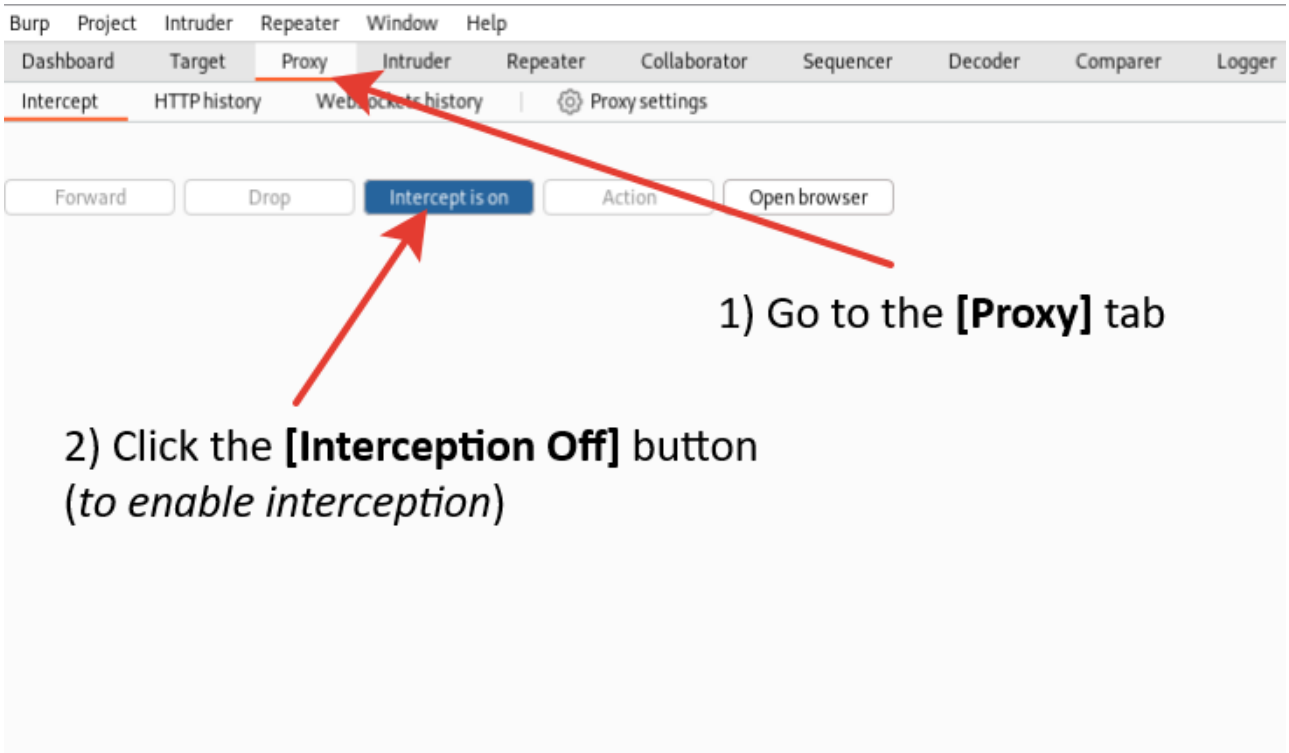
1. Run the challenge



2. Launch "*Burp Suite*".

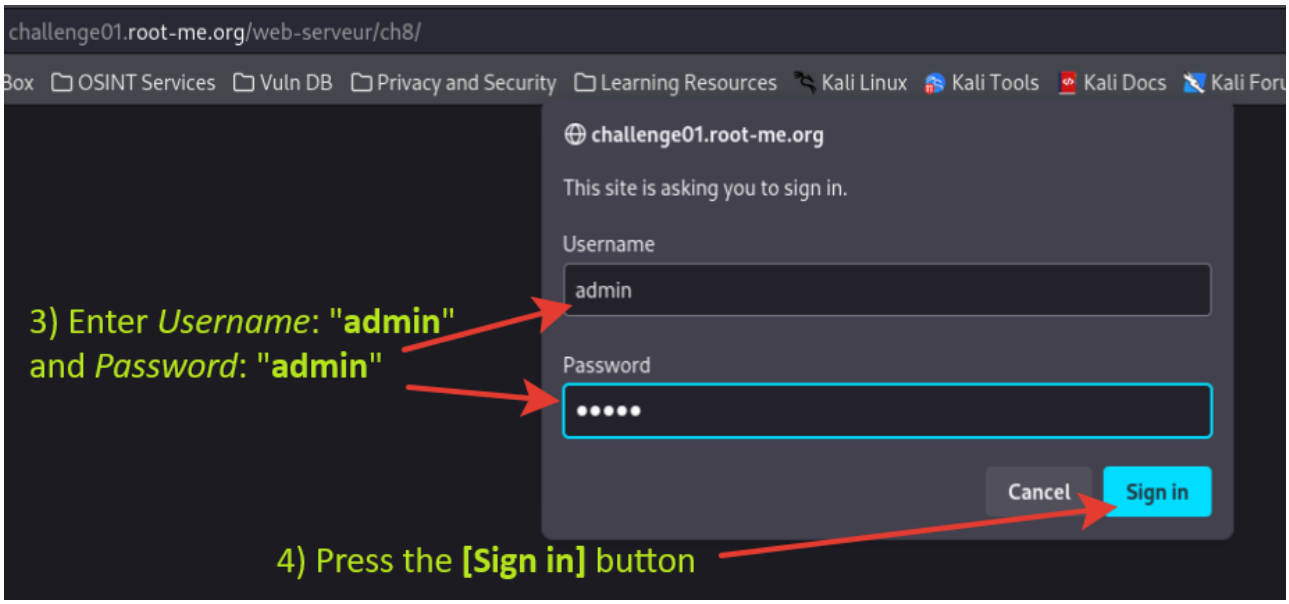
3. Go to the `[Proxy]` tab.

4. Click the `[Interception Off]` button (*to enable interception*).



5. Enter *Username:* `admin` and *Password:* `admin`.

6. Press the `[Sign in]` button.



7. After intercepting the request, the Burp Suite window will appear.

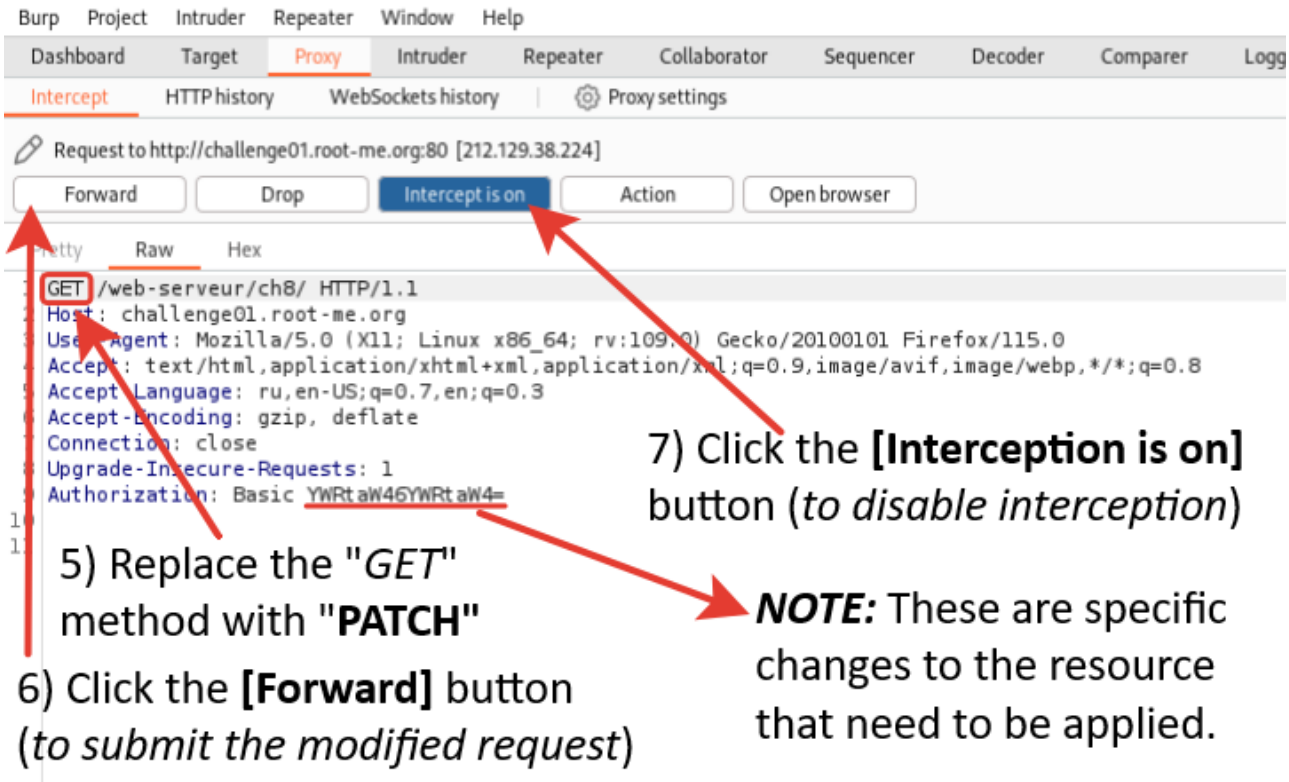
   The *GET method* is used to retrieve data from the server. This is a **read-only** method. But inside the intercepted request we can find the line `Authorization: Basic YWRtaW46YWRtaW4=` - these are specific changes to the resource. To apply the change, we need to use another method that can change the resource. These methods:

   - *POST* - creates a new resource,
   - *PUT* - updates an existing resource, but its body must contain the complete structure of the modified resource,
   - *PATCH* - updates an existing resource and contains in its body only specific changes for the resource.

   In this case, we **must replace** the `GET` method with `PATCH` to apply the authorization state change.

8. Click the `[Forward]` button (*to submit the modified request*).

9. Click the `[Interception is on]` button (*to disable interception*).



10. Return to the browser

11. Collect the "*Flag*"