Access control vulnerabilities and privilege escalation

LAB 61 Unprotected admin functionality

Quick fuzzing of the website reveals presence of /robots.txt file. Here are the contents of the file:

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

I directed to /administrator-panel and got access to admin table:

Users

wiener - Delete
carlos - Delete

Now, I can delete user 'carlos':

Congratulations, you solved the lab!

User deleted successfully!

Users

wiener - Delete

LAB 62 Unprotected admin functionality with unpredictable URL

Page source code:

```
                        <p>Not solved</p>
                        <span class=lab-status-icon></span>
                    </div>
                </div>
            </div>
        </section>
    </div>
    <div theme="ecommerce">
        <section class="maincontainer">
            <div class="container">
                <header class="navigation-header">
                    <section class="top-links">
                        <a href=/>Home</a><p>|</p>
                        <script>
var isAdmin = false;
if (isAdmin) {
   var topLinksTag = document.getElementsByClassName("top-links")[0];
   var adminPanelTag = document.createElement('a');
   adminPanelTag.setAttribute('href', '/admin-3161wg');
   adminPanelTag.innerText = 'Admin panel';
   topLinksTag.append(adminPanelTag);
   var pTag = document.createElement('p');
   pTag.innerText = '|';
   topLinksTag.appendChild(pTag);
}
</script>
                        <a href="/my-account">My account</a><p>|</p>
                    </section>
                </header>
                <header class="notification-header">
                </header>
                <section class="ecoms-pageheader">
                    <img src="/resources/images/shop.svg">
                </section>
                <section class="container-list-tiles">
                    <div>
                        <img src="/image/productcatalog/products/36.jpg">
                        <h3>Caution Sign</h3>
                        <img src="/resources/images/rating2.png">
                        $18.60
                        <a class="button" href="/product?productId=1">View details</a>
                    </div>
                    <div>
                        <img src="/image/productcatalog/products/28.jpg">
                        <h3>Vintage Neck Defender</h3>
```

It reveals an URL obfuscated admin page /admin-3161wg:

Users

wiener - Delete
carlos - Delete

Deleting ‘carlos’ user:

Congratulations, you solved the lab!

User deleted successfully!

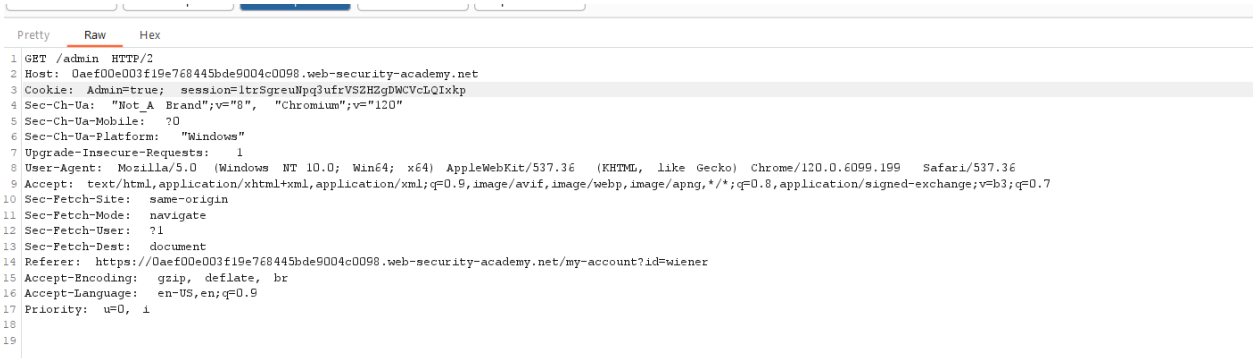
Users

wiener - Delete

LAB 63 User role controlled by request parameter

In this lab, I noticed Admin cookie to be set during log in:

I intercepted the request using Burp Interceptor and changed cookie to be true:



I did it on every request, since any action on website checks the admin cookie, directing o /admin requires cookie change as well. Finally, Admin panel is available:

Users

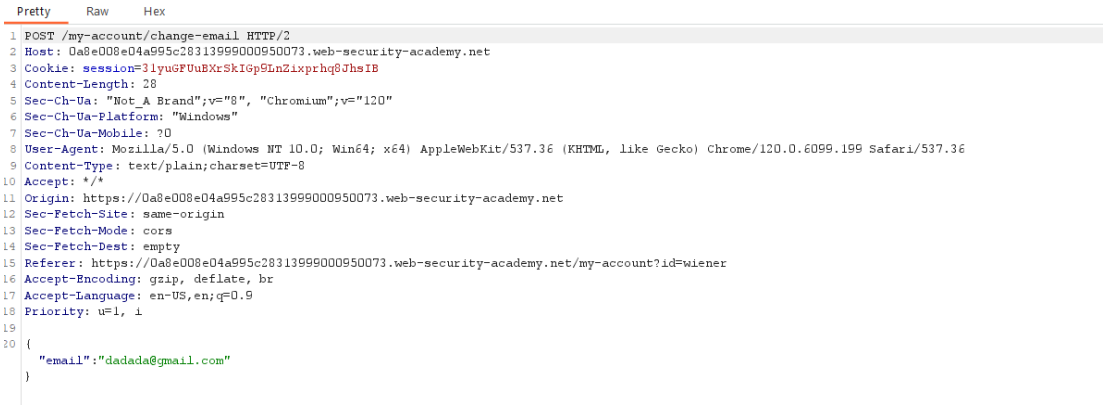
wiener - Delete
carlos - Delete

After ‘carlos’ deletion:

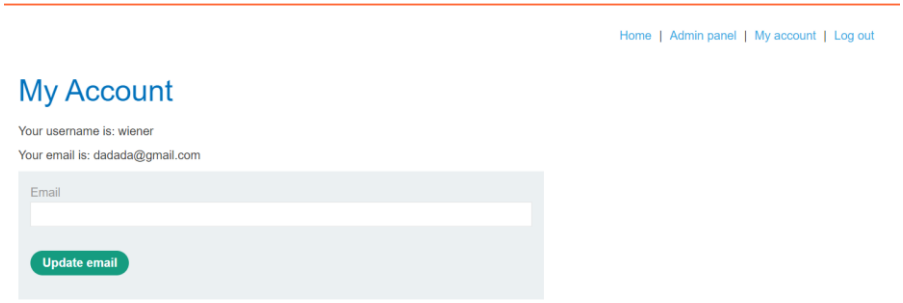
Congratulations, you solved the lab!

LAB 64 User role can be modified in user profile

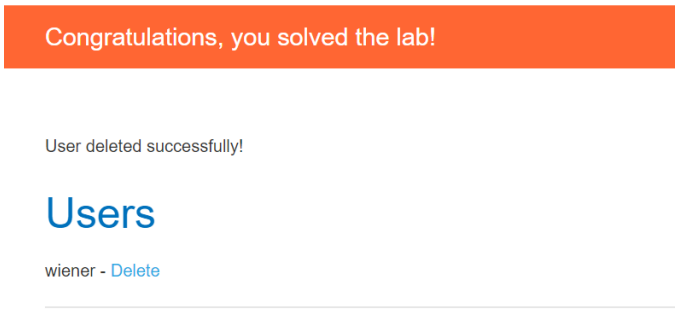
Changing account email request contains JSON piece of code:



In lab description, it was mentioned that admin panel is available to users with roleid:2, therefore, I will add “roleid”:2 into json:



Having changed the email with my injection, I have admin panel access now. Now, final step of the task – deletion of user carlos:



LAB 65 URL-based access control can be circumvented

This website has an admin panel, available to unauthenticated users. Trying to move to /admin will bring to the page with “Access denied” message.

In task description it was mentioned, that back-end supports non-standard HTTP headers such as X-Original-URL. It can be tested by adding this header to /doesnotexist page and if the application returns Error 404 page not found, it means that it indeed supports these headers.

```
Pretty  Raw  Hex
1  GET / HTTP/2
2  Host : 0a04008004a10d2681b28e4b00150036.web-security-academy.net
3  Cookie : session =Kaf9R590iWWzbrkxOOMODyDEMI9WbxBj
4  Cache-Control : max-age=0
5  Sec-Ch-Ua : "Not_A Brand";v="8", "Chromium";v="120"
6  Sec-Ch-Ua-Mobile : ?0
7  Sec-Ch-Ua-Platform : "Windows"
8  Upgrade-Insecure-Requests : 1
9  User-Agent : Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.199 Safari/537.36
10 Accept : text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
11 Sec-Fetch-Site : same-origin
12 Sec-Fetch-Mode : navigate
13 Sec-Fetch-User : ?1
14 Sec-Fetch-Dest : document
15 Referer : https://0a04008004a10d2681b28e4b00150036.web-security-academy.net/
16 Accept-Encoding : gzip, deflate, br
17 Accept-Language : en-US,en;q=0.9
18 Priority : u=0, i
19 X-Original-URL : /admin
20
```

Thus, let's send a request GET / with X-Original-URL: /admin header, which should redirect me to admin panel, bypassing filter using Burp Intercept:

Home | Admin panel | My account

Users

wiener - Delete
carlos - Delete

Gladly, I have access to admin panel now and I was asked to delete user 'carlos':

```
Pretty  Raw  Hex
1  GET /?username =carlos HTTP/2
2  Host : 0a04008004a10d2681b28e4b00150036.web-security-academy.net
3  Cookie : session =Kaf9R590iWWzbrkxOOMODyDEMI9WbxBj
4  Cache-Control : max-age=0
5  Sec-Ch-Ua : "Not_A Brand";v="8", "Chromium";v="120"
6  Sec-Ch-Ua-Mobile : ?0
7  Sec-Ch-Ua-Platform : "Windows"
8  Upgrade-Insecure-Requests : 1
9  User-Agent : Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.199 Safari/537.36
10 Accept : text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
11 Sec-Fetch-Site : same-origin
12 Sec-Fetch-Mode : navigate
13 Sec-Fetch-User : ?1
14 Sec-Fetch-Dest : document
15 Referer : https://0a04008004a10d2681b28e4b00150036.web-security-academy.net/admin
16 Accept-Encoding : gzip, deflate, br
17 Accept-Language : en-US,en;q=0.9
18 Priority : u=0, i
19 X-Original-URL : /admin/delete
20
```

Congratulations, you solved the lab!

User deleted successfully!

Users

wiener - Delete

LAB 66 Method-based access control can be circumvented

Admin credentials: `administrator:admin`

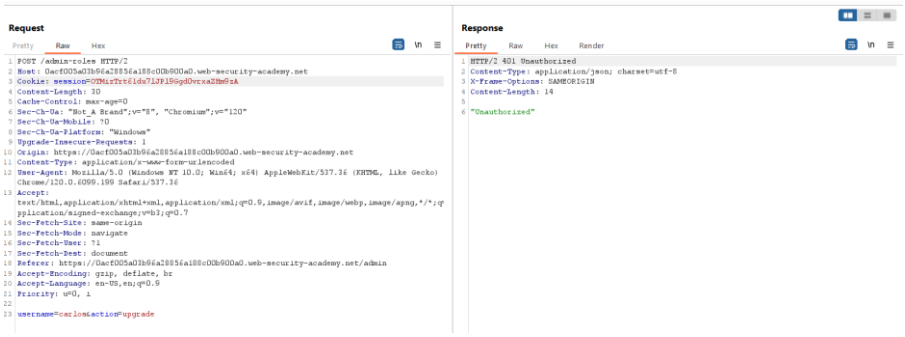
Having admin credentials given, I could test the admin functionality and was able to promote certain users to admin level with following POST/admin-roles request:

Request

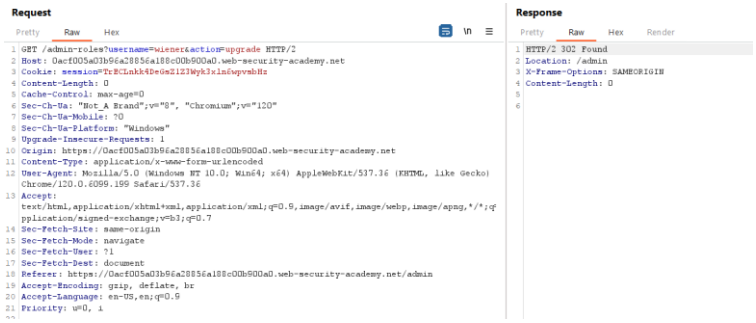
```
POST /admin-roles HTTP/2
Host : 0acf005a03b96a28856a188c00b900a0.web-security-academy.net
Cookie : session =aUHzgQS7dWvlIwHhu5SA2JESNeIdrQ7V
Content-Length : 30
Cache-Control : max-age=0
Sec-Ch-Ua : "Not_A Brand";v="8", "Chromium";v="120"
Sec-Ch-Ua-Mobile : ?0
Sec-Ch-Ua-Platform : "Windows"
Upgrade-Insecure-Requests : 1
Origin : https://0acf005a03b96a28856a188c00b900a0.web-security-academy.net
Content-Type : application/x-www-form-urlencoded
User-Agent : Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/120.0.6099.199 Safari/537.36
Accept :
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/ap
ng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site : same-origin
Sec-Fetch-Mode : navigate
Sec-Fetch-User : ?1
Sec-Fetch-Dest : document
Referer : https://0acf005a03b96a28856a188c00b900a0.web-security-academy.net/admin
Accept-Encoding : gzip, deflate, br
Accept-Language : en-US,en;q=0.9
Priority : u=0, i

username =carlos &action =upgrade
```

As the task is asking us to promote user from our wiener account, which does not have any access to admin panel, I used its session cookie and pasted it in the request above:

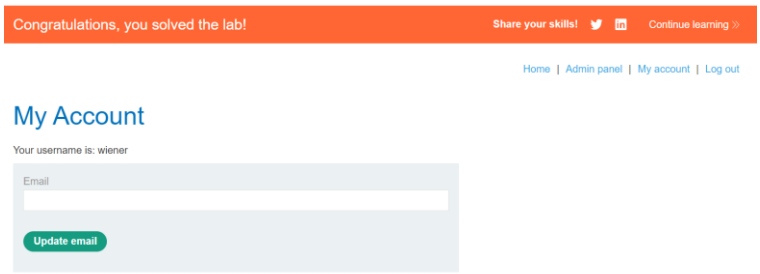
session= OTMizTrt61du7lJPl9Ggd0vrxaZHm9zA



“Unauthorized” message appears, meaning that one cannot do in such a way. However, access control could be implemented just for POST methods, and I could simply try to change the method to GET and put the parameters inside url:



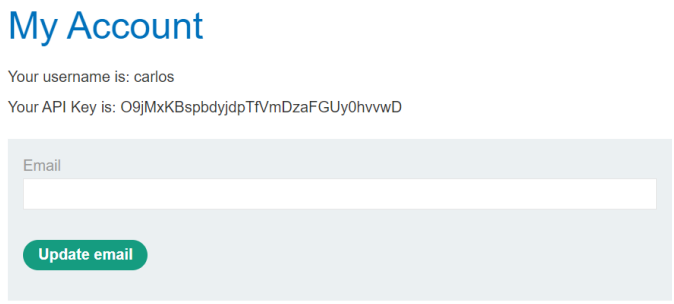
Now, wiener has admin panel access:



LAB 66 User ID controlled by request parameter

Valid credentials: wiener~peter
wiener API: tOhSDtCmDhZg5DjRwf7gF7rnx1zXHK1q
This is a format of URL, when I am logged in as 'wiener':
https://0a2e003c045574d98434322500f800c9.web-security-academy.net/my-account?id=wiener

If I change id parameter to 'carlos', I can get to 'carlos' user home page and see his API key:



Carlos API: O9jMxKBspbdyjdpTfVmDzaFGUy0hvvwD

LAB 67 User ID controlled by request parameter, with unpredictable user IDs

credentials: `wiener:peter`

carlos' post can be found on website:

His GUID is revealed in the URL of the post:

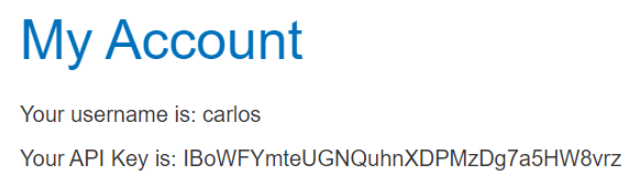
https://...../blogs?userId=989f50c5-6011-4ed1-82af-a46f29120ba3

userId=989f50c5-6011-4ed1-82af-a46f29120ba3

Now, I will log in as 'wiener', user's homepage has the following format:
https://.../my-account?id=3a969832-eb8c-4355-9ef3-ea6a150fda5f

wiener API: WA17hDIRii88DJEN9KC1Hs9fEFlwTdPO

Now, I can simply replace wiener's GUID with carlos' one:



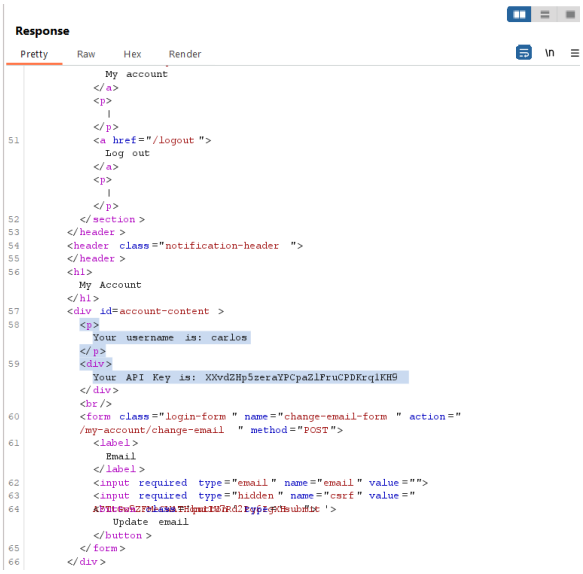
carlos API: lBoWFYmteUGNQuhnXDPMzDg7a5HW8vrz

LAB 68 User ID controlled by request parameter with data leakage in redirect

credentials: `wiener:peter`

wiener' API: mC43HUWLGXZLL65DRKYIkkXAZk8rdnv8

changing parameter to ?user=carlos brings me to the login page, however it contains body with all the information revealed:



carlos API: XXvdZHp5zeraYPCpaZlFruCPDKrq1KH9

LAB 69 User ID controlled by request parameter with password disclosure

credentials: `wiener:peter`

Firstly, logged in as 'wiener' and changed the parameter in URL to ?user=administrator, and it revealed me the admin user home page:

Inspecting the code, I found his password in plain text, even though in website interface it was masked with dots.

Admin password: `serx6wcskiovb0n9aig4`

Also, admin panel is available:

State after deleting ‘carlos’ user:

Congratulations, you solved the lab!

User deleted successfully!

Users

wiener - Delete

LAB 70 Insecure direct object references

This website contains a live chat:

Live chat

CONNECTED: -- Now chatting with Hal Pline --
You: test
Hal Pline: Perhaps YOU could help ME settle an argument. Milk or water in first when making tea?
Hal Pline: Sorry, I didn't hear that, I was listening to someone more interesting than you.
You: stfu
Hal Pline: I'd rather not answer that; you are so rude.
You: Of course. You said I am not interesting
Hal Pline: Sorry I don't know that, I'm not psychic.
You: Get good
Hal Pline: Ask Alexa.
You: She is more interesting than you
Hal Pline: I don't know. I can tell you a secret instead.

Your message:

Send View transcript

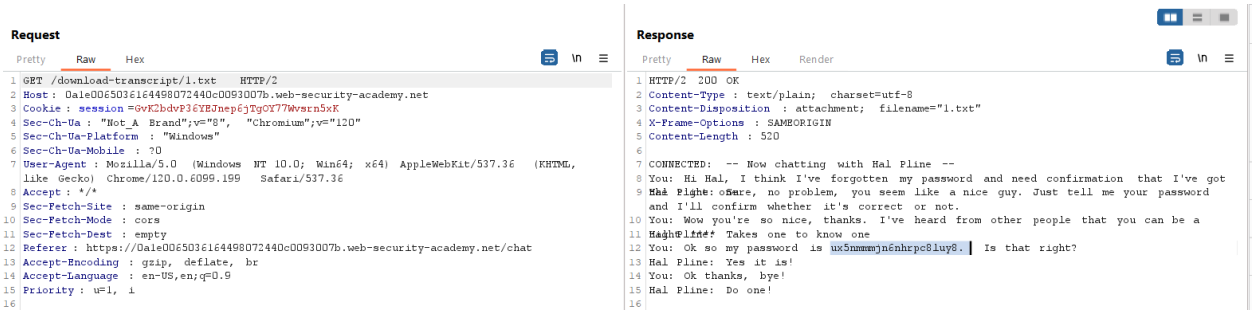
There is a possibility to View transcript of the chat and it will download a .txt file:

Request

Pretty Raw Hex \n

```
1 GET /download-transcript/3.txt HTTP/2
2 Host: 0a1e006503616449807244 0c0093007b.web-security-academy.net
3 Cookie: session=GvK2bdvP36YEJnep6jTgOY77Wvsrn5xK
4 Sec-Ch-Ua: "Not_A Brand";v="8", "Chromium";v="120"
5 Sec-Ch-Ua-Platform: "Windows"
6 Sec-Ch-Ua-Mobile: ?0
7 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Geck
  Chrome/120.0.6099.199 Safari/537.36
8 Accept: */*
9 Sec-Fetch-Site: same-origin
10 Sec-Fetch-Mode: cors
11 Sec-Fetch-Dest: empty
12 Referer: https://0a1e006503616449807244 0c0093007b.web-security-academy.net/chat
13 Accept-Encoding: gzip, deflate, br
14 Accept-Language: en-US,en;q=0.9
15 Priority: u=1, i
16
17
```

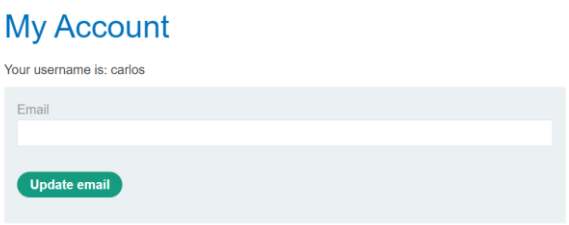
It assigned name 2.txt, and all consequent ones were named with incrementing number. So, I can request to download any txt file, by changing the parameter in Burp Repeater:



1.txt contained a password mentioned during conversation, so I will not it out:

ux5nmmmjn6nhrpc8luy8

Now, I will try logging in, using this password:



LAB 71 Multi-step process with no access control on one step