



rawSEC rENTAS CTF 2024 Preliminary Round

Write-Up

Team Name: We are High!

Chen Lik Ken, Karen Ng Jia Hui & Ong Poh Aik

Asia Pacific University of Technology & Innovation (APU)

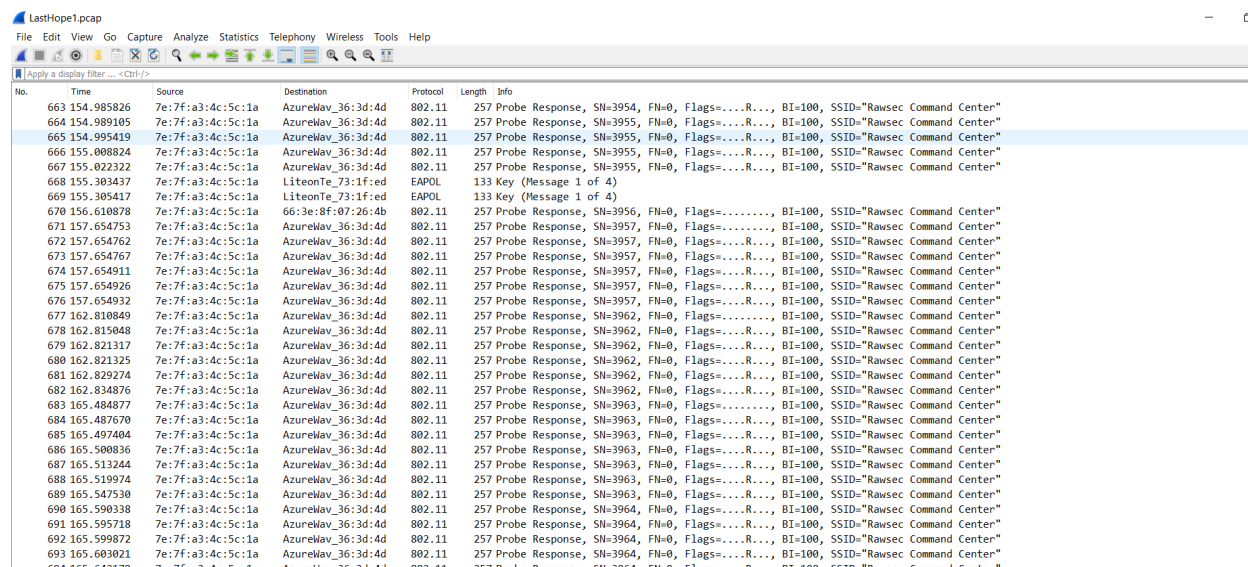
Category: Network

Challenge Name: Last Hope

A .pcap file named Last Hope was given in the challenge. .pcap files are a type of file type that usually contains data of packets captured within a network.

It was clear that we are required to get the password of the wifi, which would be submitted as a flag. To do so, we decided to use aircrack-ng to crack the password of the wifi.

For aircrack-ng to work, we filtered out all entries that contains the EAPOL protocol (Authentication protocol) as well as entries that contain the SSID (Rawsec Command Center) and exported these entries as a separate file that looks like this:



No.	Time	Source	Destination	Protocol	Length	Info
663	154.985826	7e:7f:a3:4c:5c:1a	AzureNav_36:3d:4d	802.11	257	Probe Response, SN=3954, FN=0, Flags=...R..., BI=100, SSID="Rawsec Command Center"
664	154.989185	7e:7f:a3:4c:5c:1a	AzureNav_36:3d:4d	802.11	257	Probe Response, SN=3955, FN=0, Flags=...R..., BI=100, SSID="Rawsec Command Center"
665	154.995419	7e:7f:a3:4c:5c:1a	AzureNav_36:3d:4d	802.11	257	Probe Response, SN=3955, FN=0, Flags=...R..., BI=100, SSID="Rawsec Command Center"
666	155.008824	7e:7f:a3:4c:5c:1a	AzureNav_36:3d:4d	802.11	257	Probe Response, SN=3955, FN=0, Flags=...R..., BI=100, SSID="Rawsec Command Center"
667	155.022322	7e:7f:a3:4c:5c:1a	AzureNav_36:3d:4d	802.11	257	Probe Response, SN=3955, FN=0, Flags=...R..., BI=100, SSID="Rawsec Command Center"
668	155.303437	7e:7f:a3:4c:5c:1a	LiteonTe_73:1f:ed	EAPOL	133	Key (Message 1 of 4)
669	155.305417	7e:7f:a3:4c:5c:1a	LiteonTe_73:1f:ed	EAPOL	133	Key (Message 1 of 4)
670	156.610878	7e:7f:a3:4c:5c:1a	66:3e:8f:07:26:4b	802.11	257	Probe Response, SN=3956, FN=0, Flags=...R..., BI=100, SSID="Rawsec Command Center"
671	157.654753	7e:7f:a3:4c:5c:1a	AzureNav_36:3d:4d	802.11	257	Probe Response, SN=3957, FN=0, Flags=...R..., BI=100, SSID="Rawsec Command Center"
672	157.654762	7e:7f:a3:4c:5c:1a	AzureNav_36:3d:4d	802.11	257	Probe Response, SN=3957, FN=0, Flags=...R..., BI=100, SSID="Rawsec Command Center"
673	157.654767	7e:7f:a3:4c:5c:1a	AzureNav_36:3d:4d	802.11	257	Probe Response, SN=3957, FN=0, Flags=...R..., BI=100, SSID="Rawsec Command Center"
674	157.654911	7e:7f:a3:4c:5c:1a	AzureNav_36:3d:4d	802.11	257	Probe Response, SN=3957, FN=0, Flags=...R..., BI=100, SSID="Rawsec Command Center"
675	157.654926	7e:7f:a3:4c:5c:1a	AzureNav_36:3d:4d	802.11	257	Probe Response, SN=3957, FN=0, Flags=...R..., BI=100, SSID="Rawsec Command Center"
676	157.654932	7e:7f:a3:4c:5c:1a	AzureNav_36:3d:4d	802.11	257	Probe Response, SN=3957, FN=0, Flags=...R..., BI=100, SSID="Rawsec Command Center"
677	162.810849	7e:7f:a3:4c:5c:1a	AzureNav_36:3d:4d	802.11	257	Probe Response, SN=3962, FN=0, Flags=...R..., BI=100, SSID="Rawsec Command Center"
678	162.815048	7e:7f:a3:4c:5c:1a	AzureNav_36:3d:4d	802.11	257	Probe Response, SN=3962, FN=0, Flags=...R..., BI=100, SSID="Rawsec Command Center"
679	162.821317	7e:7f:a3:4c:5c:1a	AzureNav_36:3d:4d	802.11	257	Probe Response, SN=3962, FN=0, Flags=...R..., BI=100, SSID="Rawsec Command Center"
680	162.821325	7e:7f:a3:4c:5c:1a	AzureNav_36:3d:4d	802.11	257	Probe Response, SN=3962, FN=0, Flags=...R..., BI=100, SSID="Rawsec Command Center"
681	162.829274	7e:7f:a3:4c:5c:1a	AzureNav_36:3d:4d	802.11	257	Probe Response, SN=3962, FN=0, Flags=...R..., BI=100, SSID="Rawsec Command Center"
682	162.834876	7e:7f:a3:4c:5c:1a	AzureNav_36:3d:4d	802.11	257	Probe Response, SN=3962, FN=0, Flags=...R..., BI=100, SSID="Rawsec Command Center"
683	165.484877	7e:7f:a3:4c:5c:1a	AzureNav_36:3d:4d	802.11	257	Probe Response, SN=3963, FN=0, Flags=...R..., BI=100, SSID="Rawsec Command Center"
684	165.487670	7e:7f:a3:4c:5c:1a	AzureNav_36:3d:4d	802.11	257	Probe Response, SN=3963, FN=0, Flags=...R..., BI=100, SSID="Rawsec Command Center"
685	165.497404	7e:7f:a3:4c:5c:1a	AzureNav_36:3d:4d	802.11	257	Probe Response, SN=3963, FN=0, Flags=...R..., BI=100, SSID="Rawsec Command Center"
686	165.500836	7e:7f:a3:4c:5c:1a	AzureNav_36:3d:4d	802.11	257	Probe Response, SN=3963, FN=0, Flags=...R..., BI=100, SSID="Rawsec Command Center"
687	165.513244	7e:7f:a3:4c:5c:1a	AzureNav_36:3d:4d	802.11	257	Probe Response, SN=3963, FN=0, Flags=...R..., BI=100, SSID="Rawsec Command Center"
688	165.519974	7e:7f:a3:4c:5c:1a	AzureNav_36:3d:4d	802.11	257	Probe Response, SN=3963, FN=0, Flags=...R..., BI=100, SSID="Rawsec Command Center"
689	165.547530	7e:7f:a3:4c:5c:1a	AzureNav_36:3d:4d	802.11	257	Probe Response, SN=3963, FN=0, Flags=...R..., BI=100, SSID="Rawsec Command Center"
690	165.598338	7e:7f:a3:4c:5c:1a	AzureNav_36:3d:4d	802.11	257	Probe Response, SN=3964, FN=0, Flags=...R..., BI=100, SSID="Rawsec Command Center"
691	165.598718	7e:7f:a3:4c:5c:1a	AzureNav_36:3d:4d	802.11	257	Probe Response, SN=3964, FN=0, Flags=...R..., BI=100, SSID="Rawsec Command Center"
692	165.599872	7e:7f:a3:4c:5c:1a	AzureNav_36:3d:4d	802.11	257	Probe Response, SN=3964, FN=0, Flags=...R..., BI=100, SSID="Rawsec Command Center"
693	165.603021	7e:7f:a3:4c:5c:1a	AzureNav_36:3d:4d	802.11	257	Probe Response, SN=3964, FN=0, Flags=...R..., BI=100, SSID="Rawsec Command Center"
694	165.649170	7e:7f:a3:4c:5c:1a	AzureNav_36:3d:4d	802.11	257	Probe Response, SN=3964, FN=0, Flags=...R..., BI=100, SSID="Rawsec Command Center"

Later, we copied this file into the Desktop of Kali. The terminal was opened, and the command shown below was run.

```
(kali@TP064812)-[~]  
$ cd Desktop  
  
(kali@TP064812)-[~/Desktop]  
$ aircrack-ng LastHope1.pcap -w rockyou.txt
```

Just to make the process easier, a copy of rockyou.txt, which contained all common passwords, was moved to Desktop. The command shown above will attempt to crack the password of the Wifi by using aircrack-ng in which it would analyze the network data in LastHope1.pcap and brute force its way through by testing all common passwords inside the rockyou.txt to find a match.

```
Aircrack-ng 1.7

[00:00:02] 6977/14344392 keys tested (3046.66 k/s)

Time left: 1 hour, 18 minutes, 25 seconds                                0.05%

KEY FOUND! [ anonymous ]

Master Key      : 94 7D 53 8E F7 F3 22 52 BC 89 D4 B7 DB BE 77 E3
                  A7 A8 D2 89 9A 1B 58 43 84 E3 4A 52 D5 90 BB F5

Transient Key   : 8E 41 35 02 02 91 DD EA AE 6F 04 1C 93 7E 66 D7
                  DB 2C 1E 13 D7 54 9E 77 83 D3 F2 1E 08 62 9B 59
                  53 12 38 DA 5E E0 50 BF 70 52 31 67 F9 69 91 DD
                  FF 54 08 E1 59 37 92 F9 12 5E D6 1B 3F FE 43 AC

EAPOL HMAC     : 59 CD 37 EF 5A E7 87 0E 76 54 AE E6 44 CB 90 7E

(kali@TP064812)-[~/Desktop]
$ █
```


Flag found!

RWSC {anonymous}

Category: DFIR

Challenge Name: Mobile

A 2700+ page pdf named report was given! Through relentlessly scrolling, we discovered a page where the contents seemed different from the others, where other tables usually have each row filled with their respective. On the other hand, the suspicious looking table shown below has MD5 hash and SHA1 hash changed. Raw data and the source were also different.

	
Size (Bytes)	14427
Skin Tone Percentage	0.0
Original Width	512
Original Height	512
Exif Extraction Status	Complete
Exif Data	Extraction Result: Complete ImageWidth: 512 ImageHeight: 512
MD5 Hash	
SHA1 Hash	
_rawData	8e7e00c0bd5ce227f7be204c8b7c159669c776d4
Source	<ul style="list-style-type: none">• /data/system/
Location	<ul style="list-style-type: none">• File Offset 169832
Evidence number	<ul style="list-style-type: none">• Lenovo Lenovo P70 Full Image
Recovery method	<ul style="list-style-type: none">• Carving
Item ID	3999

Based on the hint provided, which is a youtube video of a hacker helping a client unlock a swipe pattern locked phone, we were able to gain information such as:





- SHA1 was used in the protection of the swipe pattern lock of android, and was usually stored in gesture.key under /data/system
- A rainbow table exists where all possible pattern combinations and their respective SHA1 hash are listed.

At this point, the objective was to gain access to the rainbow table and that the raw data discovered in the pdf file was most likely a SHA1 hash for a swipe pattern lock on android (/data/system).

Through some searching online, we managed to discover a github repository that runs a program to crack the Android swipe pattern lock.

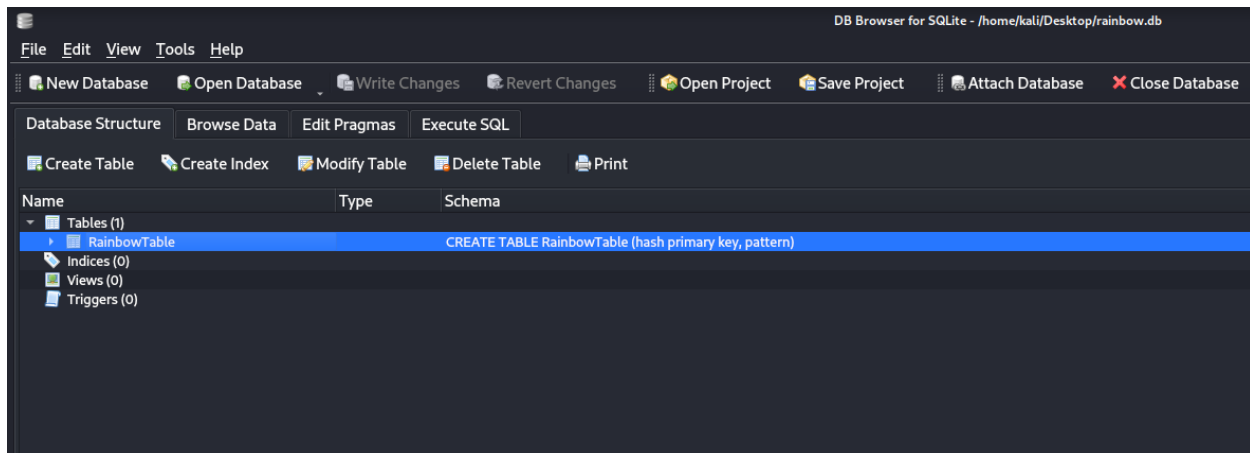
<https://github.com/Webblitchy/AndroidGestureCrack>

In this repository, a rainbow table that exists as a db file is present.

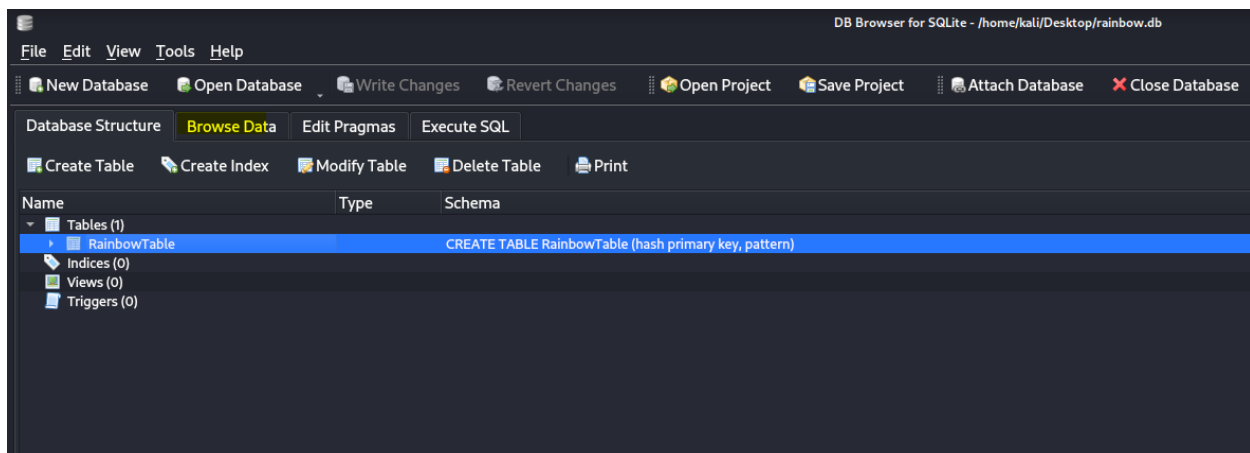
 Webblitchy	Update README	3 years ago
 README	Update README	3 years ago
 gesturecrack.py	Updated the script to run with python3	3 years ago
 rainbow.db.gz	Added rainbow table, no gunzip needed ju...	10 years ago

Out of great interest to get a copy of the rainbow table, the github repository was not cloned into Kali. Instead, we downloaded the rainbow.db.gz file and changed the file extension to .db only.

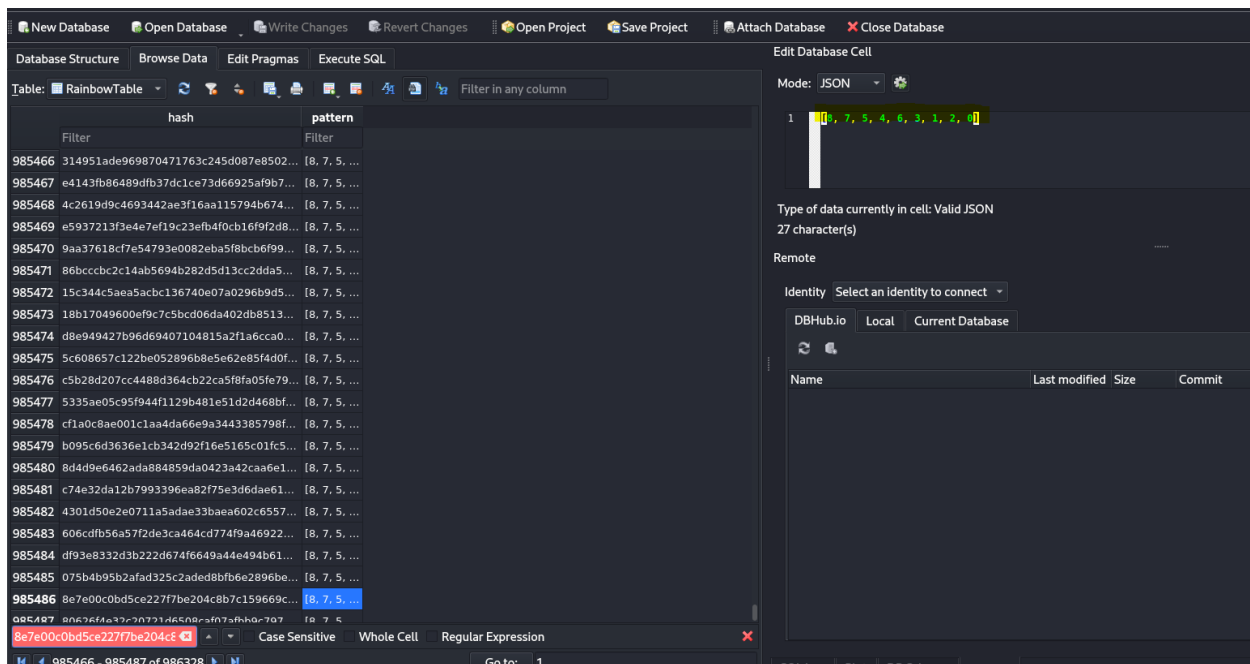
Now, with the rainbow table ready, we copied the file into Kali and opened it with SQLite.



Click on Browse data to view the contents



We then copy and pasted the suspected SHA1 entry found from the report into the search column, which resulted in us finding a matched entry as well as the pattern of the swipe lock.



Flag found!

RWSC {875463120}

Category: Web

Challenge Name: Inspirational Quotes

In this challenge, we are presented with a website with a link called “root.”

Inspirational Quotes

The future belongs to those who believe in the beauty of their dreams. - Eleanor Roosevelt

Directories

[root](#)

The link led us to root directory with a bunch of emojis with links to other directories.

Inspirational Quotes

Don't watch the clock; do what it does. Keep going. - Sam Levenson

Directories



Which led us to more emojis with links to other directories.

Inspirational Quotes

Success is stumbling from failure to failure with no loss of enthusiasm. - Winston S. Churchill

Directories



After trying random links to go to the last directory, no flag is present.

Inspirational Quotes

It always seems impossible until it's done. - Nelson Mandela

Directories

No directories found.

After trying out more links, we found out that the links within each directory are unique from other directories, therefore there will be no loop. We also found that each directory starting from root to the last directory is 5 directories deep. We guessed that we just need to brute force all directories to find the flag at the last directory among all directories. Since Gobuster is useless in this scenario, we asked ChatGPT to write us a script to crawl all links to all directories.

```

import requests
from bs4 import BeautifulSoup
from urllib.parse import urljoin

def extract_links(url):
    response = requests.get(url)
    soup = BeautifulSoup(response.text, 'html.parser')
    links = [a.get('href') for a in soup.find_all('a', href=True)]
    return [urljoin(url, link) for link in links]

def extract_links_recursive(start_url, depth):
    visited_urls = set()

    def recursive_helper(url, current_depth):
        if current_depth > depth or url in visited_urls:
            return []

        visited_urls.add(url)
        links = extract_links(url)
        nested_links = [recursive_helper(link, current_depth + 1) for link in links]

        return [url] + [item for sublist in nested_links for item in sublist]

    return recursive_helper(start_url, 0)

if __name__ == "__main__":
    start_url = 'https://byos.ctf.rawsec.com/root/' # Replace with the starting URL
    extraction_depth = 6 # Replace with the desired depth

    all_links = extract_links_recursive(start_url, extraction_depth)

    for link in all_links:
        print(link)

```

Then, we run the script and store the output in a text file.

```

(kali㉿kali)-[~/Downloads]
$ python p.py > output.txt

```

Then, we filter the text file for the last directories only to a second text file.

```

(kali㉿kali)-[~/Downloads]
$ grep -E '^([^\/*]*[/^\/*]*){9}$' output.txt > output2.txt

```

Since the last page of all directories shows “No directories found”, we just need to find a page with a different message. We wrote another script to check if the last page of each directory contains the message within `<p></p>`.


```

import requests
from bs4 import BeautifulSoup

def check_word_in_paragraph(url, keyword):
    try:
        response = requests.get(url)
        if response.status_code == 200:
            soup = BeautifulSoup(response.text, 'html.parser')
            paragraphs = soup.find_all('p')

            for paragraph in paragraphs:
                if keyword.lower() in paragraph.get_text().lower():
                    return True

    except Exception as e:
        print(f"Error checking {url}: {e}")

    return False

if __name__ == "__main__":
    # Read the list of URLs from a text file
    with open('output2.txt', 'r') as file:
        websites = file.read().splitlines()

    keyword_to_check = 'No directories found.'

    for website in websites:
        contains_word = check_word_in_paragraph(website, keyword_to_check)
        print(f"{website} contains keyword in paragraph: {contains_word}")

```

We run the script and store the output in a third text file.

```

(kali㉿kali)-[~/Downloads]
$ python s.py > output3.txt

```

Then, we filter the file for “False”, which means it does not have the “No directories found” text.

```

(kali㉿kali)-[~/Downloads]
$ grep "False" output3.txt
https://byos.ctf.rawsec.com/root/%F0%9F%A4%A4%F0%9F%A4%95%F0%9F%98%83/%F0%9F%98%94%F0%9F%98%81%F0%9F%98%95%F0%9F%98%B5/%F0%9F%98%BA%F0%9F%98%AA%F0%9F%A5%B4%F0%9F%98%87/%F0%9F%A5%B0%F0%9F%A5%B6%F0%9F%A4%A3%F0%9F%98%82/%F0%9F%A4%A7%F0%9F%98%85/index.php contains key word in paragraph: False

```

We went to the link and found the flag.

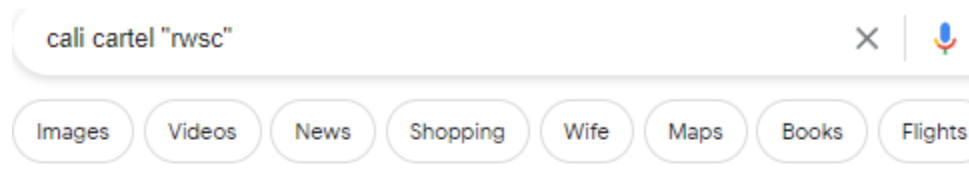


RWSC{J4CKP0T}

Category: OSINT

Cali Cartel

This question is easy. The easiest way to search for flag on google? GOOGLE DORK! Just type in what are we looking for with the keyword in quotes.



About 6,420 results (0.29 seconds)



Reddit · r/narcos

30+ comments · 3 years ago

[Jorge Salcedo, the man who snitched on Cali. : r/narcos](#)

I feel like we should not be distributing pictures of a guy who is in hiding from a murderous cartel. ... RWSC{C4L1_C4RT3L_PWN3D}. Upvote 1

Just to check if the flag is valid, it was posted recently so we tried it, and we got the flag.



Few-Act876 · 4d ago

RWSC{C4L1_C4RT3L_PWN3D}



1



Reply



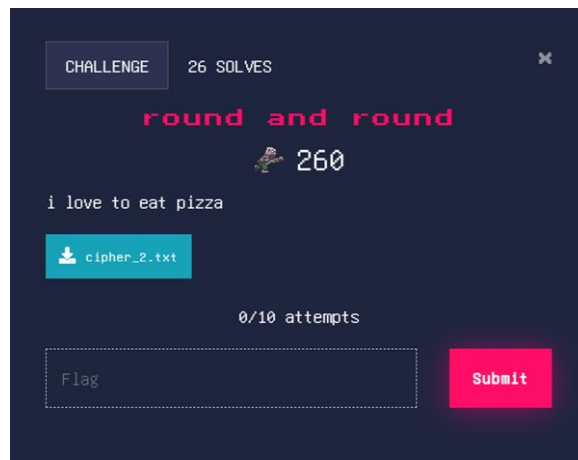
Share



RWSC{C4L1_C4RT3L_PWN3D}

Category: Cryptography

Challenge Name: round and round



After downloading and viewing the ciphertext, we guessed that it was some sort of substitution cipher because the arrangement of the ciphertext is similar to RWSC{flag}.

```
2126226{19122929121712_6121911821_26422_842928}
```

Using 2126226 and the knowledge that there are 26 letters in the English alphabet, we made an educated guess that 21 is 'R', 26 is 'W', 22 is 'S' and 6 is 'C'. This is because 'R' and 'W' are 5 characters apart, 'R' and 'S' is 1 character apart and so on.

X=1
Y=2
Z=3
A=4
B=5
C=6
D=7
E=8
F=9
G=10
H=11
I=12
J=13
K=14
L=15
M=16
N=17
O=18
P=19
Q=20
R=21
S=22
T=23
U=24
V=25
W=26

From that point, we came up with a list that matches each number to the character.

2126226{19122929121712_6121911821_26422_842928}
R W S C{P I YF _CI P H |ER _W AS _EAYFYE}

X=1
Y=2
Z=3
A=4
B=5
C=6
D=7
E=8
F=9
G=10
H=11
I=12
J=13
K=14
L=15
M=16
N=17
O=18
P=19
Q=20
R=21
S=22
T=23
U=24
V=25
W=26

However, it was quite challenging to separate the numbers, but luckily the flag was not a random string, some words were recognisable like “CIPHER” and “WAS”. So, we were sure that we were on the right track.

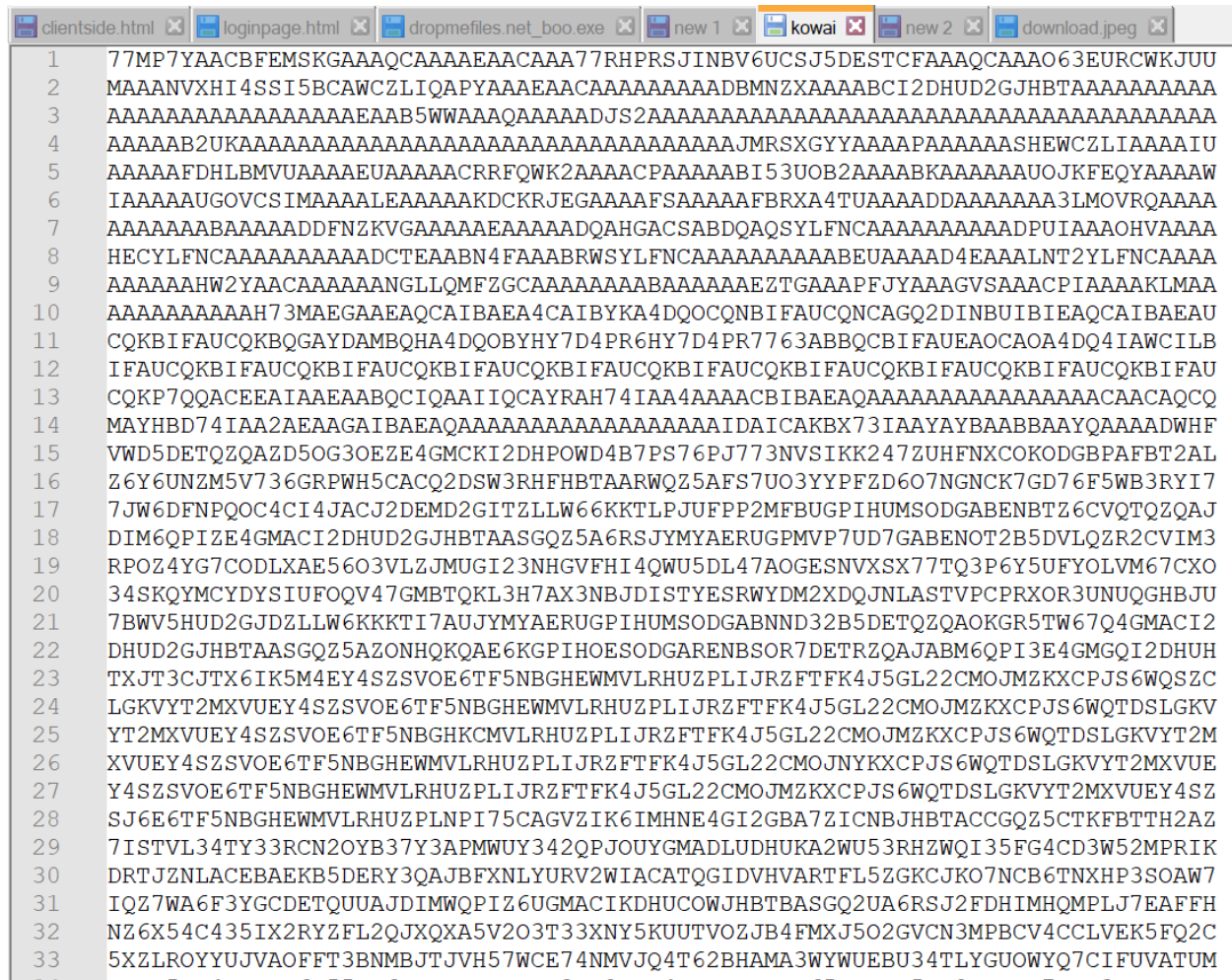
After that, we tried to take away 26s from the numbers because there are no 29th letter or 84th letter in the alphabet and there are only 26 letters in the English alphabet, and also taking away 26 would not make a difference since it is one complete round/turn, resulting in the same letter.

```
2126226{19122929121712_6121911821_26422_842928}  
R W S C{P I Z Z I N I _C I P H E R _W A S _E A Z Y}
```

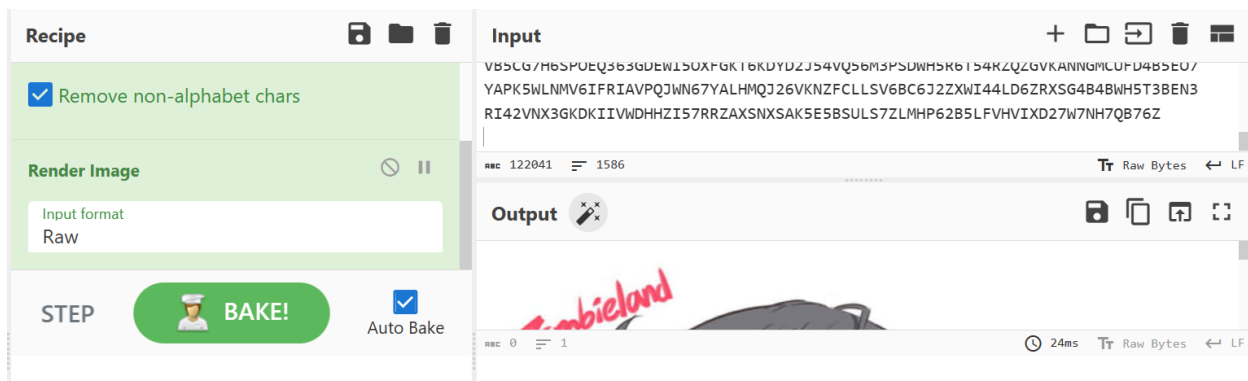
And after a few more tries to split them logically, the flag appeared.

RWSC{PIZZINI_CIPHER_WAS_EAZY}

Challenge Name: Zombeify



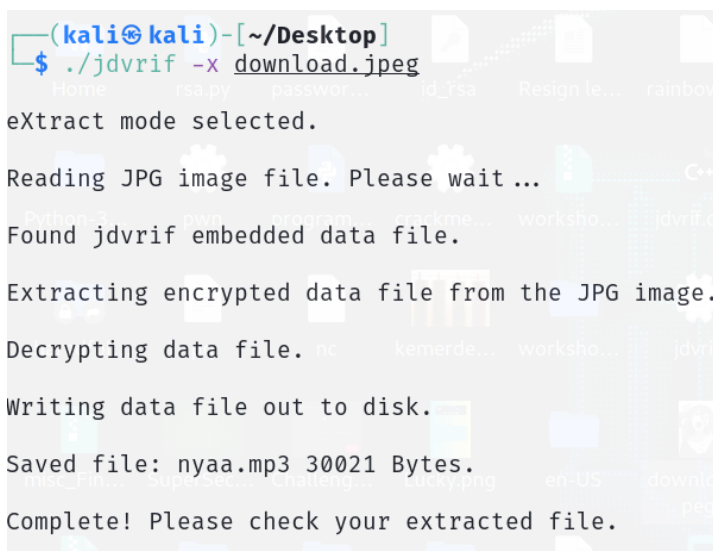
After seeing that the file cannot be opened by any applications by default, we opened the file in Notepad++ to view the contents.



Clicking on the wand adds a “Render Image” recipe and an image popped out in the output section.



From the hint given by Rydenx, we used jdvrf to extract the data from the image.



An MP3 file was hidden in the image which matches with the hint given. So, we proceeded to extract the hidden data in the MP3 file with the AudioStego tool given in the hint.

```

(kali@kali)-[~/Desktop/AudioStego/build]
$ ./hideme ~/Desktop/nyaa.mp3 -f
Doing it boss!
Looking for the hidden message ...
String detected. Retrieving it ...
Message recovered size: 43 bytes
Message: '52 57 53 43 7B 6B 75 72 30 6E 33 6B 4F 7D'♦♦U
Recovering process has finished successfully.
Cleaning memory ...

```

Using AudioStego, we found a message in the MP3 file which seems to be in hex.

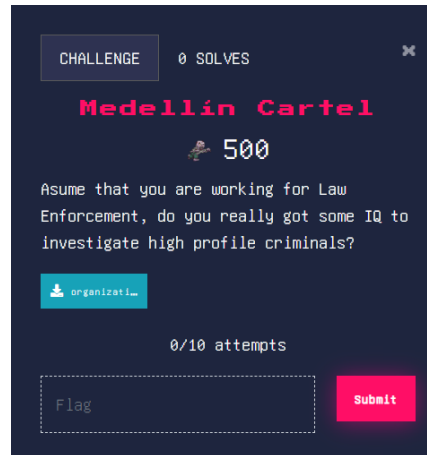
The screenshot shows the CyberChef interface with the 'From Hex' recipe selected. The input field contains the hex string '52 57 53 43 7B 6B 75 72 30 6E 33 6B 4F 7D'. The output field displays the decoded result: 'RWSC{kur0n3kO}'. The interface includes a 'Recipe' panel on the left with a 'BAKE!' button and an 'Output' panel on the right showing the result.

Putting the ciphertext inside CyberChef once again with the “From Hex” recipe revealed the flag.

RWSC{kur0n3kO}

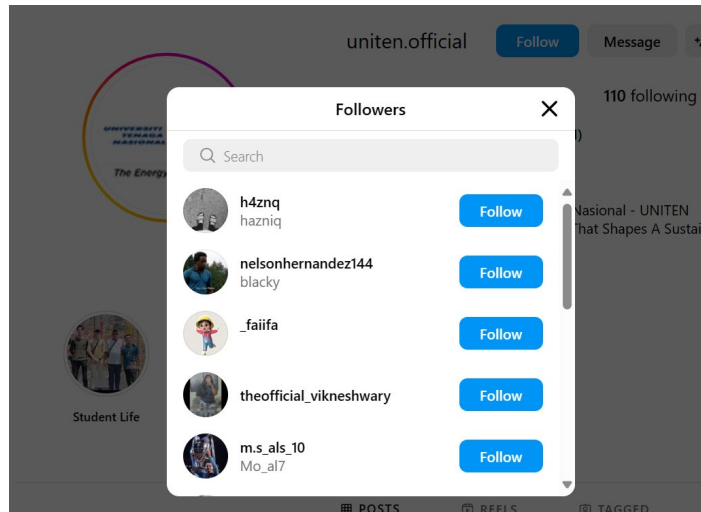
Category: OSINT

Challenge Name: Meddelin Cartel



Medellin Cartel:

- You might think the given picture, is useless. Guessy. But its not. If u are about to investigate a crime group. You have to criminal-profile every single member of the group. E.g what they do, eat, visit, etc. If u recon, Blacky AKA Nelson Hernandez is the only sicarios that has went to uniten IG. U should focus on that acc only. For real - If u went to other sicarios, u will get lost. Next step? Technical stuff. Dig the flag inside the IG, metadata is there. That is how u profile someone, dig deeper as u could on that acc.



From the hint given, we went to UNITEN's Instagram account to check the followers list and found Nelson Hernandez in the list.

https://www.instagram.com/nelsonhernandez144/



nelsonhernandez144

Follow

...

1 post

1 follower

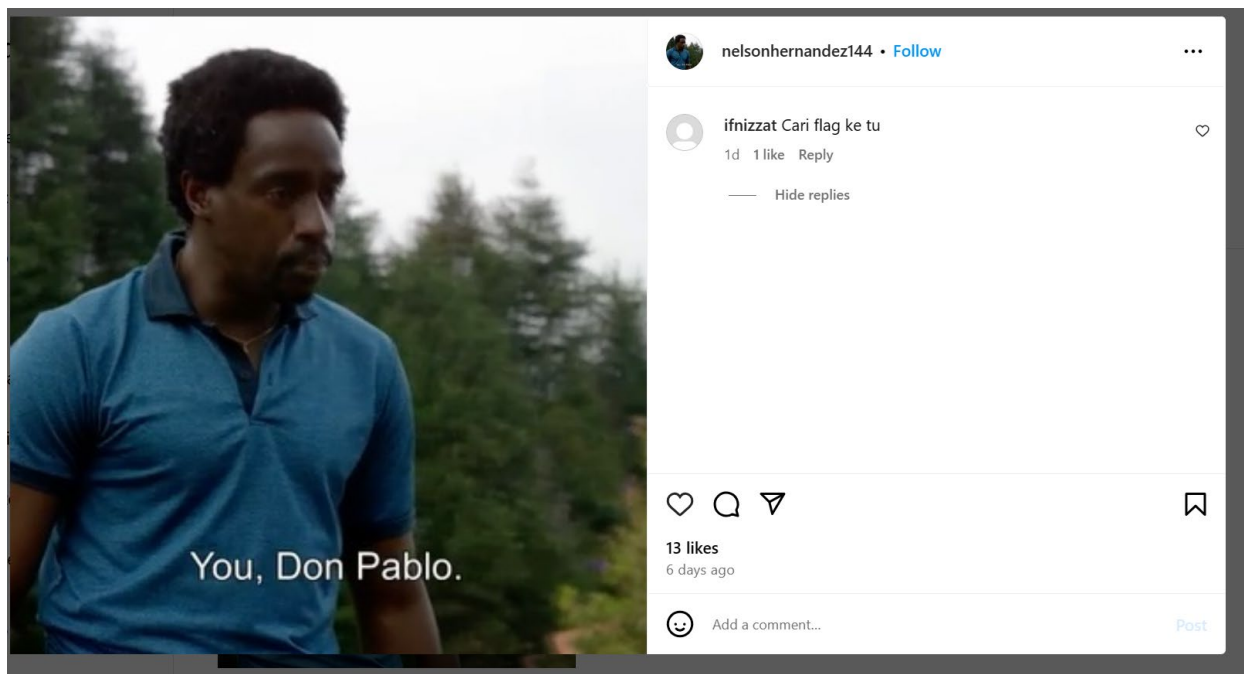
1 following

blacky

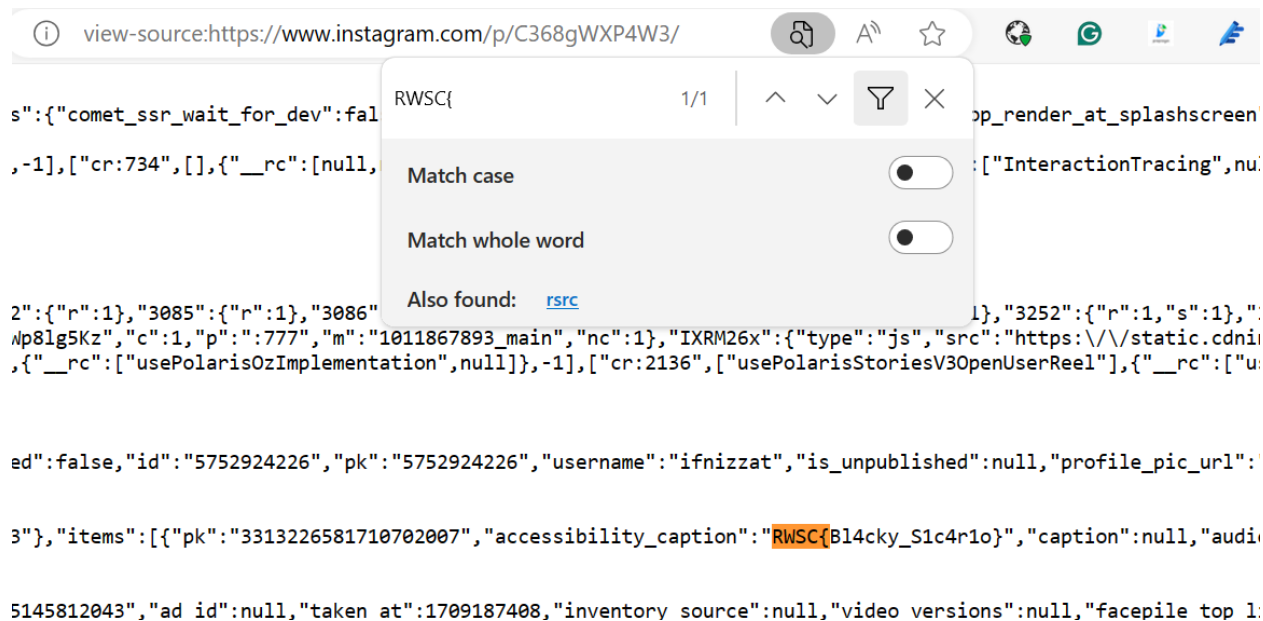
POSTS

TAGGED





After looking at the account and the posts, there was not much to see. Nothing was really related to the flag.

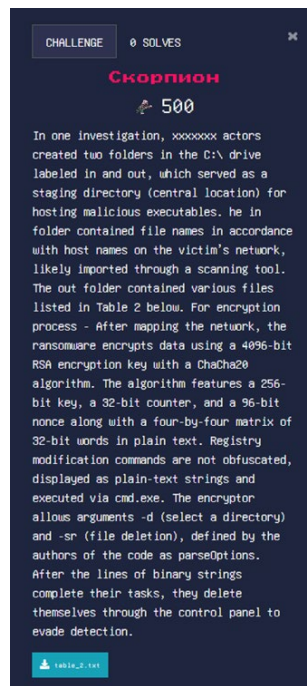


However, the hint stated that the flag is in fact inside his Instagram account. So, just as last resort, we viewed the source code of his Instagram post and did a search for "RWSC{". Unbelievably, we found the flag.

RWSC{Bl4cky_S1c4r1o}

Category: Threat Intelligence

Challenge Name: Скорпион



MyCERT
Malaysia Computer Emergency Response Team

CyberSecurity
MALAYSIA

ABOUT US SERVICES

3.2 Rhysida Ransomware Characteristics

3.2.1 Execution

In one investigation, Rhysida actors created two folders in the C:\ drive labeled in and out, which served as a staging directory (central location) for hosting malicious executables. The in folder contained file names in accordance with host names on the victim's network, likely imported through a scanning tool. The out folder contained various files listed in Table 2 below. Rhysida actors deployed these tools and scripts to assist system and network-wide encryption.

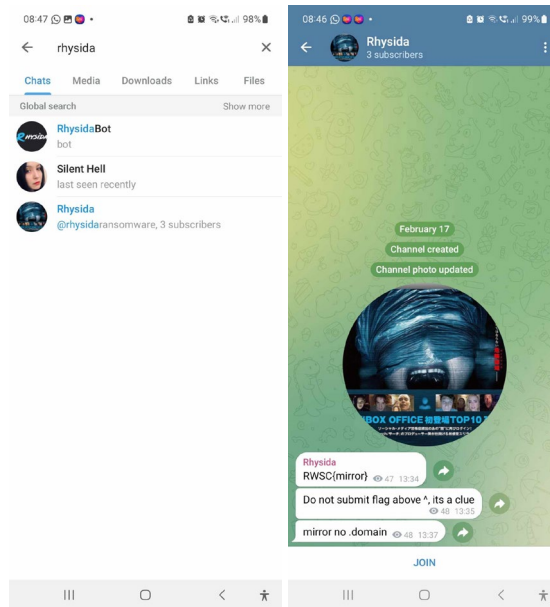
File Name	Hash (SHA256)	Description
conhost.exe	6633fa85bb234a75927b23417313e51a4c155e12f71da3959e168851a600b010	A ransomware binary.
psexec.exe	078163d5c16f64caa5a14784323fd51451b8c831c73396b967b4e35e6879937b	A file used to execute a process on a remote or local host.

From the challenge description, we know that the “xxxxxxx” stands for Rhysida from a quick search online as the contents matched with CyberSecurity Malaysia’s article on Rhysida. However, “RWSC{Rhysida}” was not the flag.

Threat Intel

- Telegram -> Darkweb

Finally, the hint from the organisers revealed that we need to go to Telegram first.



A quick search for “rhysida” reveals 3 results. After getting confirmation from the organisers that only the third “Rhysida” channel was theirs, we found the flag format and was hinted to find the mirror link that belongs to Rhysida.

new group: Rhysida #66

✓ Closed

yoryio opened this issue on May 28, 2023 · 3 comments



yoryio commented on May 28, 2023

host location

rhysidafohrhyy2aszi7bm32tnjat5xri65fopcxkdfxhi4tidsg7cad.onion

group name

Rhysida

group information

New ransomware group spotted in the Dark Web.
This group doesn't list their victims in their website.

Quick search online revealed this link, however, this was not the flag. So, we proceeded to search on Twitter as we could not find other mirror links using the browser's search engine.

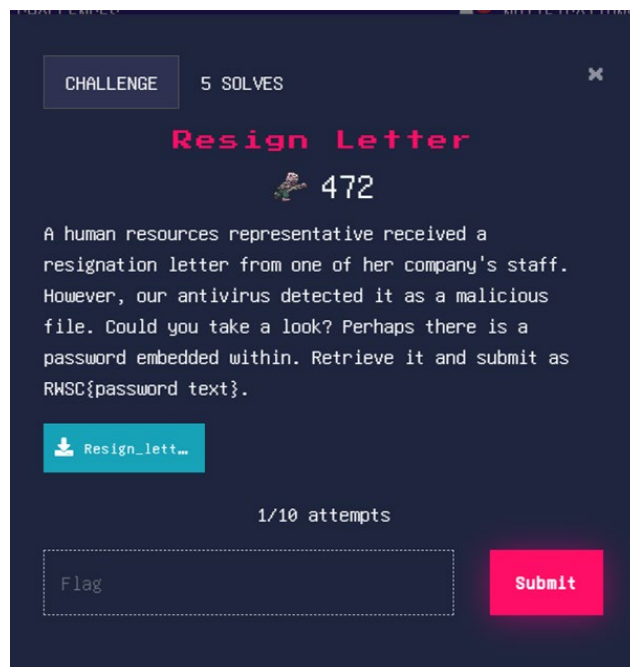


We found the link on Twitter which was posted by Dominic and submitted it as the flag.

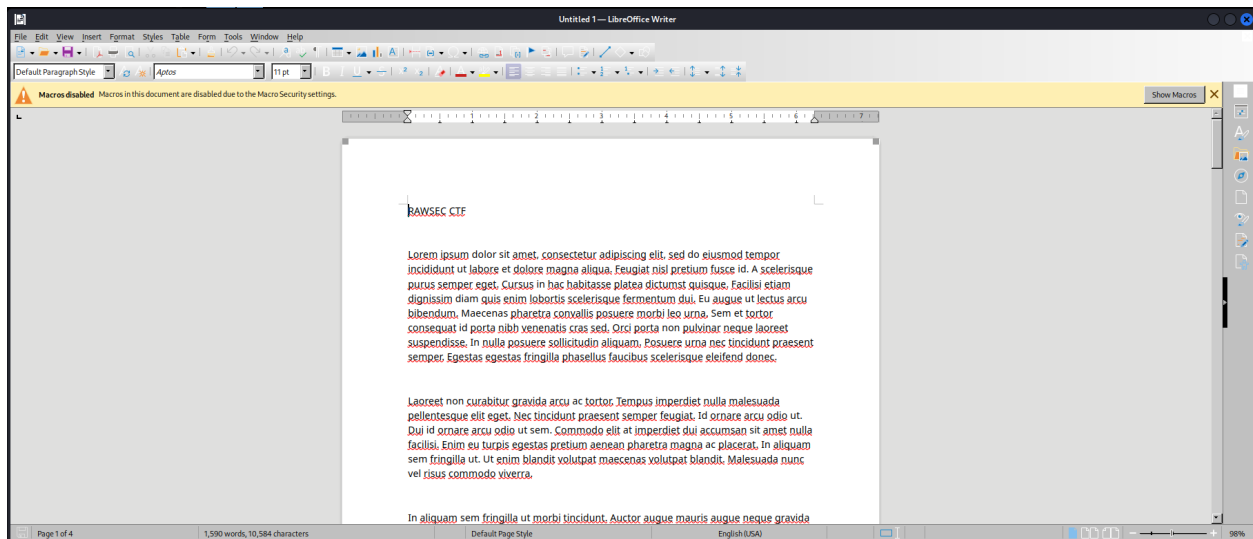
**RWSC{rhysidafc6lm7qa2mkiukbezh7zuth3i4wof4mh2audkym
scjm6yegad}**

Category: Reverse Engineering

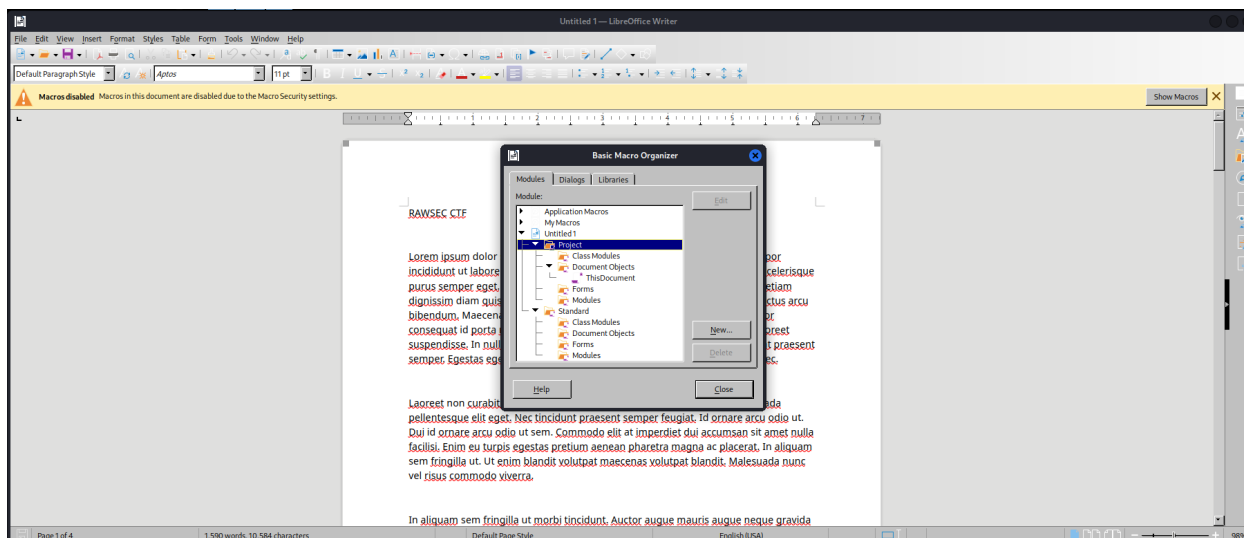
Challenge Name: Resign Letter



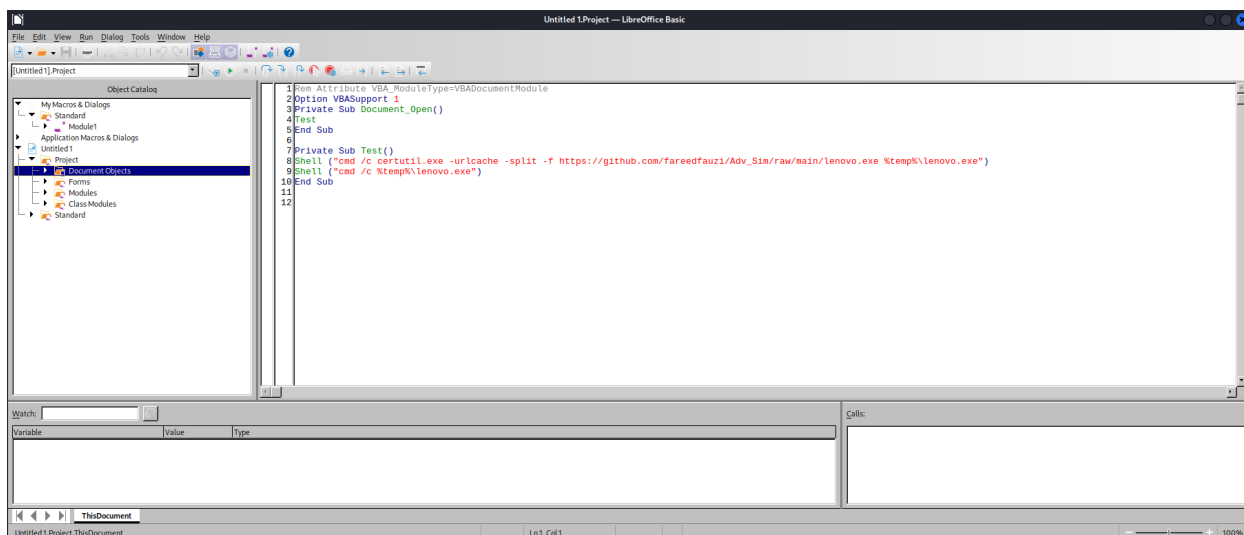
Downloading the attachment, we saw that it was a .dotm file.



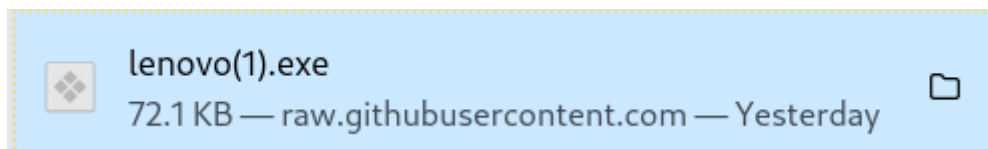
We opened the file with LibreOffice and saw that there were macros disabled.

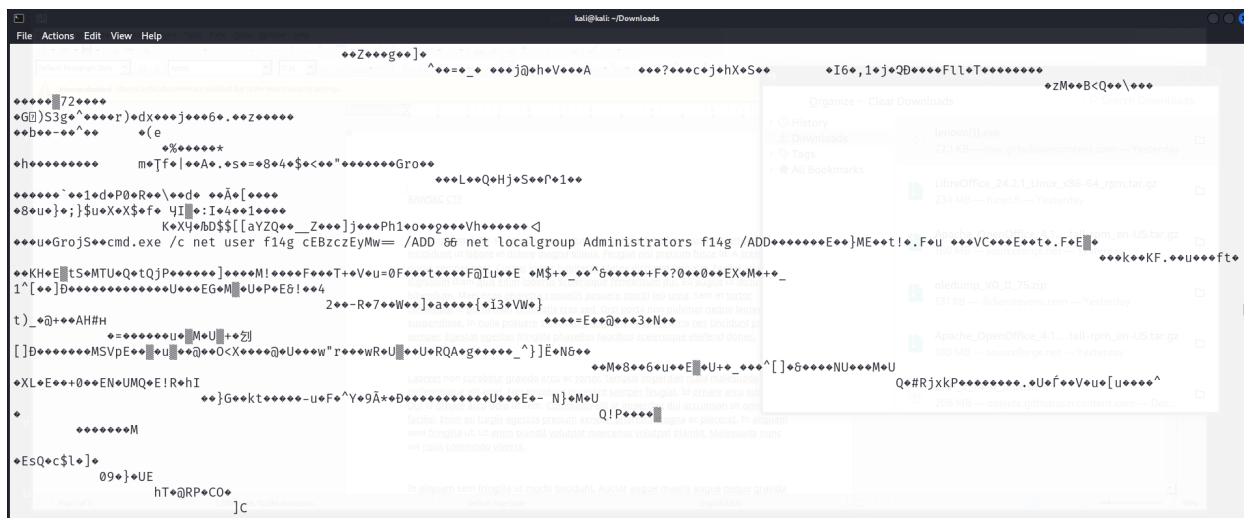


Then, we clicked on “Show Macros” and browsed through the folders that belonged to the Resign Letter Template (Untitled1).

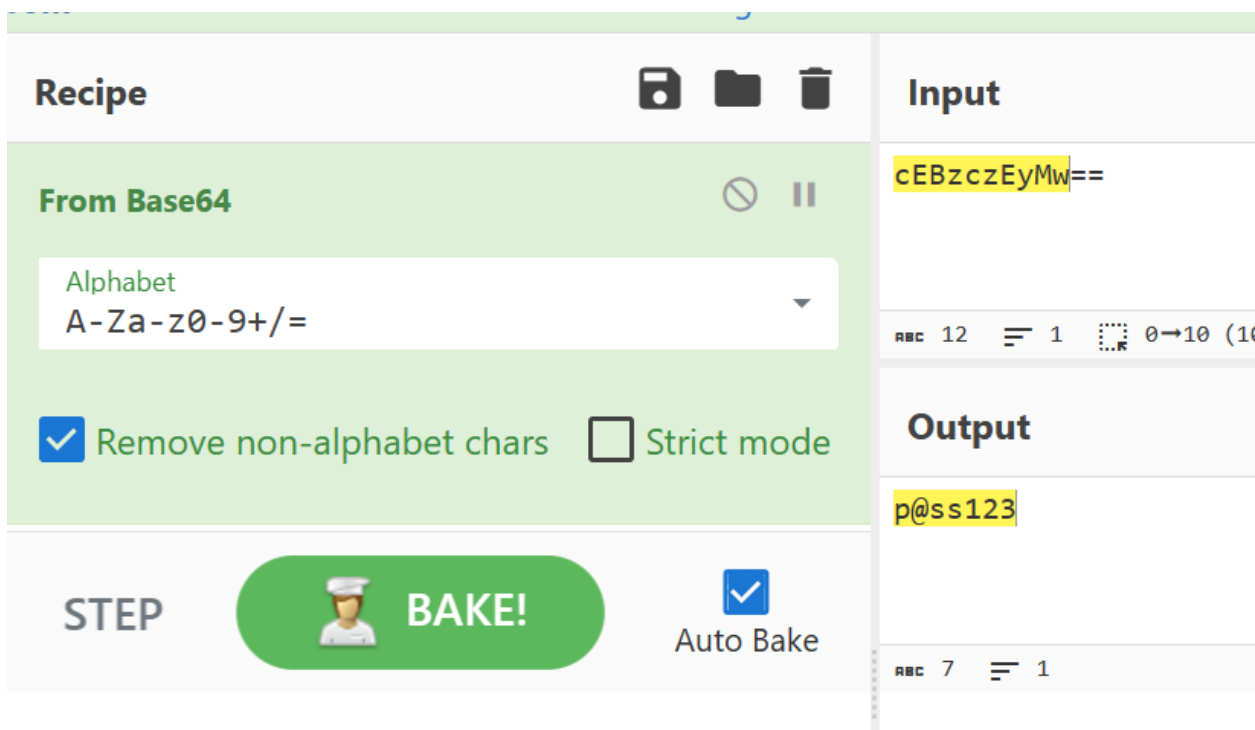


We found something named “ThisDocument” and clicked “Edit” to view it. We found a suspicious GitHub link that leads to a lenovo.exe file.





Using the link, we downloaded the .exe file and analysed the file by printing the contents in the console. We found a suspicious line where “f14g” was included. There was also an encoded string next to it, which was likely the flag.



We copied the string and pasted it in CyberChef and used the “From Base64” recipe to decode it. The output is the flag (password text).

RWSC{p@ss123}