

# WRITEUP GAME OF HACKERS (GOH) UNIKL 2023

USIM Team Name: sk1d s3c

Flag format: gohunikl2023{}

## CRYPTOGRAPHY

ReadMe (100 Points) Credit to: Danish

Given ReadMe.txt:

Dear Professional ; This letter was specially selected to be sent to you . If you no longer wish to receive our publications simply reply with a Subject: of "REMOVE" and you will immediately be removed from our mailing list . This mail is being sent in compliance with Senate bill 2416 ; Title 5 , Section 301 ! THIS IS NOT A GET RICH SCHEME ! Why work for somebody else when you can become rich within 25 DAYS . Have you ever noticed nobody is getting any younger and people are much more likely to BUY with a credit card than cash ! Well, now is your chance to capitalize on this ! WE will help YOU deliver goods right to the customer's doorstep and increase customer response by 120% ! You can begin at absolutely no cost to you ! But don't believe us . Mrs Simpson who resides in Idaho tried us and says "I've been poor and I've been rich - rich is better" . We are a BBB member in good standing . We urge you to contact us today for your own future financial well-being ! Sign up a friend and you get half off ! God Bless . Dear Internet user , You made the right decision when you signed up for our mailing list . If you are not interested in our publications and wish to be removed from our lists, simply do NOT respond and ignore this mail . This mail is being sent in compliance with Senate bill 1624 ; Title 3 ; Section 309 . Do NOT confuse us with Internet scam artists . Why work for somebody else when you can become rich inside 95 weeks ! Have you ever noticed nearly every commercial on television has a .com on it & more people than ever are surfing the web ! Well, now is your chance to capitalize on this . We will help you decrease perceived waiting time by 130% and increase customer response by 120% . You can begin at absolutely no cost to you ! But don't believe us ! Mr Ames of New Jersey tried us and says "I was skeptical but it worked for me" ! This offer is 100% legal ! Do not go to sleep without ordering ! Sign up a friend and you'll get a discount of 30% . Thank-you for your serious consideration of our offer .

**CIPHER IDENTIFIER**  
Cryptography > Cipher Identifier

**ENCRYPTED MESSAGE IDENTIFIER**

★ CIPHERTEXT TO RECOGNIZE ?  
time by 150% and increase customer response by 120%. You can begin at absolutely no cost to you ! But don't believe us ! Mr Ames of New Jersey tried us and says "I was skeptical but it worked for me" ! This offer is 100% legal ! Do not go to sleep without ordering ! Sign up a friend and you'll get a discount of 30% . Thank-you for your serious consideration of our offer

★ CLUES/KEYWORDS (IF ANY) gohunik12023{

► ANALYZE

See also: Frequency Analysis – Index of Coincidence

**SYMBOLS IDENTIFIER**

► Go to: Symbols Cipher List

**Answers to Questions (FAQ)**

**What is a cipher identifier? (Definition)**

A encryption detector is a computer tool designed to recognize encryption/encoding from a text message. The detector performs cryptanalysis, examines various features of the text, such as letter distribution, character repetition, word length, etc. to determine the type of encryption and guide users to the right tools based on the type of encryption identified.

**How to decrypt a cipher text?**

To decrypt / decipher an encoded message. it is necessary to know the

We tried the Acrostic Extractor, Trevanian Cipher and Cardan Grille but they do not end well. So we search spamming decoder on the Internet.

Google - spamming decoder

About 1,100,000 results (0.34 seconds)

**Sponsored**

- TitanHQ https://www.titanhq.com : SpamTitan™ Spam Filtering – Impersonation Protection Filters 99.9% Of Email Threats. Online Spam Filtering. Book a Demo With Our Experts Today. Need a Phishing Protection Solution? Try SpamTitan. Protect your Business, Brand...
- spammimic https://spammimic.com : Decode Paste in a spam-encoded message: Alternate decodings: Decode spam with a password · Decode fake spreadsheet · Decode fake PGP · Decode fake Russian ...
- spammimic - hide a message in spam Hide messages in spam ... Decode · Explanation · Credits · FAQ & Feedback · Terms · Français. First time here ... Decode · Encode · Explanation · Terms of service

People also ask :

The screenshot shows a web browser window with the URL [spammimic.com/decode.shtml](http://spammimic.com/decode.shtml). The page title is "Decode". On the left, there's a sidebar with links: Encode, Decode (which is highlighted), Explanation, Credits, FAQ & Feedback, Terms, and Français. The main content area has a heading "Paste in a spam-encoded message:" followed by a large text box containing a spam message. The message is as follows:

. Dear Internet user , You made the right decision  
when you signed up for our mailing list . If you are  
not interested in our publications and wish to be removed  
from our lists, simply do NOT respond and ignore this  
mail . This mail is being sent in compliance with Senate  
bill 1624 ; Title 3 ; Section 309 . Do NOT confuse  
us with Internet scam artists . Why work for somebody  
else when you can become rich inside 95 weeks ! Have  
you ever noticed nearly every commercial on television  
has a .com on in it & more people than ever are surfing  
the web ! Well, now is your chance to capitalize on  
this . We will help you decrease perceived waiting  
time by 130% and increase customer response by 120%  
. You can begin at absolutely no cost to you ! But  
don't believe us ! Mr Ames of New Jersey tried us and  
says "I was skeptical but it worked for me" ! This  
offer is 100% legal ! Do not go to sleep without ordering  
! Sign up a friend and you'll get a discount of 30%  
. Thank-you for your serious consideration of our offer

Below the text box is a button labeled "Decode" which is circled in pink. Underneath it, it says "Alternate decodings:" followed by two bullet points:

- Decode spam with a password
- Decode fake spreadsheet [\[PDF\]](#)

The screenshot shows a web browser window with the URL [spammimic.com/decode.cgi](http://spammimic.com/decode.cgi). The page title is "Decoded". The sidebar on the left is identical to the previous page. The main content area has a heading "Decoded" and a message: "Your spam message Dear Professional ; This letter was spec... decodes to: gohunikl2023{f0und\_Me}". There is also an "Encode" button next to the decoded text. Below this, it says "Look wrong?, try the [old version](#)". At the bottom, there is a copyright notice: "Copyright © 2000-2023 spammimic.com, All rights reserved".

The flag is gohunikl2023{f0und\_Me}

## Cipher 101 (100 Points)

PS: About two teams solves

Given text.txt:

haFep aaoicehan ixhcs igtnt pf borsTnptmaireinyotei heieobo ri heirenemigtnr latoi s rnu d redolTnefiu entcltfect.wrettiihyaFep veloyns iiosa co reo pst nb rifsront.ruhe,eai GN2{PANTeRi ec ihri rnpsto ihta ragspanetcaatr nazga atr cosaseiidhme f"al. oecyt h esg switndaao In hs al,adtecpxt sra f yrw.Frdcyto,tecpxt switni h aeza atr costeris n h rgmlmsaei eosrc e raigcaatr ignly h ubro al sddtria h opeyo h npin oee,dsieissmlct,teRi ec ihri unbet rpaayi,mkn tmr utbefreuinlo erainlproe ahrt ouetcyo o euecmuiain o hscalne h lggvni OUIL03CYTWSU} InCestsinp trelt rei zpeas ceur i" r,ese t glagersnhirt d soepnhirt t os zpeasha,deiaegscctbenhcsaa.em rseemsemx teyoHv p pi InCesla ctslag eliodtarctausrehrsnphrc mcoFt lgf esHK2ROF

The screenshot shows the 'Cipher Identifier' tool from dCode. The main area displays a list of cipher types: Transposition Cipher, Double Transposition Cipher, Substitution Cipher, Skip Cipher, Shift Cipher, Homophonic Cipher, Rail Fence (Zig-Zag) Cipher, Caesar Box Cipher, Scytale Cipher, Redefence Cipher. Below this is a detailed analysis of the provided text, which includes a 'SYMBOLS IDENTIFIER' link highlighted by a pink arrow.

Although it said it is Transposition Cipher, but it does not end well too. So, Rail Fence Cipher is pretty common these days.

The screenshot shows the 'Rail Fence (Zig-Zag) Cipher' tool from dCode. It provides a detailed description of the cipher and its parameters. The 'AUTOMATIC DECRYPTION' button is highlighted by a pink arrow. The results section also contains a Caesar Box Cipher section and a summary of related topics.

We need to hit the AUTOMATIC DECRYPTION until it shows up readable texts with flag.  
The flag is GOUNIKL2023(CRYPTOWASFUN)

## Train Of Bleu (150 Points)

PS: About three teams managed to get the flag

Given Secret\_Recipe.zip (AES encryption password) and P4ssc0de.txt:

"uasoo ciog t onbrew rmcnhrsfeoml dhhteo h tuef idetieanjtgssr etea srtooh eaitph guhpciruennoosatsaxteeotr webn r k irsnsnhor eihifvws aeedaeonsteoi stgya tcfo.as ndoia rdie t e neefhnetteoicusnoogs tth snpneoeifaup airossrlmb trfa,lsofa,eqbb ta,nrsiuu p tubidhc iinzugp kseoneeit saeqhaena cit th.issx rpomi nsfsisai etiiw negerisrrt hscnygh veh o rane e rrttrnrphasa lrtae hacoeai t ei ondmeHhsc1slennea iio ifta ocsâ€" Oleresc3p 0cb h tnunfle0lelrouraaelct ulut ayldrn ry sysfnbt setcc s lifturoaf ni"leaisevursainoain.os hovery ntd ndgss stui tsesn yd iede rse ihctr scrlel.cusfdtevhttiaydewelsoe nnesrrtp rcedce hdo iaoy Gatf xnewtscatnp fu eci rso

Cipher Identifier from dcode.fr and boxentriq cannot detect which cipher. Once again, we use the Rail Fence Cipher. This time, we need to use the Rail Fence Decoder in CyberChef. Why? Because given the hint number 40 from the file name itself, P4ssc0de.

The screenshot shows the CyberChef interface with the "Rail Fence Cipher Decode" operation selected. The input field contains the provided ciphertext. The "Key" field is set to 40, and the "Offset" field is set to 0. The output section displays the decrypted message: "in the heart of paris lies a closely Guarded secret—a recipe passed down through generations, known to Only a select few. this culinary masterpiece boasts a fusion of unique ingredients sourced exclusively from the cobblestone streets of paris. combining the elusive essence of truffles Harvested beneath the moonlit skies of the french c0untryside with the subt13 sophistication of aged bordeaux vinegar, this recipe transcends conventional flavors. infused with the whispers of parisian hist0ry and the artistry of local artisans, this secret creation tantalizes taste buds and transports diners on an exquisite gastronomic journey through the enchanting city of paris."

Output retrieved:

"in the heart of paris lies a closely Guarded secret—a recipe passed down through generations, known to Only a select few. this culinary masterpiece boasts a fusion of unique ingredients sourced exclusively from the cobblestone streets of paris. combining the elusive essence of truffles Harvested beneath the moonlit skies of the french c0untryside with the subt13 sophistication of aged bordeaux vinegar, this recipe transcends conventional flavors. infused with the whispers of parisian hist0ry and the artistry of local artisans, this secret creation tantalizes taste buds and transports diners on an exquisite gastronomic journey through the enchanting city of paris."

We noticed unusual capital letters and numbers, so we can take it as a clue to unlock the password-protected Secret\_Recipe.zip file.

The clue is GOH0310 (password for Secret\_Recipe.zip)

After unzipping the Secret\_Recipe.zip, we obtained a Secret\_Recipe.txt:

Parisian Secret Croissants

Ingredients:

2 1/4 cups all-purpose flour  
1/4 cup granulated sugar  
1 teaspoon salt

1 tablespoon active dry yeast  
 1/2 cup warm water  
 1 cup cold unsalted butter, sliced thinly  
 1/2 cup milk  
 1 egg, beaten (for egg wash)

Instructions:

- Activate the Yeast: In a small bowl, dissolve the yeast in warm water. Let it sit for 5-10 minutes until frothy.
- Prepare the Dough: In a large mixing bowl, combine flour, sugar, and salt. Add the activated yeast mixture and milk. Mix until a dough forms.
- Incorporate the Butter: Roll out the dough on a floured surface into a rectangle. Place the thinly sliced cold butter evenly over two-thirds of the dough. Fold the unbuttered third over the buttered middle third, then fold the other buttered third over the top. This creates three layers.
- Chill the Dough: Wrap the dough in plastic wrap and refrigerate for 30 minutes.
- Roll and Fold: Remove the dough from the refrigerator and roll it out into a rectangle again. Fold it into thirds, as before. Chill for another 30 minutes.
- Repeat Rolling and Folding: Repeat the rolling and folding process two more times, chilling the dough for 30 minutes between each step. This creates the flaky layers characteristic of croissants.
- Shape the Croissants: Roll out the dough into a large rectangle and cut it into triangles. Roll each triangle starting from the wide end towards the tip to form the classic croissant shape.
- Proofing: Place the shaped croissants on a baking sheet lined with parchment paper. Cover them loosely with a kitchen towel and let them proof in a warm place for about 1-2 hours, or until doubled in size.
- Preheat and Bake: Preheat the oven to 375°F (190°C). Brush the croissants with beaten egg for a shiny finish. Bake for 15-20 minutes or until golden brown.
- Enjoy the secret recipe from  
 Vm0weGQxTXIVWGhVYmtwUFZtMW9WVII3WkRSV01XeH1xa2M1VmxCk2JETldiWFF3WVZVeFYxTnNiR1ZpUjFGM1ZrUkdZV014VG5OYVJtUlhUVEZKZWxkWGRHdFRNVmw0Vj1R1ZXSkdXbTIVVnpGdIRXeGFjVk5ZYUZkTmF6VjVWR3hhYfSc1duTmpSbWhXhYV1ZkMGQxVXhVbGhsUm5Cc1ZsUkdSbFZYTVCVWJGcFdZMFpTVjFaV2NGTmFSRVpEvId4Q1ZVMUVNRDA9.

This secret recipe holds the key to the quintessential Parisian croissant—crispy on the outside, tender and flaky on the inside, a true delight for the senses!

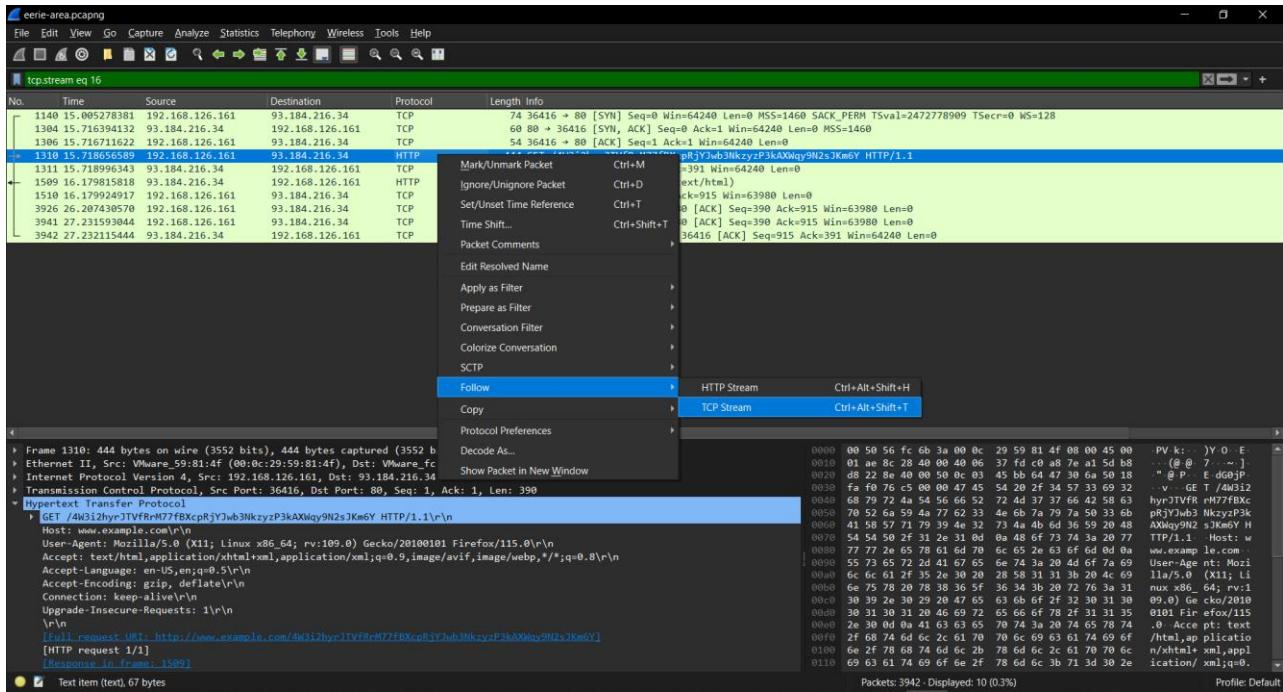
The screenshot shows the CyberChef interface with four stacked "From Base64" sections. Each section has "Remove non-alphabet chars" checked and "Strict mode" unchecked. The first section's input is the long Base64 string provided above. The second section's input is the output of the first. The third section's input is the output of the second. The fourth section's input is the output of the third. The final output is "gohunikl2023{N1c3\_Cyb3R\_ch3f}".

Use CyberChef again using From Base64 8 times to retrieve the flag: gohunikl2023{N1c3\_Cyb3R\_ch3f}

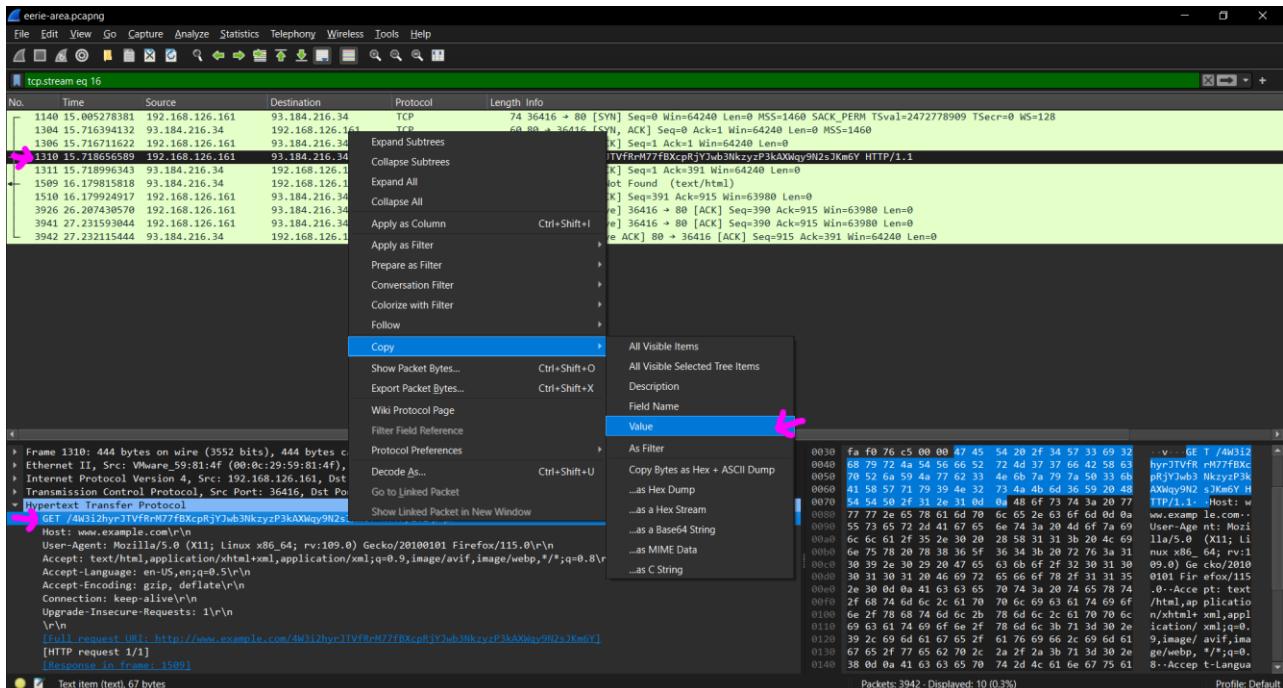
# FORENSIC

## The realm of Eerie-Area (150 Points) Credit to: Danish

Given eerie-area.pcapng



First, you need to right-click on anything with TCP Protocol. We found something unusual text which is a Base58 text. Copy the value of the HTTP.



4W3i2hyrJTVfRrM77fBXcpRjYJwb3NkzyzP3kAXWqy9N2sJKm6Y

The screenshot shows a web browser with two tabs open. The active tab is 'Base58 Converter - Online Base' at dcode.fr/base-58-cipher. The URL in the address bar is https://dropmefiles.net/en/5BhkPHCz4Q. The page contains a form for decoding Base58 strings into ASCII characters, with various output formats like Hexadecimal, Decimal, Octal, Binary, and Integer Number options. Below the form, there's a section for 'FROM A NUMBER' where an integer can be converted to Base 58. On the right side, there's a sidebar with links to similar tools like Base64, Base64 Coding, and Base64 Encoding.

A website was retrieved : <https://dropmefiles.net/en/5BhkPHCz4Q>

The screenshot shows a web browser displaying a file download page from dropmefiles.net. The page has a header 'DropMeFiles' with a 'STOP WAR IN UKRAINE' banner. Below the header, it says 'List of files:' and shows a single file entry: 'boo.exe' (197 B). There are download and share buttons next to the file entry. At the bottom, there are statistics: 41 likes, 0 dislikes, and a timestamp of 00:46 19.11.23. A 'DOWNLOAD ALL' button is also present.

boo.exe retrieved that can't be executed in Windows. Every time we obtained a suspicious file. Use Kali Linux to use command "file" to check the exact file extension.

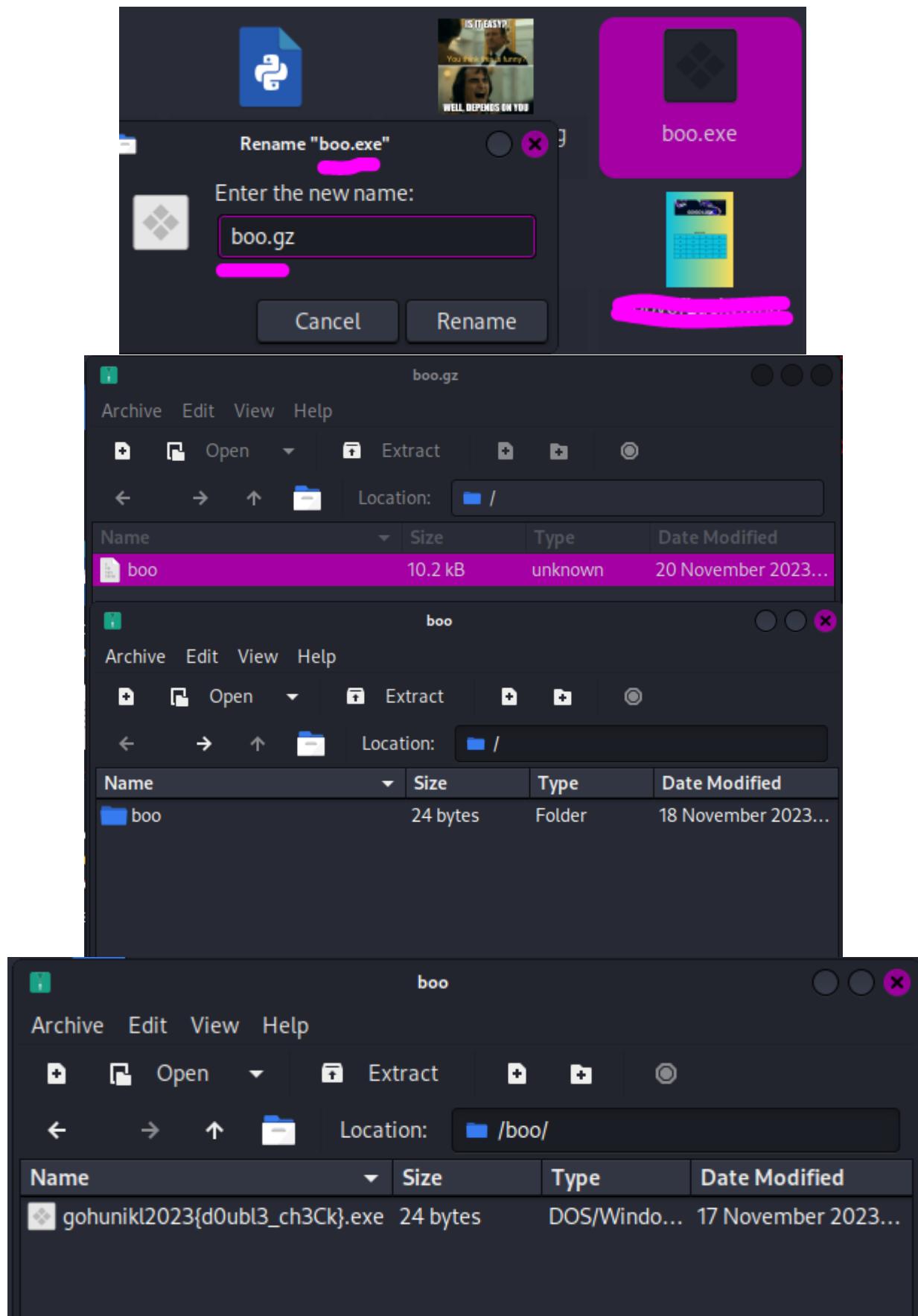
```

File Actions Edit View Help
(kali㉿kali)-[~/Downloads]
$ file boo.exe
boo.exe: gzip compressed data, from Unix, original size modulo 2^32 10240
Places
(kali㉿kali)-[~/Downloads]
$ 

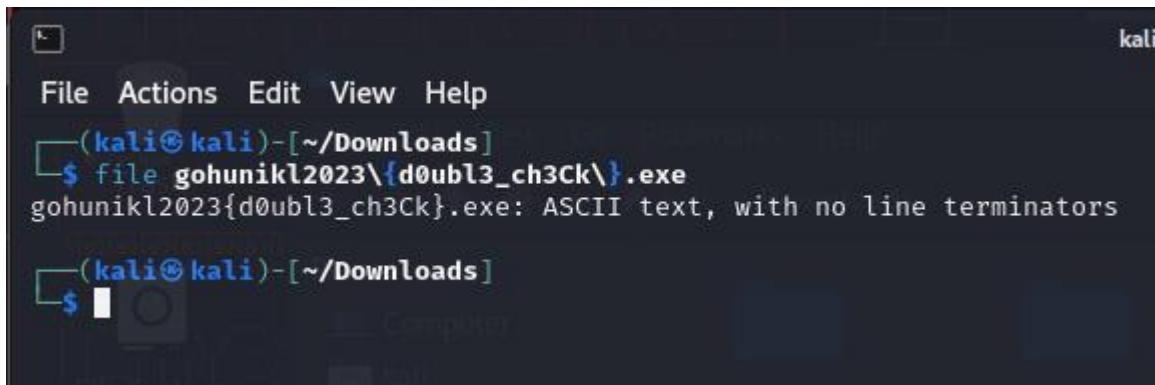
```

The screenshot shows a terminal window on a Kali Linux system. The user is in the ~/Downloads directory. They run the 'file' command on the file 'boo.exe'. The output indicates that 'boo.exe' is a gzip compressed data file from Unix, with an original size of 10240 bytes. The terminal interface includes a navigation bar with icons for file operations and a status bar at the bottom.

So it should be in .gz file format so rename it with .gz extension. Then, unzip it to retrieve an object named boo.

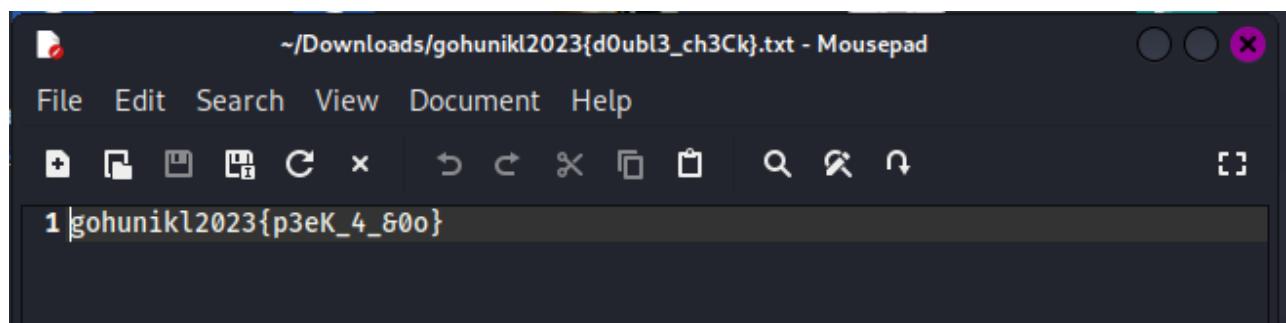


So this is not the flag yet. Double check means you have to use the 'file' Kali Linux command again.



```
(kali㉿kali)-[~/Downloads]
$ file gohunikl2023{d0ubl3_ch3Ck}.exe
gohunikl2023{d0ubl3_ch3Ck}.exe: ASCII text, with no line terminators
```

Rename it to .txt



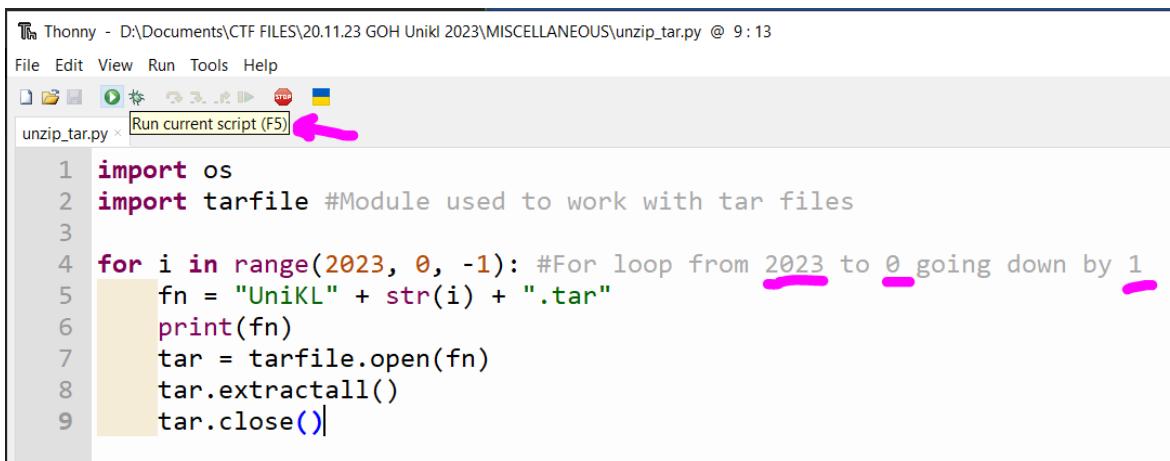
The flag is gohunikl2023{p3eK\_4\_&0o}

## MISCELLANEOUS

### Just Unzip (100 Points)

Given UniKL2023.tar (a compressed file that can be extracted). First extract will get us UniKL2022.tar

So we need to craft a small Python script for it in order to unzip it until it gets UniKL1.tar (make sure the Python script in the same folder with UniKL2023.tar)



```
Thonny - D:\Documents\CTF FILES\20.11.23 GOH UniKL 2023\MISCELLANEOUS\unzip_tar.py @ 9:13
File Edit View Run Tools Help
Run current script (F5) (highlighted)
unzip_tar.py
1 import os
2 import tarfile #Module used to work with tar files
3
4 for i in range(2023, 0, -1): #For loop from 2023 to 0 going down by 1
5     fn = "UniKL" + str(i) + ".tar"
6     print(fn)
7     tar = tarfile.open(fn)
8     tar.extractall()
9     tar.close()
```

```

import os
import tarfile #Module used to work with tar files

for i in range(2023, 0, -1): #For loop from 2023 to 0 going down by 1
    fn = "UniKL" + str(i) + ".tar"
    print(fn)
    tar = tarfile.open(fn)
    tar.extractall()
    tar.close()

```

PS: From 20mb to about 15GB of total files XD

|   |                     |          |           |
|---|---------------------|----------|-----------|
|  UniKL1.tar    | 11/20/2023 9:30 PM  | TAR File | 10 KB     |
|  UniKL2023.tar | 11/21/2023 12:20 PM | TAR File | 20,230 KB |

unzip the UniKL1.tar and we will obtain a picture of flag.png

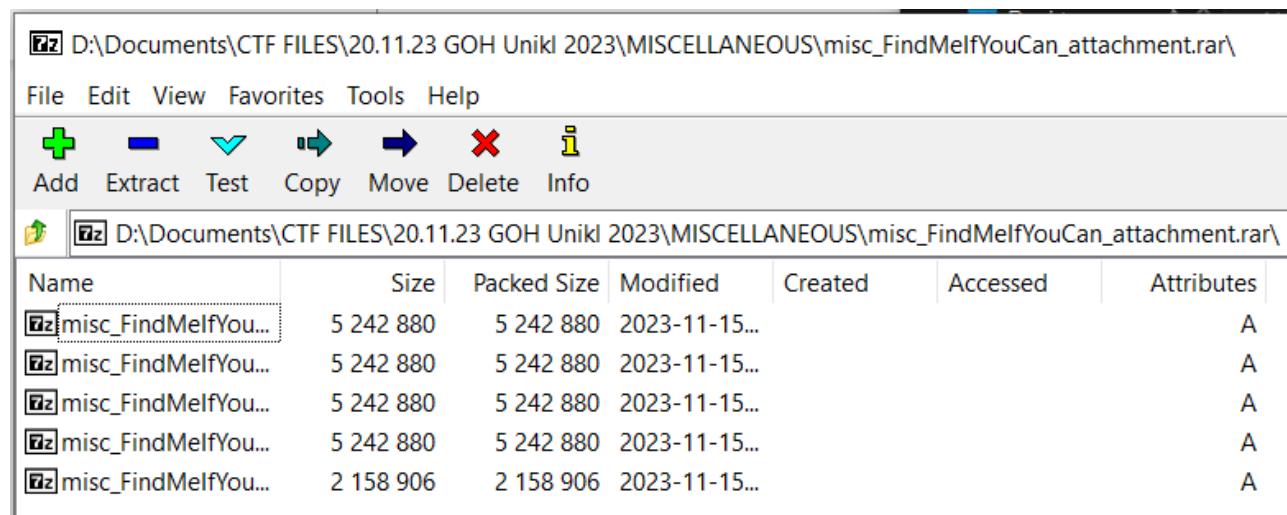
# gohunkl2023{1\_H0p3\_U\_D0nt\_d0\_1t\_2023\_TiMe}

The flag is gohunkl2023{1\_H0p3\_U\_D0nt\_d0\_1t\_2023\_TiMe}

## FindMelfYouCan (100 Points)

PS: About 8 teams found the flag

From misc\_FindMelfYouCan\_attachment.rar , we need to find flag



| Name  | Size      | Packed Size | Modified      | Created | Accessed | Attributes |
|---|-----------|-------------|---------------|---------|----------|------------|
|  misc_FindMelfYou... | 5 242 880 | 5 242 880   | 2023-11-15... |         |          | A          |
|  misc_FindMelfYou... | 5 242 880 | 5 242 880   | 2023-11-15... |         |          | A          |
|  misc_FindMelfYou... | 5 242 880 | 5 242 880   | 2023-11-15... |         |          | A          |
|  misc_FindMelfYou... | 5 242 880 | 5 242 880   | 2023-11-15... |         |          | A          |
|  misc_FindMelfYou... | 2 158 906 | 2 158 906   | 2023-11-15... |         |          | A          |

| D:\Documents\CTF FILES\20.11.23 GOH Unikl 2023\ |        |             |   |
|---|--------|-------------|---|
| File Edit View Favorites Tools Help             |        |             |   |
| Add Extract Test Copy Move Delete Info          |        |             |   |
| Name  | Size   | Packed Size | I |
| 1b2ef063cda82c6...                              | 57 387 | 57 387      |   |
| 1b4ceacb516a02...                               | 57 387 | 57 387      |   |
| 1e2e2269819f0e0...                              | 57 387 | 57 387      |   |
| 2b0b4b542ebabc...                               | 57 387 | 57 387      |   |
| 2c9dd70a7159013...                              | 57 387 | 57 387      |   |
| 3b5dd3aeff8d39...                               | 57 387 | 57 387      |   |
| 4eacec592d1c8c2...                              | 57 387 | 57 387      |   |
| 5c2537c433125fe1...                             | 57 387 | 57 387      |   |
| 8bd546779a3880c...                              | 57 387 | 57 387      |   |
| 15db8eab4797094...                              | 57 387 | 57 387      |   |
| 25b60de67c22d68...                              | 57 387 | 57 387      |   |
| 29edbceb4dff9daf...                             | 57 387 | 57 387      |   |
| 52bcd65c8713e03...                              | 57 387 | 57 387      |   |
| 65b237f72251891...                              | 57 387 | 57 387      |   |
| 67e0620747566f3...                              | 57 387 | 57 387      |   |
| 099be1658be9451...                              | 57 387 | 57 387      |   |
| 337e9c26308a4a1...                              | 57 387 | 57 387      |   |
| 384d16e945d68cd...                              | 57 387 | 57 387      |   |
| 937b68d455b71c7...                              | 42 864 | 42 864      |   |
| f044e26811f6ff93...                             | 57 387 | 57 387      |   |

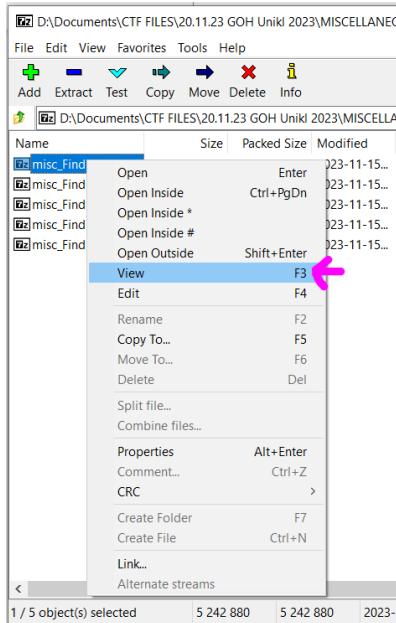
| D:\Documents\CTF FILES\20.11.23 GOH Unikl 2023\MISCELLANEOUS\misc_FindMelfYouCan_attachment.rar\misc_FindMelfYouCan_attachment |       |             |          |         |          |            |           |
|--|-------|-------------|----------|---------|----------|------------|-----------|
| File Edit View Favorites Tools Help  |       |             |          |         |          |            |           |
| Add Extract Test Copy Move Delete Info   |       |             |          |         |          |            |           |
| Name   | Size  | Packed Size | Modified | Created | Accessed | Attributes | Encrypted |
| 1cd55ecd700bcae...   | 2 867 | 2 867       |          |         |          |            |           |
| 2affda98dbe7e4a...   | 2 867 | 2 867       |          |         |          |            |           |
| 2b5bf0e8b797cf1...   | 2 867 | 2 867       |          |         |          |            |           |
| 9d970269ee058cff...  | 2 867 | 2 867       |          |         |          |            |           |
| 11eab125e519ac1...   | 2 867 | 2 867       |          |         |          |            |           |
| 27e885d7f0a9225...   | 2 867 | 2 867       |          |         |          |            |           |
| 55c0e3905d85c1e...   | 2 867 | 2 867       |          |         |          |            |           |
| 55ea24b3057027c...   | 2 867 | 2 867       |          |         |          |            |           |
| 58f29cf9c8ddfbb5...  | 2 867 | 2 867       |          |         |          |            |           |
| 94b0c755f9e8f8ee...  | 2 867 | 2 867       |          |         |          |            |           |
| 864dc273bd2009f...   | 2 867 | 2 867       |          |         |          |            |           |
| 983cc91afffb4343f...   | 2 867 | 2 867       |          |         |          |            |           |
| 442163307ff94b0e...  | 2 867 | 2 867       |          |         |          |            |           |
| a12fafdb2fe5dc84...  | 2 867 | 2 867       |          |         |          |            |           |
| b2f8f13682b7358...   | 2 867 | 2 867       |          |         |          |            |           |
| baa2677b81ad7e4...   | 2 867 | 2 867       |          |         |          |            |           |
| d5e20b1b67cbac2...   | 2 867 | 2 867       |          |         |          |            |           |
| dd554b5b8ef995f...   | 2 867 | 2 867       |          |         |          |            |           |
| ea47bb5b036c5d9...   | 2 867 | 2 867       |          |         |          |            |           |
| ec4ad9c6008fbea...   | 2 867 | 2 867       |          |         |          |            |           |
| flag.txt   | 47    | 47          | 2        |         |          |            |           |

flag - Notepad

```
gohunikl2023{zV6MSLHT.404_4pKV35XYeQ-QSqwN6Bz}
```

As you can see, there are a lot of folders inside folders, and sometimes you will obtain the flag.txt containing fake flags (totally a lot, like every folder has one each)

However, there's a tool that ease your flag finding by using this step:



This output window will be shown:

```

m1e26811f6ff930b9c4feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4a1c8afdbable2ed48db20ba8a32|15916f58531b2a723ec467817acf1511.flag.txt ::@|gohuni1k2023(zRwtI2VYcryEiWkSP)
:f6ff930b94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4a1c8afdbale2ed48db20ba8a32|18e3b9aa84fa932abf1.flag.txt ::@|gohuni1k2023(axbXGHTy_4044_DnVC9wfP;UtgB2Gz}
:f6ff930b94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4a1c8afdbale2ed48db20ba8a32|26afea171eb89cc7832cd7d8aa9fe91.flag.txt ::@|gohuni1k2023(KEKA5Fn_4044_yqp215Fk8ckth15qgt}
:f6ff930b94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4a1c8afdbale2ed48db20ba8a32|385cf2dd9e58438b2e70179b98c9ba3.flag.txt ::@|gohuni1k2023(eY2CTxFv_4044_DFG7KPC_JGtMu1Yjfz}
:f6ff930b94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4a1c8afdbale2ed48db20ba8a32|08e0d471482ff70eb120b7c7891124.flag.txt ::@|gohuni1k2023(pppHYKKD_4044_7VKHSAF01x90Wm30yE}
:f6ff930b94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4a1c8afdbale2ed48db20ba8a32|f1.flag.txt ::@|gohuni1k2023(OFwqkcur4_044004H8001_09KcalBt}
:f6ff930b94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4a1c8afdbale2ed48db20ba8a32|f2.flag.txt ::@|gohuni1k2023(Pgi1h2u7_4044_71788glFwkr1FBnPO}
:f930bc94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4a1c8afdbale2ed48db20ba8a32|f3.flag.txt ::@|gohuni1k2023(18e3b9aa84fa932abf1.flag.txt ::@|gohuni1k2023(Ybzkb3bjY_4044_dx0Be5Kk6<cIwsIVw9}
ff930bc94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4a1c8afdbale2ed48db20ba8a32|f4.flag.txt ::@|gohuni1k2023(171fffd7d21|26afea171eb89cc7832cd7d8aa9fe91.flag.txt ::@|gohuni1k2023(xWjusI62_4044_11at72Z0DxAgUFcB13}
ff930bc94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4a1c8afdbale2ed48db20ba8a32|f5.flag.txt ::@|gohuni1k2023(w6dAcEzs_4044_ofkyRudb9x682AICM6}
f6ff930b94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4a1c8afdbale2ed48db20ba8a32|f6.flag.txt ::@|gohuni1k2023(pk2u909C_4044_MxmQnf3GP<7L01vRr}
ff930bc94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4a1c8afdbale2ed48db20ba8a32|f7.flag.txt ::@|gohuni1k2023(13m0epri<4_044RL65ewRTT_1kg0syd}
ff930bc94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4a1c8afdbale2ed48db20ba8a32|f8.flag.txt ::@|gohuni1k2023(jup5gNc_4044_3MCfQnAsa<ULT1ndC}
f6ff930bc94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4a1c8afdbale2ed48db20ba8a32|f9.flag.txt ::@|gohuni1k2023(gzBbtDOr_4044_1u28Gh5Rx<cqeKJXSE}
ff930bc94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4a1c8afdbale2ed48db20ba8a32|f10.flag.txt ::@|gohuni1k2023(xKchNoo_4044_inUu41V6j<lo6yfpcH}
f6ff930bc94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4a1c8afdbale2ed48db20ba8a32|f11.flag.txt ::@|gohuni1k2023(4f7vMyg_4044_wqoqfH22<CU3Jrb04}
ff930bc94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4a1c8afdbale2ed48db20ba8a32|f12.flag.txt ::@|gohuni1k2023(pod5XzfB_4044_FlgLPavH>Gbt2wjt}

7a|11ebab125e519ac1cb8860f1baee9df1\|04be4a1c8afdbale2ed48db20ba8a32|f13.flag.txt ::@|gohuni1k2023(qobTi9v4_4044_727puwRKx_KtBuR0Ve}
:f930bc94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4a1c8afdbale2ed48db20ba8a32|f14.flag.txt ::@|gohuni1k2023(0h6YfMIZ_4044_734bHXBcA<eFb2pMb}
:f6ff930bc94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4a1c8afdbale2ed48db20ba8a32|f15.flag.txt ::@|gohuni1k2023(jk3j9Hg5_4044_V3mpTgGuFx5s4Vhw}
ff930bc94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4a1c8afdbale2ed48db20ba8a32|f16.flag.txt ::@|gohuni1k2023(4f7vMyg_4044_xofxsD5gx<GUzVRjig)

iff930bc94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4a1c8afdbale2ed48db20ba8a32|f17.flag.txt ::@|gohuni1k2023(Pu0t3wAo_4044_D9NrVpgf5<IRCT12NK}
ff930bc94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4a1c8afdbale2ed48db20ba8a32|f18.flag.txt ::@|gohuni1k2023(XZqj1Hqs_4044_6ToKe6wD0c2RGDU6m}
f6ff930bc94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4a1c8afdbale2ed48db20ba8a32|f19.flag.txt ::@|gohuni1k2023(a65cun1c4_044InjpkCfbv_jlj7Xcvx}
ff930bc94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4a1c8afdbale2ed48db20ba8a32|f20.flag.txt ::@|gohuni1k2023(L1Mc4Nu_4044_wmxDb0ch<xByv1ha2}
ff930bc94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4a1c8afdbale2ed48db20ba8a32|f21.flag.txt ::@|gohuni1k2023(1VepbxH2_4044_yhyuSpH<SKhRan3j}
ff930bc94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4a1c8afdbale2ed48db20ba8a32|f22.flag.txt ::@|gohuni1k2023(H1krAn6w_4044_A1sjYQdGycs0Qj1Fwj}
f6ff930bc94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4a1c8afdbale2ed48db20ba8a32|f23.flag.txt ::@|gohuni1k2023(8Bgpk07n_4044_Age19d24<7V7D02da}
f6ff930bc94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4a1c8afdbale2ed48db20ba8a32|f24.flag.txt ::@|gohuni1k2023(TfPwfoH_4044_qMlxM7PcPBymVMP}
f26811f6ff930bc94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4a1c8afdbale2ed48db20ba8a32|f25.flag.txt ::@|gohuni1k2023(GoneJPK<4_044uehnHuHTD_8R0KzXhW}
e26811f6ff930bc94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4a1c8afdbale2ed48db20ba8a32|f26.flag.txt ::@|gohuni1k2023(jxIgwg9_4044_z00sycklx<VdqUFbOp}
f6ff930bc94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4a1c8afdbale2ed48db20ba8a32|f27.flag.txt ::@|gohuni1k2023(QdkB170_4044_2an0aj7hkux317uzj}
f6ff930bc94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4a1c8afdbale2ed48db20ba8a32|f28.flag.txt ::@|gohuni1k2023(8Ck97z31_4044_Vw1VDPBL<Q8052048}
f6ff930bc94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4a1c8afdbale2ed48db20ba8a32|f29.flag.txt ::@|gohuni1k2023(TEk1zfl1_4044_07aeq1Vs<gy37zaug}
f6ff930bc94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4a1c8afdbale2ed48db20ba8a32|f30.flag.txt ::@|gohuni1k2023(w2CQUBa_4044_xg7neF<P5RmZneQl}
f6ff930bc94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4a1c8afdbale2ed48db20ba8a32|f31.flag.txt ::@|gohuni1k2023(QRMrWg4<4_04416svBrzv_CUUmhs3G}
.f6ff930bc94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4b135967d3cc05d034f8fc349|15916f58531b2a723ec467817acf1511.flag.txt ::@|gohuni1k2023(jzsAqt3_4044_NCnvBtBv5<ElxHqlzX}
.f6ff930bc94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4b135967d3cc05d034f8fc349|18e3b9aa84fa932abf1.flag.txt ::@|gohuni1k2023(ZYlmuvBK_4044_CE3XGblzacs9K31sz}
.f6ff930bc94feefb34687a|11ebab125e519ac1cb8860f1baee9df1\|04be4b135967d3cc05d034f8fc349|26afea171eb89cc7832cd7d8aa9fe91.flag.txt ::@|gohuni1k2023(kmAdQwhz_4044_5HM7ljueV<awhNmjwE}

```

You can view another 4 .rar files too and you will notice that there are no flag in the 4<sup>th</sup> and 5<sup>th</sup> rar. Thus, things get easier.

Copy and paste all 3 viewing output into a new .txt file.

outputt - Notepad

```

File Edit Format View Help
\{6d5584016aa694b5b224fb8b46203ab7\}15916f58531b2a723ec467817acf1511\flag.txt ??@gohunikl2023{hNbU0G45..4044_rKh5EUjBe<xZ11jdvr}
\{6d5584016aa694b5b224fb8b46203ab7\}18e3b9aa84af93a2b1b56c2d0c9e5eb1\flag.txt ??@gohunikl2023{gQnDBqfh..4044_hDv4g1s1KNYDkw4mi}
\{6d5584016aa6a694b5b224fb8b46203ab7\}26afea171eb89c7832cd7d8a3f9ea91\flag.txt ??@gohunikl2023{5L1v708j..4044_6fk8vYcaV<ntaHgbia}

\{6d5584016aa694b5b224fb8b46203ab7\}385cf2dd9e84383b2e0179b98c9b3a8\flag.txt ??@gohunikl2023{gJzOLf7e..4044_xuZoCa1bo<Phld5rwA}
\{6d5584016aa694b5b224fb8b46203ab7\}98e0a5571482f70e0b120b77b891124\flag.txt ??@gohunikl2023{guUSBsc..4044_eh2vojrP<id221clM1}
\{6d5584016aa694b5b224fb8b46203ab7\}15916f58531b2a723ec467817acf1511\flag.txt ??@gohunikl2023{19Hz0e0w54..044HmAcf1ly.svh91V0q}
\{8cf8595899746856668a485822b26a96\}18e3b9aa84af93a2b1b56c2d0c9e5eb1\flag.txt ??@gohunikl2023{LMH5831..4044_zGCJuowXp5<syroboGM}
\{8cf8595899746856668a485822b26a96\}26afea171eb89c7832cd7d8a3f9ea91\flag.txt ??@gohunikl2023{suqC3k8..4044_rKhZvms0<DPfLAsjtV}
\{8cf8595899746856668a485822b26a96\}70e120b77b891124\flag.txt ??@gohunikl2023{K06pkInd..4044_AoVRbqNt<xLfaArb}
\{8cf8595899746856668a485822b26a96\}98e0a5571482ff70e120b77b891124\flag.txt ??@gohunikl2023{aos1nlw..4044_qz25bts6<cB6w25t}
\{8cf8595899746856668a485822b26a96\}46856668a485822b26a96\98e0a5571482ff70e120b77b891124\flag.txt ??@gohunikl2023{BB1L00X01..4044_qTxBzBXV<uj8MgnSk}
\{8cf8595899746856668a485822b26a96\}flag.txt ??@gohunikl2023{grkPAgGu..044HdsK4Pn2l.OPrEwneu}
\{928022ecbf4103e09ad943425870552\}26afea171eb89c7832cd7d8a3f9ea91\flag.txt ??@gohunikl2023{zHMeZxuc..4044_dHfCmrguR<bAv21U83}
\{928022ecbf4103e09ad943425870552\}18e3b9aa84af93a2b1b56c2d0c9e5eb1\flag.txt ??@gohunikl2023{KsA5jBz..4044_tlnbApDBD<AjyzmnPp}
\{928022ecbf4103e09ad943425870552\}385cf2dd9e84383b2e70179b98c9b3a8\flag.txt ??@gohunikl2023{2w41NxA..4044_EscCtHia<KNTf1AxH}
\{928022ecbf4103e09ad943425870552\}98e0a5571482ff70e120b77b891124\flag.txt ??@gohunikl2023{zRBtzx2..4044_MDNgUTDBx<rcc8DLRpD}
\{928022ecbf4103e09ad943425870552\}flag.txt ??@gohunikl2023{041dq1x54..044hdX16S0b.EPu4SD0}
94b4eb135967d3ccc065d034f8fcf349..15916f58531b2a723ec467817acf1511\flag.txt ??@Zgohunikl2023{NgRI8mt..4044_Bm7EV7cul<fdwIUmSR}
\{94b4eb135967d3ccc065d034f8fcf349..26afea171eb89c7832cd7d8a3f9ea91\flag.txt ??@gohunikl2023{AhePsAq..4044_93IGlnKrnj<9HNRMThX}
\{94b4eb135967d3ccc065d034f8fcf349..26afea171eb89c7832cd7d8a3f9ea91\flag.txt ??@gohunikl2023{1756hKX1..4044_yddqZUMGTr<Vhxuoop}
\{94b4eb135967d3ccc065d034f8fcf349..26afea171eb89c7832cd7d8a3f9ea91\flag.txt ??@gohunikl2023{OlaU9qd..4044_nKqSfq1h..f1NDD_m3333??}
\{94b7fb6a..338b248c8aa81fc2e8767023\}15916f58531b2a723ec467817acf1511\flag.txt ??@gohunikl2023{044K6fXsZm0..HettWGr}
\{94b7fb6a..338b248c8aa81fc2e8767023\}18e3b9aa84af93a2b1b56c2d0c9e5eb1\flag.txt ??@gohunikl2023{0NzTxeEq..4044_q11MuDe<3NReRecpt}
\{94b7fb6a..338b248c8aa81fc2e8767023\}26afea171eb89c7832cd7d8a3f9ea91\flag.txt ??@gohunikl2023{162303wY..4044_VCzaas1ln<AzRiopo}
\{94b7fb6a..338b248c8aa81fc2e8767023\}385cf2dd9e84383b2e70179b98c9b3a8\flag.txt ??@gohunikl2023{gtKzGxHs..4044_2VYH0MdW<OyRjPs2v}
\{94b7fb6a..338b248c8aa81fc2e8767023\}98e0a5571482ff70e120b77b891124\flag.txt ??@gohunikl2023{D99170g..4044_IgamInom5<vw2IPVTX}
\{94b7fb6a..338b248c8aa81fc2e8767023\}flag.txt ??@gohunikl2023{teRmEpWk<..04487KhnAb0d7.q2PzsUNF}
\{a86091ce213173e28581f419c30674\}18e3b9aa84af93a2b1b56c2d0c9e5eb1\flag.txt ??@gohunikl2023{Lzz83z4k..4044_vS7a8uXQD<z3AUQnWv}
36091ce213173e28581f419c30674\18e3b9aa84af93a2b1b56c2d0c9e5eb1\flag.txt ??@tgohunikl2023{Le4K1Gp..4044_PLF0YHdu<dh2DpSRC}
\{a86091ce213173e28581f419c30674\}26afea171eb89c7832cd7d8a3f9ea91\flag.txt ??@nghohunikl2023{zdMMETY..4044_e3G065ZhmXhzscQwl}
\{a86091ce213173e28581f419c30674\}98e0a5571482ff70e120b77b891124\flag.txt ??@gohunikl2023{fU0h1OpS..4044_Q3EC2Bh61t<c11gTPE}
\{a86091ce213173e28581f419c30674\}98e0a5571482ff70e120b77b891124\flag.txt ??@gohunikl2023{S0TnZ33v..4044_antdsXbj<sD35I05rZ}
\{a86091ce213173e28581f419c30674\}flag.txt ??@gohunikl2023{RCu5a1rq4..0440RGacCcN.Udrf1Fu6}
\{f81dea7db309fbcc8b1b51914a4afbf\}15916f58531b2a723ec467817acf1511\flag.txt ??@gohunikl2023{bYqlzsrc..4044_ki03Tnifd<HpkZuFr}
\{f81dea7db309fbcc8b1b51914a4afbf\}18e3b9aa84af93a2b1b56c2d0c9e5eb1\flag.txt ??@
```

Ln 53234, Col 223    100%    Windows (CRLF)    UTF-8

There are 2 ways to find the flag:

- 1) Find unique symbols such as '????', '!' and try your luck.
- 2) Bring this to the Kali Linux, use the grep command to filter out 4044,404,40,4\_ (observed the flag pattern) by using the command: grep 'gohunikl2023' output.txt | grep -Ev '4044|404|40|4\_' (recommended)

Reference Link: <https://osxdaily.com/2018/04/05/how-exclude-word-grep-command-line/>

The flag is gohunikl2023{h0oo000w\_d1dcUuuu\_f1NDD\_m3333??}

## STEGANOGRAPHY

### Cring (50 Points) Credit to: Danish

Given CringCring.wav file with short old phone sounds (12 seconds of dialling sounds)

Every audio that relates to telephone, we can use the DTMF Decoder tool.

The screenshot shows a web application titled "DTMF Decoder". At the top, there is a file input field labeled "Choose File" containing "CringCring.wav", a sensitivity threshold input field set to "0.050", and a "Decode" button. Below the input fields, the word "Output" is displayed in bold. Under "Output", a list of 12 DTMF digit sequences is shown, each preceded by "000s":  
0001s 666..444..999..222..777..  
0002s 333..111..999..555..999..  
0003s 888..222..666..888..555..  
0004s 666..222..666..666..444..  
0005s 222..555..999..111..333..  
0006s 999..999..777..111..111..  
0007s 111..444..222..111..222..  
0008s 333..555..555..888..222..  
0009s 222..111..555..111..666..  
0010s 999..999..000..333..555..  
0011s 222..999..555..777..777..  
0012s 999..444..111..777..333..  
At the bottom of the output section, the text "Decoded: 649273195982685626642591399711142123558221516990352957794173" is displayed.

Decoded: 649273195982685626642591399711142123558221516990352957794173

We brought the decoded output of numbers to Cipher Identifier Boxentriq.

The screenshot shows the "boxentriq.com/code-breaking/cipher-identifier" page. At the top, there is a text input field labeled "Enter Ciphertext here" containing the decoded text "649273195982685626642591399711142123558221516990352957794173". Below the input field are buttons for "Analyze Text", "Copy", "Paste", and "Text Options...". A note below the input field states: "Note: To get accurate results, your ciphertext should be at least 25 characters long." In the "Analysis Results" section, it says "Your ciphertext is likely of this type: **Decimal Code (click to read more)**".

In order to decode the decimal code, you must convert it to hex first. Then, convert from hex to text (ascii).

The screenshot shows a web browser window with the URL [binaryhexconverter.com/decimal-to-hex-converter](https://binaryhexconverter.com/decimal-to-hex-converter). The page title is "Decimal to Hexadecimal Converter". A text input field contains the decimal value "64927319598268562664". A green "Convert" button is next to it. To the right, the converted hex value "676F68756E696B6C323032337B4372" is displayed in a text input field. Below the input fields is a link "swap conversion: [Hex to Decimal](#)".

Decimal Value (max: 9223372036854775807) Hexadecimal Value  
64927319598268562664 Convert 676F68756E696B6C323032337B4372  
swap conversion: [Hex to Decimal](#)

Decimal to hex conversion result in base numbers  
(64927319598268562664259139971114212355822151699035295779  
= (676F68756E696B6C323032337B4372316E675F4372316E677D)<sub>16</sub>

## Decimal System

The decimal numeral system is the most commonly used and the standard system in daily

The screenshot shows a web browser window with the URL [rapidtables.com/convert/number/hex-to-ascii.html](https://www.rapidtables.com/convert/number/hex-to-ascii.html). The page title is "Hex to ASCII Text String Converter". It instructs users to enter hex bytes with any prefix / postfix / delimiter and press the "Convert" button. An example is given: "(e.g. 45 78 61 6d 70 6C 65 21)".

From To  
Hexadecimal Text  
Open File

Paste hex numbers or drop file  
676F68756E696B6C323032337B4372316E675F4372316E677D

Character encoding  
ASCII

Convert Reset Swap  
gohunikl2023{Cr1ng\_Cr1ng}

The flag is gohunikl2023{Cr1ng\_Cr1ng}

# REVERSE ENGINEERING

## Reverse Script (50 Points) Credit to: Haziq

Given flag.txt.enc and a Python script reverse.py:

```
import sys

a = "!\#$%&()'*+,-./0123456789;:<=>?@ABCDEFGHIJKLMNPQRSTUVWXYZ" + \
    "[\\]^_`abcdefghijklmnopqrstuvwxyz{|}~"

def arg133(arg432):
    if arg432 == a[38] + a[78] + a[67] + a[89] + a[72] + a[43] + a[75] + a[31] + a[53] + a[50] + a[42] + a[16] + a[77] + a[70] +
        a[38] + a[71] + a[16] + a[67] + a[15] + a[81] + a[19] + a[39]:
            return True
    else:
        print(a[51] + a[71] + a[64] + a[83] + a[94] + a[79] + a[64] + a[82] + a[82] + a[86] + a[78] +
              a[81] + a[67] + a[94] + a[72] + a[82] + a[94] + a[72] + a[77] + a[66] + a[78] + a[81] +
              a[81] + a[68] + a[66] + a[83])
        sys.exit(0)
    return False

def arg111(arg444):
    return arg122(arg444.decode(), a[62])

def arg232():
    arg282 = input(a[47] + a[75] + a[68] + a[64] + a[82] + a[68] + a[94] + a[68] + a[77] + a[83] +
                   a[68] + a[81] + a[94] + a[66] + a[78] + a[81] + a[81] + a[68] + a[66] + a[83] +
                   a[94] + a[16] + a[82] + a[83] + a[94] + a[79] + a[64] + a[82] + a[82] + a[86] +
                   a[78] + a[81] + a[67] + a[94] + a[69] + a[78] + a[81] + a[94] + a[69] + a[75] +
                   a[64] + a[70] + a[25] + a[94])

    if arg282 == "":
        print("Password 1 is empty")
        sys.exit(0)

    arg383 = input(a[47] + a[75] + a[68] + a[64] + a[82] + a[68] + a[94] + a[68] + a[77] + a[83] +
                   a[68] + a[81] + a[94] + a[66] + a[78] + a[81] + a[81] + a[68] + a[66] + a[83] +
                   a[94] + a[17] + a[82] + a[83] + a[94] + a[79] + a[64] + a[82] + a[82] + a[86] +
                   a[78] + a[81] + a[67] + a[94] + a[69] + a[78] + a[81] + a[94] + a[69] + a[75] +
                   a[64] + a[70] + a[25] + a[94])

    if arg383 == "":
        print("Password 2 is empty")
        sys.exit(0)

    if arg282 is not None and arg383 is not None:
        arg888 = arg282 + arg383
        return arg888

def arg132():
    return open('flag.txt.enc', 'rb').read()

def arg112():
    print(a[54] + a[68] + a[75] + a[66] + a[78] + a[76] + a[68] + a[94] + a[65] + a[64] + a[66] +
          a[74] + a[13] + a[13] + a[13] + a[94] + a[88] + a[78] + a[84] + a[81] + a[94] + a[69] +
          a[75] + a[64] + a[70] + a[11] + a[94] + a[84] + a[82] + a[68] + a[81] + a[25])
```

```

def arg122(arg432, arg423):
    arg433 = arg423
    i = 0
    while len(arg433) < len(arg432):
        arg433 = arg433 + arg423[i]
        i = (i + 1) % len(arg423)
    return "".join([chr(ord(arg422) ^ ord(arg442)) for (arg422, arg442) in zip(arg432, arg433)])\n\n
arg444 = arg132()
arg432 = arg232()
arg133(arg432)
arg112()
arg423 = arg111(arg444)
print(arg423)
sys.exit(0)

```

Thonny - D:\Documents\CTF FILES\20.11.23 GOH Unikl 2023\REVERSE ENGINEERING\reverse.py @ 19 : 1

File Edit View Run Tools Help

asal.py reverse.py Assistant

```

1 import sys
2
3 a = "!\"#$%&'()*+,-./0123456789:;=>?@ABCDEFGHIJ
4     "[\\]^_`abcdefghijklmnopqrstuvwxyz{|}~ "
5
6 def arg133(arg432):
7     if arg432 == a[38] + a[78] + a[67] + a[89]:
8         return True
9     else:
10        print(a[51] + a[71] + a[64] + a[83] +
11              a[81] + a[67] + a[94] + a[72] +
12              a[81] + a[68] + a[66] + a[83])
13        sys.exit(0)
14    return False
15
16
17 def arg111(arg444):
18     return arg122(arg444.decode(), a[62])
19
20
21 def arg232():
22     arg282 = input(a[47] + a[75] + a[68] + a[6

```

< >

Shell

```

>>> %Run reverse.py
Please enter correct 1st password for flag: fsf
Please enter correct 2st password for flag: w3rw
That password is incorrect

```

Local Python 3 • Thonny's Python

First, execute the script to analyze the code. After a session with Blackbox AI, it said:

The program takes three arguments and processes them to return the correct decryption key. It utilizes XOR operations and repeating sequences of characters.

The function arg232() takes the first argument. If it's empty, the program terminates.

The function arg332() takes the second argument. If it's empty, the program terminates.

The function arg444() takes the third argument.

Solution:

```
def arg133(arg432):
    if arg432 == a[38] + a[78] + a[67] + a[89] + a[72] + a[43] + a[75] +
a[31] + a[53] + a[50] + a[42] + a[16] + a[77] + a[70] + a[38] + a[71] +
a[16] + a[67] + a[15] + a[81] + a[19] + a[39]:
        return True
    else:
        print(a[51] + a[71] + a[64] + a[83] + a[94] + a[79] + a[64] +
a[82] + a[82] + a[86] + a[78] +
a[81] + a[67] + a[94] + a[72] + a[82] + a[94] + a[72] +
a[77] + a[66] + a[78] + a[81] +
a[81] + a[68] + a[66] + a[83])
        #sys.exit(0) //Line 13 should be removed
    return False
```

Line 13 'sys.exit(0)' should be removed to prevent process exit.

The screenshot shows the Thonny Python IDE interface. The top bar displays 'Thonny - D:\Documents\CTF FILES\20.11.23 GOH Unikl 2023\REVERSE ENGINEERING\reverse.py @ 15:1'. The main window shows the code editor with the following content:

```
1 import sys
2
3 a = "!\\"#$%&'()*+-./0123456789:;=>?@ABCDEFGHIJKLMNPQRSTUVWXYZ{|}~ "
4
5 def arg133(arg432):
6     if arg432 == a[38] + a[78] + a[67] + a[89]:
7         return True
8     else:
9         print(a[51] + a[71] + a[64] + a[83] +
10            a[81] + a[67] + a[94] + a[72] +
11            a[81] + a[68] + a[66] + a[83])
12         #sys.exit(0) //Line 13 should be removed
13     return False
14
15
16
17 def arg111(arg444):
18     return arg122(arg444.decode(), a[62])
19
20
21 def arg232():
22     arg282 = input(a[47] + a[75] + a[68] + a[6]
```

The bottom shell window shows the execution of the script:

```
>>> %Run reverse.py
Please enter correct 1st password for flag: tsse
Please enter correct 2st password for flag: fs
That password is incorrect
Welcome back... your flag, user:
gohunikl2023{m0n4rch.s3cr37}
```

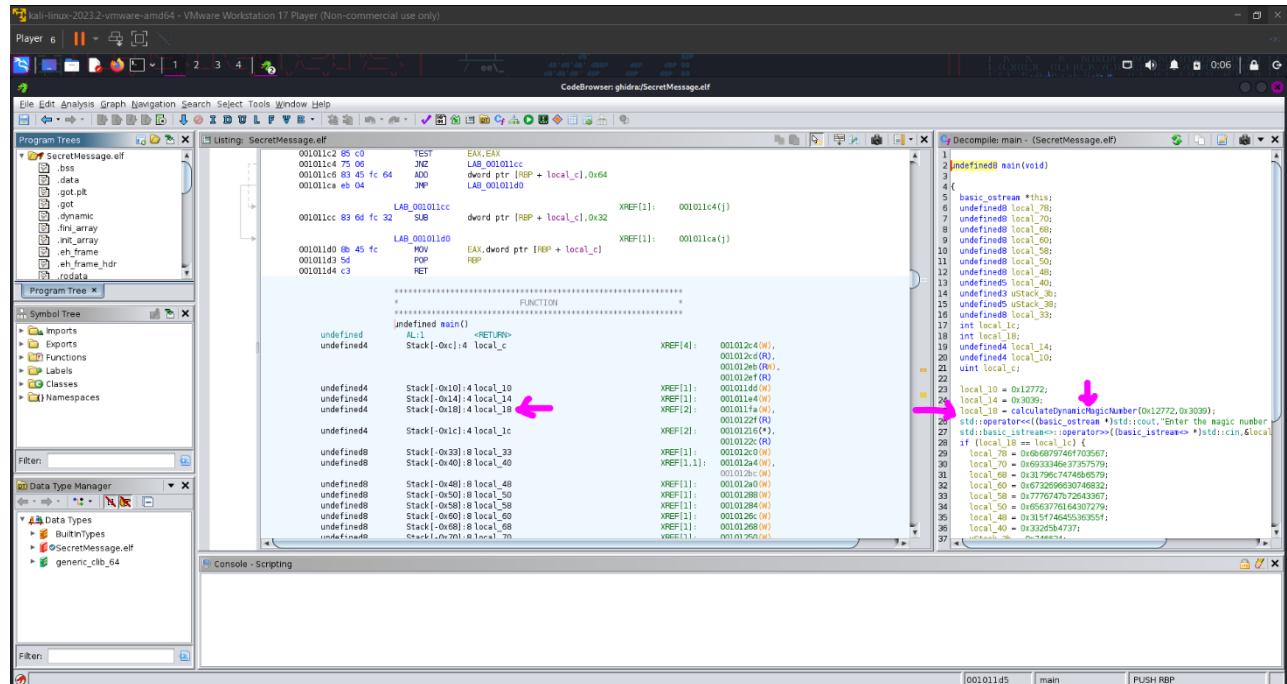
The flag is gohunikl2023{m0n4rch.s3cr37}

## Magic Number (100 Points) Credit to: Haziq

Given an ELF file SecretMessage.elf

For ELF file, we can practice ghidra in Kali Linux with reference link:

[https://youtu.be/oTD\\_ki86c9I?si=9OeVSQrkTOKiFDsC&t=450](https://youtu.be/oTD_ki86c9I?si=9OeVSQrkTOKiFDsC&t=450)



0x12772 to decimal

0x3039 to decimal

About 4 results (0.21 seconds)

0x12772 = 75634

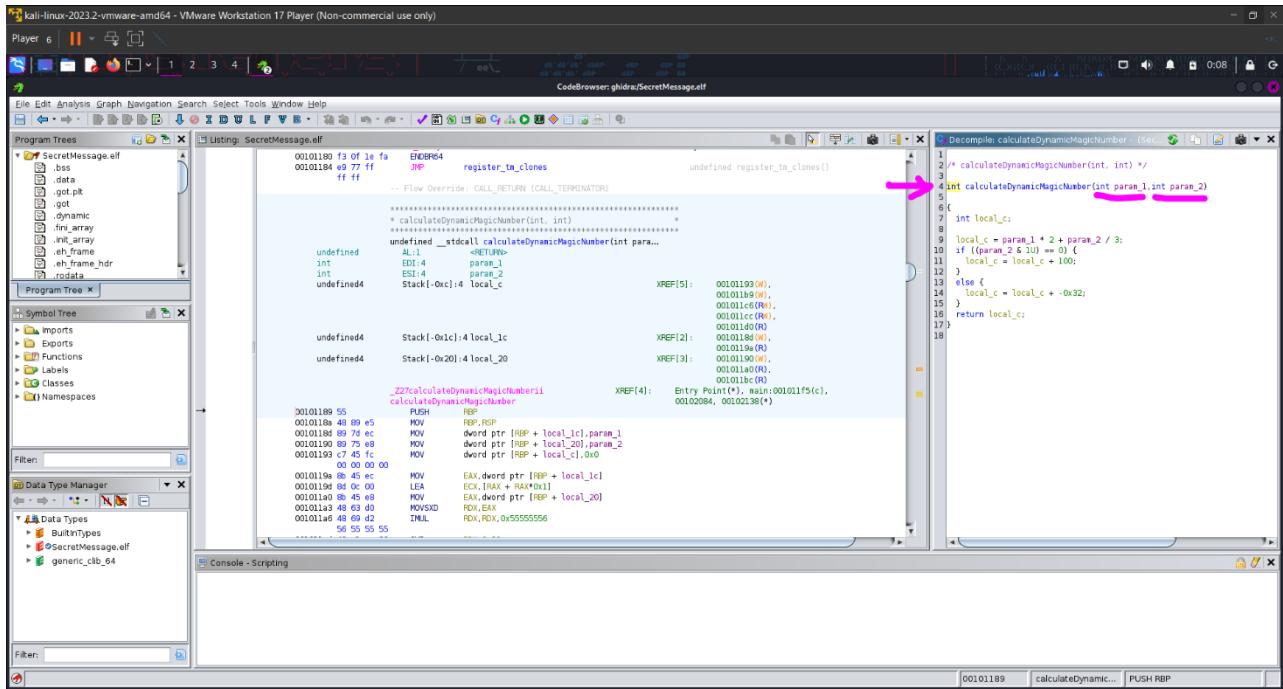
|     |     |                |   |   |   |    |
|-----|-----|----------------|---|---|---|----|
| Rad | Deg | x <sup>y</sup> | ( | ) | % | AC |
| Inv | sin | ln             | 7 | 8 | 9 | ÷  |
| π   | cos | log            | 4 | 5 | 6 | ×  |
| e   | tan | √              | 1 | 2 | 3 | -  |
| Ans | EXP | x <sup>y</sup> | 0 | . | = | +  |

About 1,190 results (0.31 seconds)

0x3039 = 12345

|     |     |                |   |   |   |    |
|-----|-----|----------------|---|---|---|----|
| Rad | Deg | x <sup>y</sup> | ( | ) | % | AC |
| Inv | sin | ln             | 7 | 8 | 9 | ÷  |
| π   | cos | log            | 4 | 5 | 6 | ×  |
| e   | tan | √              | 1 | 2 | 3 | -  |
| Ans | EXP | x <sup>y</sup> | 0 | . | = | +  |

As you can see, there is a function local\_18 = calculateDynamicMagicNumber(0x12772,0x3039); with converted value parameter 75634 and 12345 in decimal respectively. Double click on the calculateDynamicMagicNumber function.



We can observe the calculation source code for param\_1 and param\_2.  
So we can calculate the 'magic number' with the parameter given.

```

1
2 /* calculateDynamicMagicNumber(int, int) */
3
4 int calculateDynamicMagicNumber(int param_1,int param_2)
5
6{
7     int local_c;
8
9     local_c = param_1 * 2 + param_2 / 3;
10    if ((param_2 & 1U) == 0) {
11        local_c = local_c + 100;
12    }
13    else {
14        local_c = local_c + -0x32;
15    }
16    return local_c;
17}
18

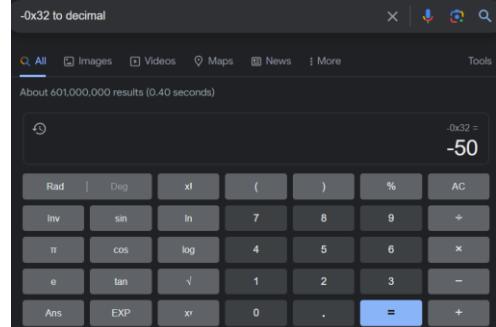
```

param\_1 = 75634, param\_2 = 12345

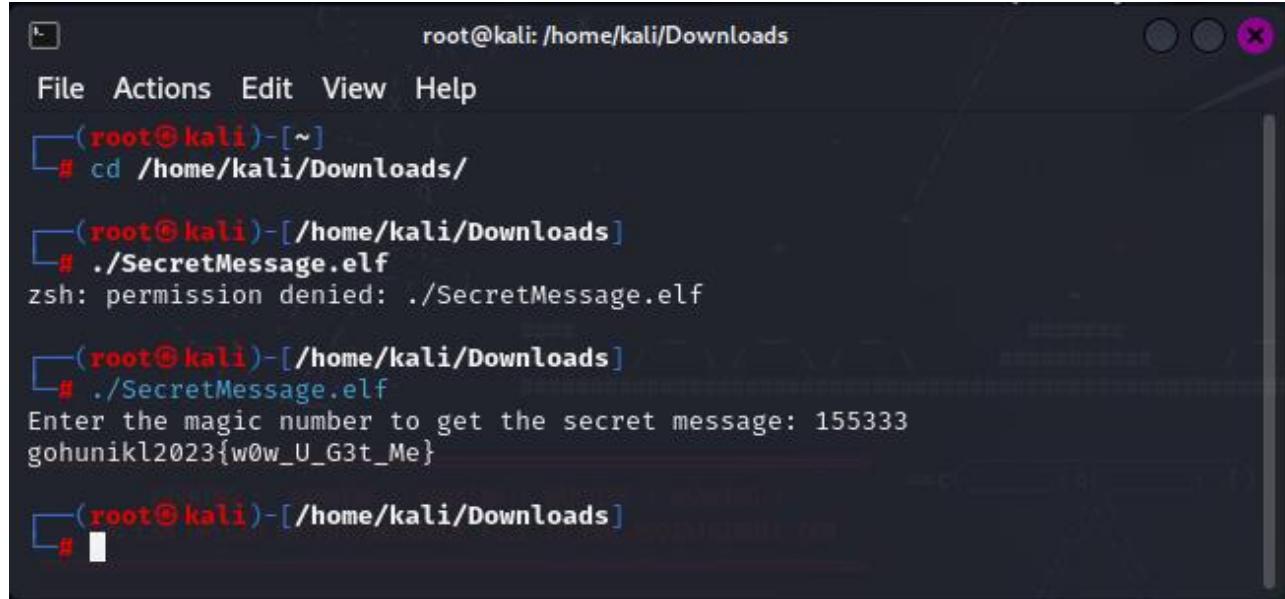
$$75634 * 2 = 151268, 12345 / 5 = 4115 \\ 151268 + 4115 = 155383$$

So value local\_c is 155,383.

Next, the if statement checks if 12345 is even or odd. Since 12345 is odd, the process continue to the else statement. The local\_c value will minus 50 (155383 – 50). And BOOMM! That's the magic number.. 155333



Now, execute the .elf file with the input of the magic number 155333 and the flag will be printed out.



A terminal window titled "root@kali: /home/kali/Downloads". The session starts with the root user navigating to the Downloads directory. It then attempts to run the "SecretMessage.elf" file, which is denied due to permission. After entering the magic number 155333, the program outputs the flag "gohunikl2023{w0w\_U\_G3t\_Me}".

```
root@kali: /home/kali/Downloads
File Actions Edit View Help
[root@kali ~]
# cd /home/kali/Downloads/
[root@kali /home/kali/Downloads]
# ./SecretMessage.elf
zsh: permission denied: ./SecretMessage.elf
[root@kali /home/kali/Downloads]
# ./SecretMessage.elf
Enter the magic number to get the secret message: 155333
gohunikl2023{w0w_U_G3t_Me}
[root@kali /home/kali/Downloads]
```

The flag is gohunikl2023{w0w\_U\_G3t\_Me}

### Reverse It (150 Points) Credit to: Haziq

Given the description of the challenge: 杯櫻漣毒(日)社筈 | 數剥癥姆叫杼

And a python code:

```
my_string = "My_Strings"
my_result = ""

for i in range(0, len(my_string), 2):
    my_first_letter = ord(my_string[i]) << 8
    my_second_letter = ord(my_string[i + 1])
    my_final = chr(my_first_letter + my_second_letter)
    my_result += my_final

print(my_result)
```

It seems that the Python works as ascii encoder to (like Chinese) code..

Reverse it by creating a new Python code to decode this given clue 杯櫻漣毒(日)社筈 | 數剥癥姆叫杼 ..... a little bit hardwork of coding this is the python code to decode.

```
chinese = "杯櫻漣毒(日)社筈 | 數剥癥姆叫杼"

flag_lol = ""

for i in range(0, len(chinese)):

    combine = ord(chinese[i])
```

```

first_letter = chr((combine >> 8) & 0xFF)

second_letter = chr(combine & 0xFF)

flag_lol += first_letter + second_letter

print(flag_lol)

```

The screenshot shows the Thonny Python IDE interface. The main window displays the following Python script:

```

1 chinese = "杯櫻漬苺(日)社筍 | 敷剥癥姆叫杆"
2
3 flag_lol = ""
4
5 for i in range(0, len(chinese)):
6
7     combine = ord(chinese[i])
8
9
10    first_letter = chr((combine >> 8) & 0xFF)
11
12    second_letter = chr(combine & 0xFF)
13
14
15    flag_lol += first_letter + second_letter
16
17 print(flag_lol)

```

The code uses the `ord` function to get the ASCII value of each character in the string `chinese`. It then separates each character into its high byte (shifted right by 8 bits and ANDed with 0xFF) and low byte (ANDed with 0xFF). These two bytes are concatenated to form the `flag_lol` string.

In the bottom right corner of the code editor, there is an **Assistant** panel with the following message:

The code in [testttt.py](#) looks good.  
If it is not working as it should, then consider using some general [debugging techniques](#).

[Was it helpful or confusing?](#)

Below the code editor, the **Shell** tab shows the output of the script:

```

gohunikl2023{N1ce_Revers1ng}
>>>

```

The flag is gohunikl2023{N1ce\_Revers1ng}