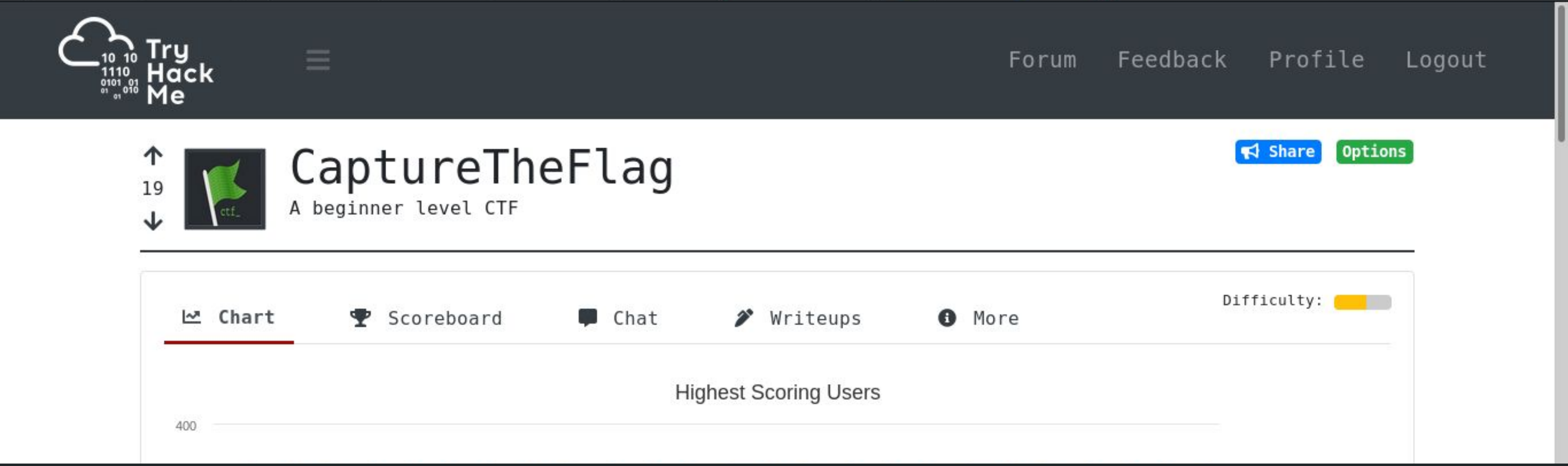
Forensics and Codes

CTF: CaptureTheFlag

https://www.tryhackme.com/room/functf

No SSH Connection Required



CaptureTheFlag

A beginner level CTF

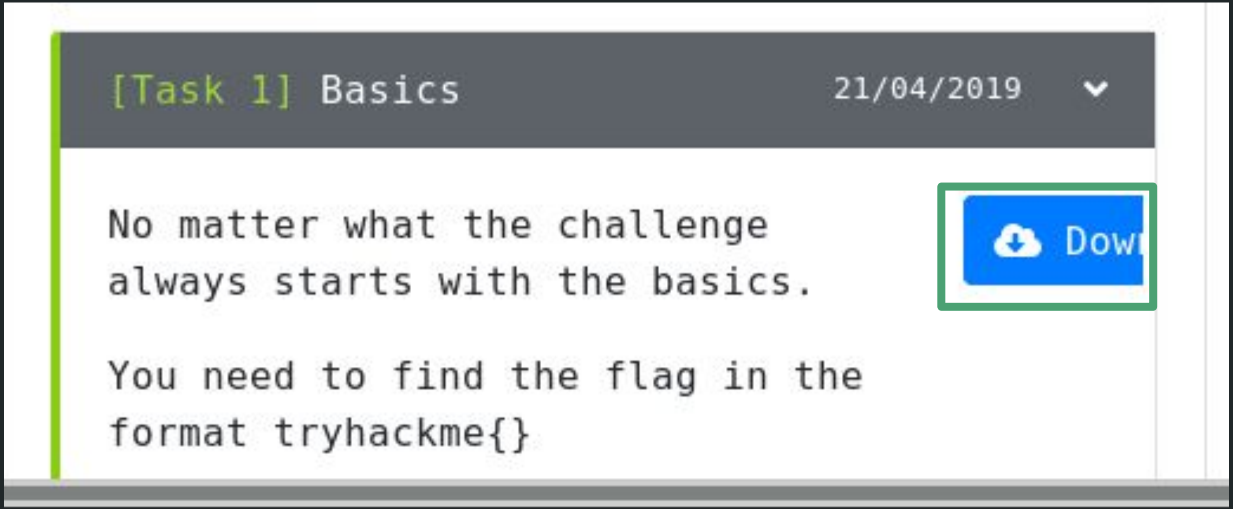
Tools you need:

- stegosuite
- Img_cat
- strings

- For some part of the challenge is choosing the right tool

How to play:

Download the image and read the hint



[Task 1] Basics 21/04/2019

No matter what the challenge always starts with the basics.

You need to find the flag in the format tryhackme{}

[Part 1] #1 Do Images have strings?

#1 Do Images have strings?

The hint here is strings

strings - print the sequences of printable characters in files

```
root@kali:~/Downloads# strings Basic.jpg

JFIF
ICC_PROFILE
...

tryhackme{7h1s_i5_wh4t_strings_d0es}  ⇐ ANSWER TO #1
```

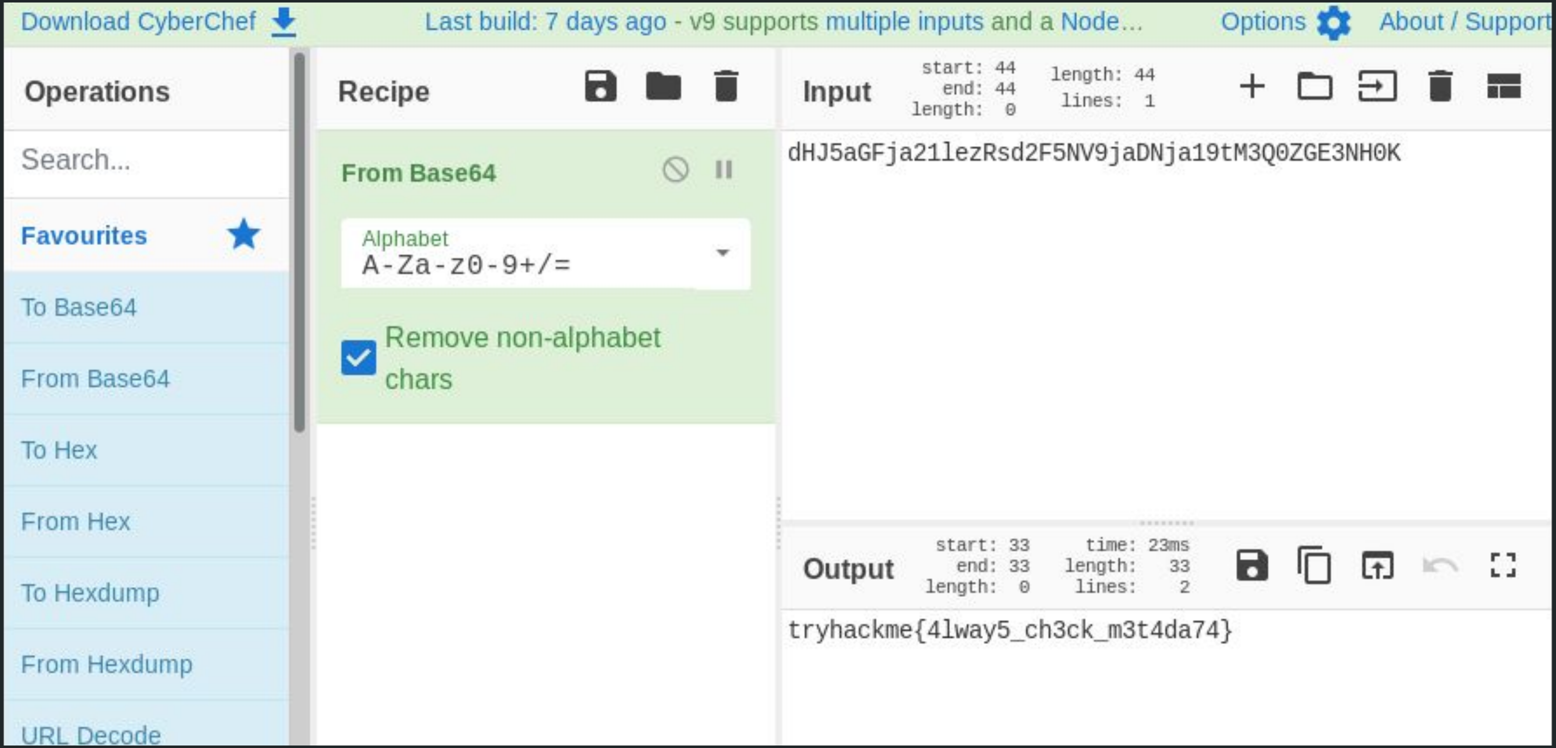
[Part 1] #2 Metadata or EXIF data?.....ah!! I'm so confused

#2 Metadata or EXIF data?.....ah!! I'm so confused

➜ Metadata or Exif data can be viewed with exiftool

```
root@kali:~/Downloads# exiftool Basic.jpg

...

Comment                    : dHJ5aGFja21lezRsd2F5NV9jaDNja19tM3Q0ZGE3NH0K

Image Width                : 404

Image Height                    : 404

Encoding Process                : Progressive DCT, Huffman coding
```

#2 Metadata or EXIF data?.....ah!! I'm so confused



From Base64

Alphabet
A-Za-z0-9+/=

☑ Remove non-alphabet chars

```
start: 44    length: 44
  end: 44     lines:  1
length:  0
```

```
dHJ5aGFja21lezRsd2F5NV9jaDNja19tM3Q0ZGE3NH0K
```

```
start: 33      time: 23ms
  end: 33    length:   33
length:  0     lines:    2
```

```
tryhackme{4lway5_ch3ck_m3t4da74}
```

[Part 2] #1 Find the flag.

#1 Find the flag.

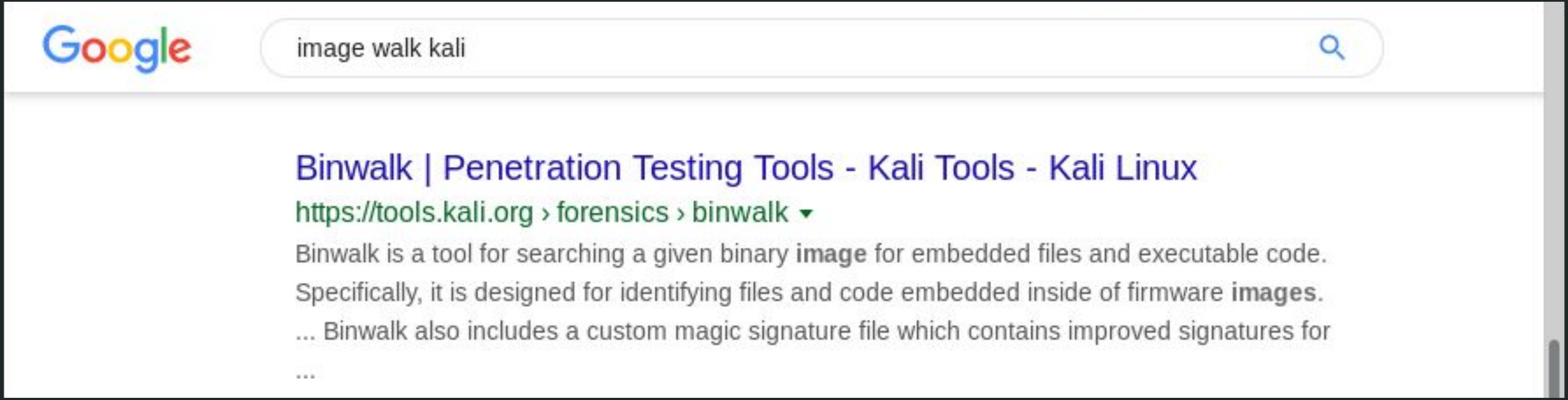
Download the next image: walk.jpg

I can make this easy just by telling you the tool or maybe you can read the title again and figure out your self.

P.S - It's a very famous, open source tool :)

#1 Find the flag.

The image name is walk, so:



Binwalk | Penetration Testing Tools - Kali Tools - Kali Linux

https://tools.kali.org › forensics › binwalk

Binwalk is a tool for searching a given binary **image** for embedded files and executable code. Specifically, it is designed for identifying files and code embedded inside of firmware **images**. ... Binwalk also includes a custom magic signature file which contains improved signatures for ...

The tool is binwalk

#1 Find the flag.

binwalk:

binwalk - tool for searching binary images for embedded files and executable code

Param:

-e, --extract   Automatically extract known file types

#1 Find the flag.

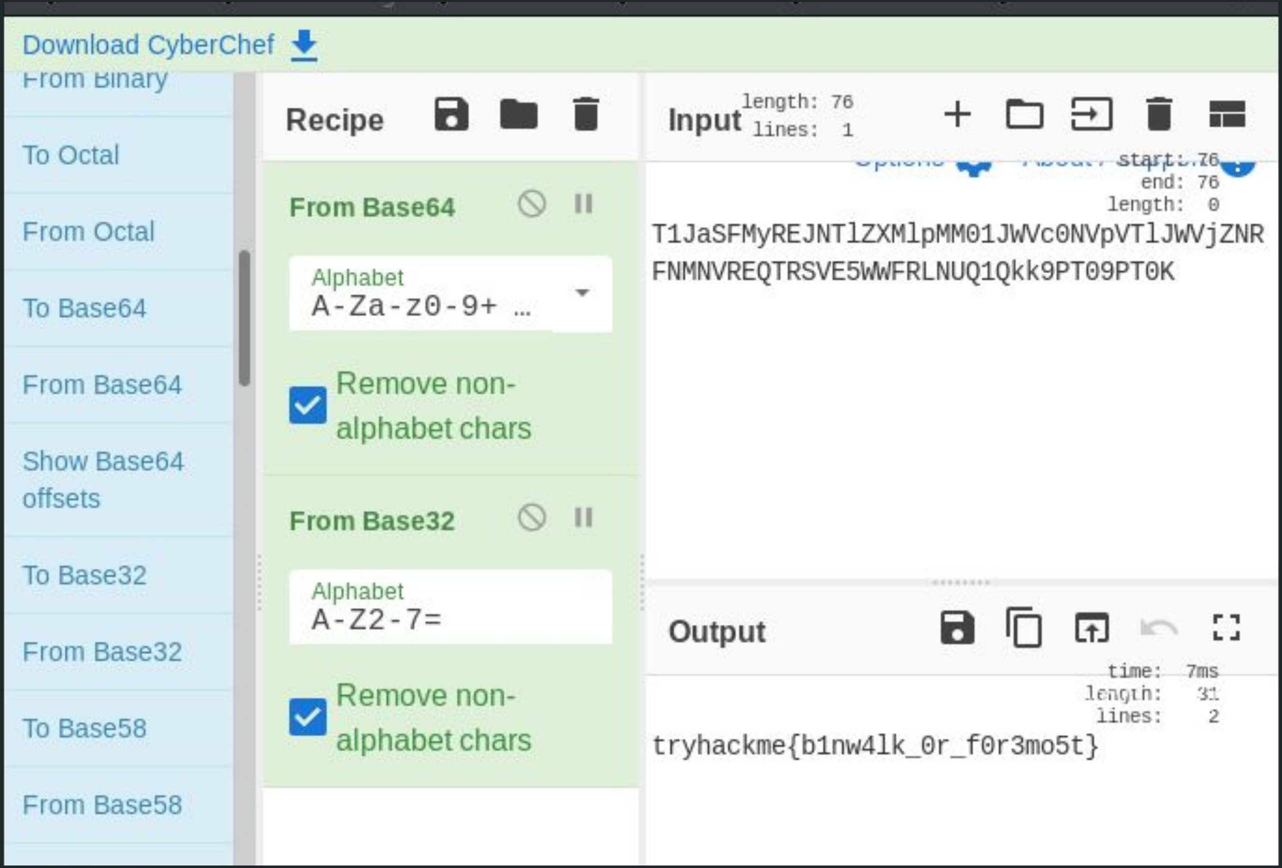
```
root@kali:~/Downloads# binwalk -e walk.jpg

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
0             0x0             JPEG image data, JFIF standard 1.01
30            0x1E            TIFF image data, big-endian, offset of first image directory: 8
170610            0x29A72       gzip compressed data, from Unix, last modified:
2019-04-21 08:25:56
root@kali:~/Downloads#ls

 _walk.jpg.extracted
```

#1 Find the flag.

```
root@kali:~/Downloads# cd _walk.jpg.extracted/
root@kali:~/Downloads/_walk.jpg.extracted# ls
29A72  29A72.gz
root@kali:~/Downloads/_walk.jpg.extracted# cat 29A72
PaxHeader/flag.txt000644 001750 001751 00000000066 13457024252 014506
xustar00mzfrmzfr000000 000000 30 mtime=1555835050.729934811
24 SCHILY.fflags=extent
flag.txt000644 001750 001751 00000000352 13457024252 012533
0ustar00mzfrmzfr000000 000000 hmm..So you've got the flag.txt file good!!
Now let's play a bit with bases
This is the flag but it's encoded twice with 2 different bases. Figure it out

T1JaSFMyREJNTlZXMlpMM01JWVc0NVpVTlJWVjZNRFNMNVREQTRSVE5WWFRL
NUQ1Qkk9PT09PT0K
```

#1 Find the flag.

I love cyberchef



Recipe

From Base64

Alphabet
A-Za-z0-9+ ...

☑ Remove non-alphabet chars

From Base32

Alphabet
A-Z2-7=

☑ Remove non-alphabet chars

Input

```
length: 76
lines:  1
start: 76
  end: 76
length:  0
```

```
T1JaSFMyREJNTlZXMlpMM01JWVc0NVpVTlJWVjZNR
FNMNVREQTRSVE5WWFRLNUQ1Qkk9PT09PT0K
```

Output

```
time:   7ms
length:  31
lines:    2
```

```
tryhackme{b1nw4lk_0r_f0r3mo5t}
```

[Part 3] #1 Find the Flag

#1 Find the flag.

Download the next image: hide.jpg

Hint: You know the drill, focus on the Title.

#1 Find the flag.

This tool is really popular:

steghide - a steganography program

To extract:

Example:

```
$ steghide extract -sf picture.jpg

 Enter passphrase:

wrote extracted data to "secret.txt".
```

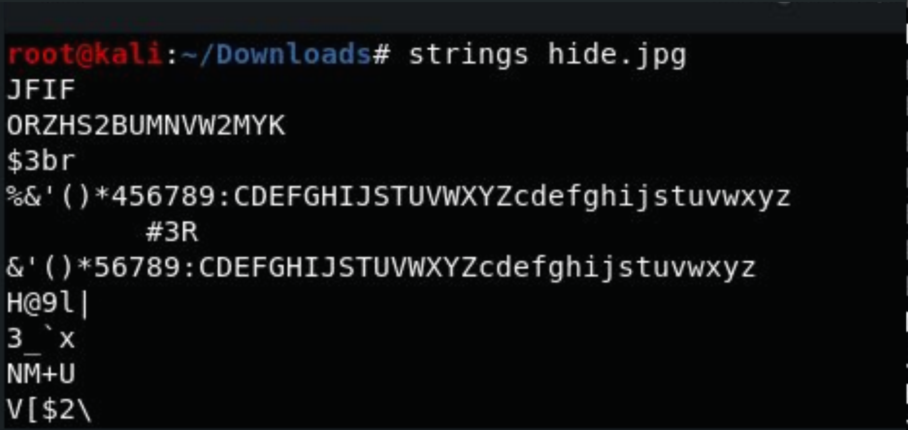
#1 Find the flag.

```
root@kali:~/Downloads# steghide extract -sf hide.jpg

Enter passphrase:

steghide: could not extract any data with that passphrase!

root@kali:~/Downloads#
```

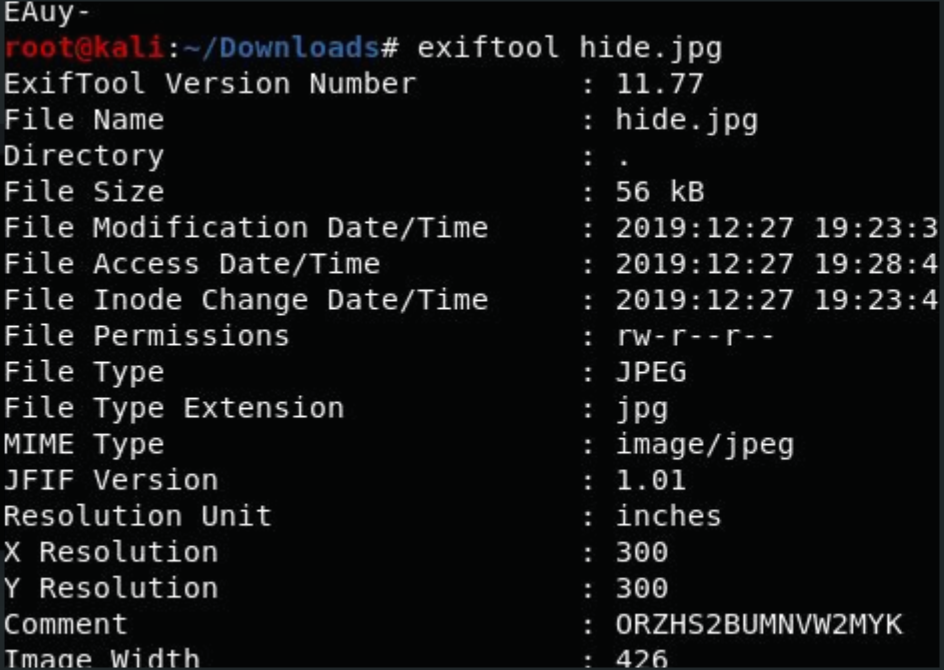
Oh No. The passphase must be hidden in the image.

#1 Find the flag.

You can find the password 2 ways:



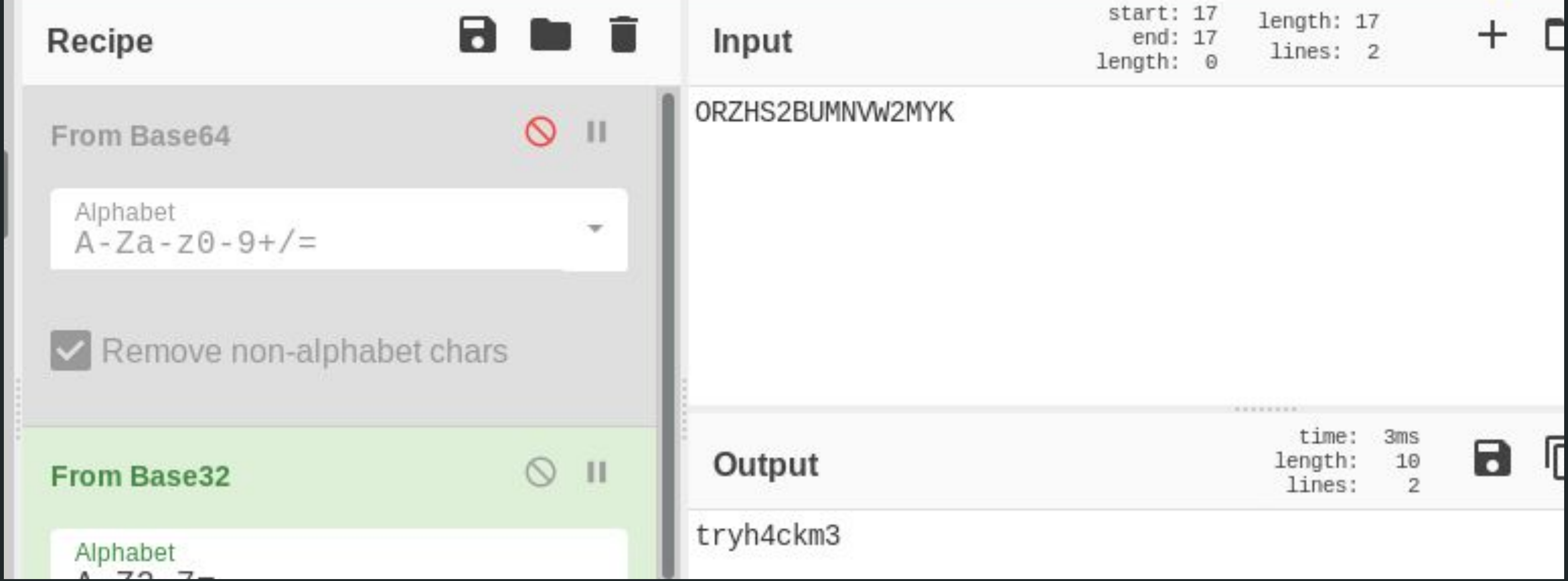
```
root@kali:~/Downloads# strings hide.jpg
JFIF
ORZHS2BUMNVW2MYK
$3br
%&'()*456789:CDEFGHIJSTUVWXYZcdefghijstuvwxyz
        #3R
&'()*56789:CDEFGHIJSTUVWXYZcdefghijstuvwxyz
H@9l|
3_`x
NM+U
V[$2\
```



```
EAuy-
root@kali:~/Downloads# exiftool hide.jpg
ExifTool Version Number         : 11.77
File Name                       : hide.jpg
Directory                       : .
File Size                       : 56 kB
File Modification Date/Time     : 2019:12:27 19:23:3
File Access Date/Time           : 2019:12:27 19:28:4
File Inode Change Date/Time     : 2019:12:27 19:23:4
File Permissions                : rw-r--r--
File Type                       : JPEG
File Type Extension             : jpg
MIME Type                       : image/jpeg
JFIF Version                    : 1.01
Resolution Unit                 : inches
X Resolution                    : 300
Y Resolution                    : 300
Comment                         : ORZHS2BUMNVW2MYK
Image Width                     : 426
```

#1 Find the flag.

- Tried ORZHS2BUMNVW2MYK
- Realized it was encoded
- Used cyberchef



From Base64

Alphabet
A-Za-z0-9+/=

☑ Remove non-alphabet chars

From Base32

Alphabet
A-Z2-7=

```
start: 17    length: 17
  end: 17     lines:  2
length: 0
```

```
ORZHS2BUMNVW2MYK
```

```
  time: 3ms
length:  10
 lines:   2
```

```
tryh4ckm3
```

#1 Find the flag.

```
root@kali:~/Downloads# steghide extract -sf hide.jpg
Enter passphrase:

wrote extracted data to "flag-1.txt".

root@kali:~/Downloads# cat flag-1.txt
Steghide is a great tool to find some hidden data that couldn't be extracted using binwalk.
```

Note: steghide doesn't need password always

tryhackme{st3gh1d3_i5_l0v3}

[Part 4] #1 Find the flag.

#1 Find the flag.

Download: stegano.png

Hint:

Hiding data in LSB are a very common process. Especially in CTFs.

The most famous tool used for this is KDE68

P.S: Name of the tool is encrypted in a version of ROT cipher.

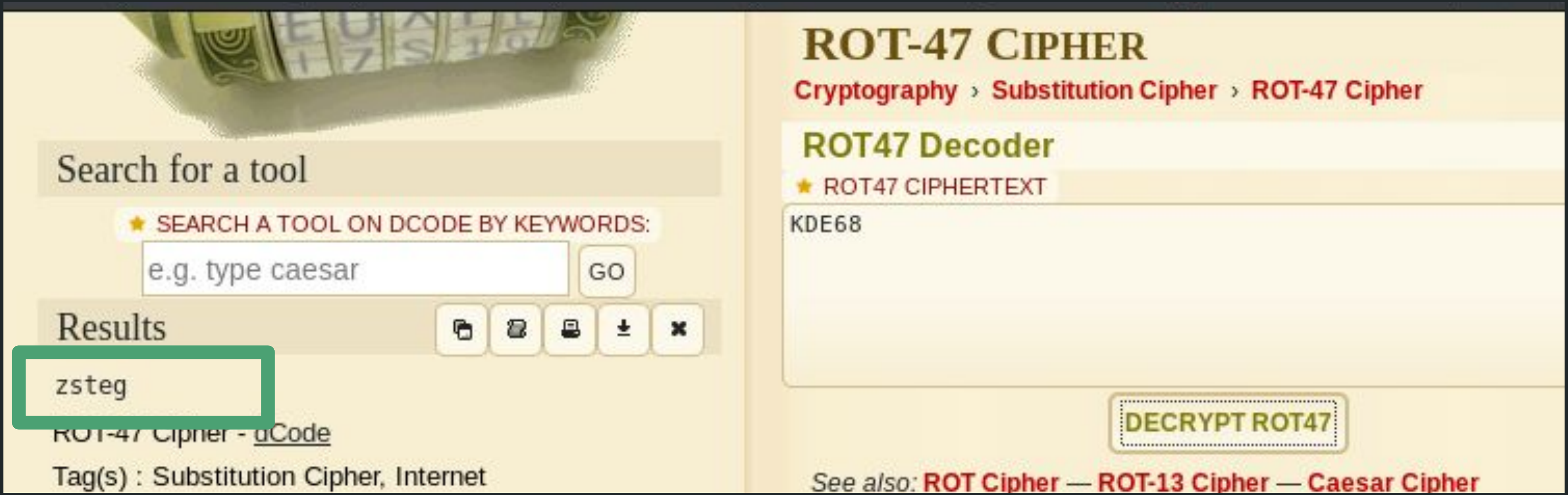
P.P.S: I repeat decode KDE68 to find the name of the tool.

(Hint look up ROT13 variants)

#1 Find the flag.

Decode KDE68

- Tried a bunch of different things until something worked



```
zsteg
```

ROT-47 Cipher - dCode

Tag(s) : Substitution Cipher, Internet

ROT-47 CIPHER

Cryptography › Substitution Cipher › ROT-47 Cipher

ROT47 Decoder

★ ROT47 CIPHERTEXT

```
KDE68
```

See also: ROT Cipher — ROT-13 Cipher — Caesar Cipher

#1 Find the flag.

➔ You have to download zsteg

https://github.com/zed-0xff/zsteg

➔ Extract and run:

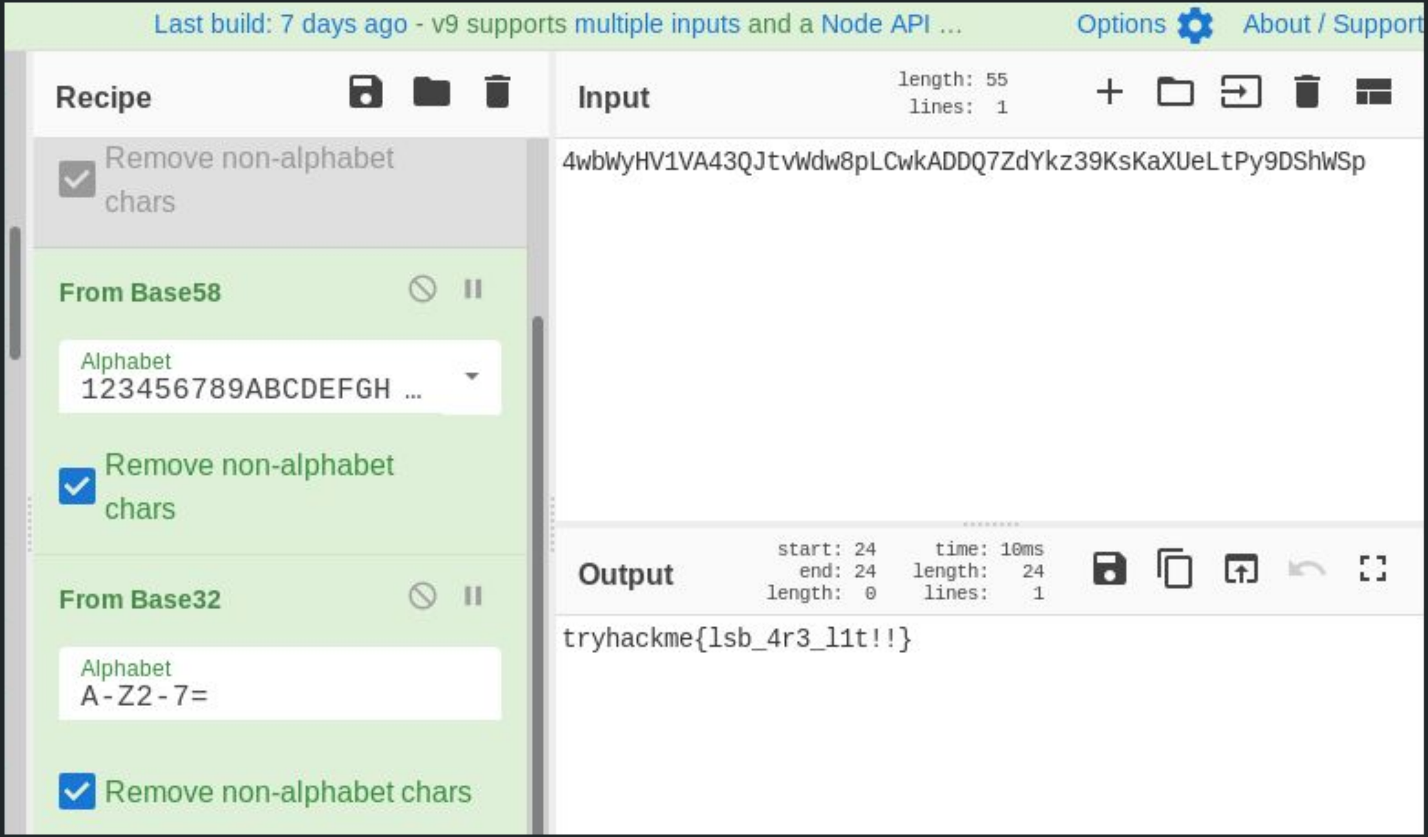
```
root@kali:~/zsteg-master# gem install zsteg
```

#1 Find the flag.

```
root@kali:~/Downloads# zsteg stegano.png
imagedata              .. text: "ywx46+%)"
b1,bgr,lsb,xy          .. text:
"=flag=4wbWyHV1VA43QJtvWdw8pLCwkADDQ7ZdYkz39KsKaXUeLtPy9DShWSp\n
....
```

#1 Find the flag.

I love cyberchef



Last build: 7 days ago - v9 supports multiple inputs and a Node API ...

Options

About / Support

Recipe

☑ Remove non-alphabet chars

From Base58

Alphabet
123456789ABCDEFGH ...

☑ Remove non-alphabet chars

From Base32

Alphabet
A-Z2-7=

☑ Remove non-alphabet chars

Input

```
length: 55
 lines:  1
```

```
4wbWyHV1VA43QJtvWdw8pLCwkADDQ7ZdYkz39KsKaXUeLtPy9DShWSp
```

Output

```
 start: 24     time: 10ms
   end: 24   length:   24
length:  0    lines:    1
```

```
tryhackme{lsb_4r3_l1t!!}
```

[Part 5] #1 Since you've been working hard..I wanted to hand out the flag to you but my dumb friend messed the whole image. Fix the image to get the flag.

#1 Fix the image.

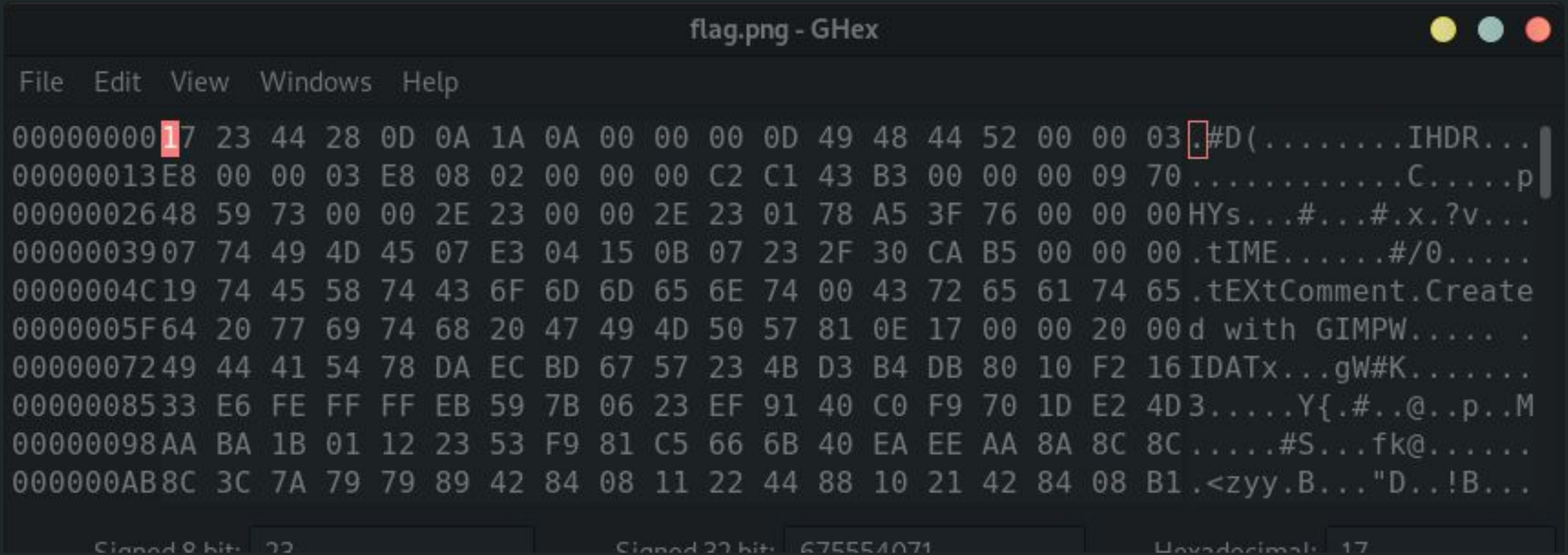
Download: flag.png

There are a lot of ways to mess a file. The most common one is to play with its headers.

NOTE: The flag is not in the tryhackme{}. For submission add tryhackme{} around the found flag.

#1 Fix the image.

Open the image with ghex and check the file signature: 17 23 44 28 0D 0A 1A ..

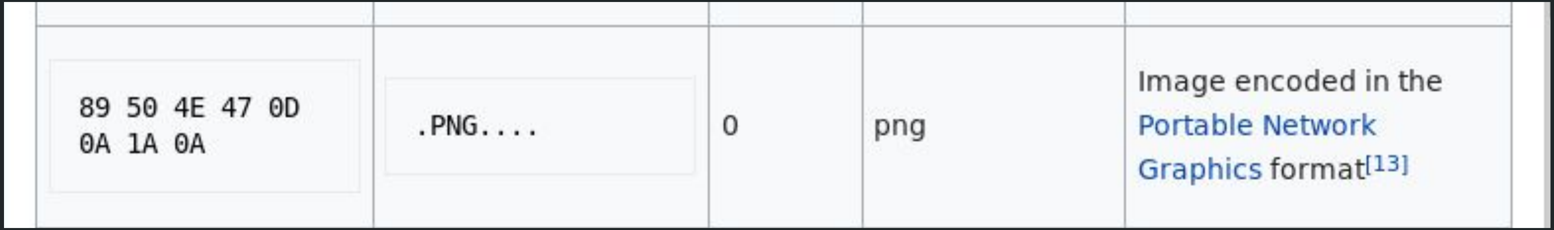


```
0000000017 23 44 28 0D 0A 1A 0A 00 00 00 0D 49 48 44 52 00 00 03.#D(........IHDR...
00000013E8 00 00 03 E8 08 02 00 00 00 C2 C1 43 B3 00 00 00 09 70.............C.....p
0000002648 59 73 00 00 2E 23 00 00 2E 23 01 78 A5 3F 76 00 00 00HYs...#...#.x.?v...
0000003907 74 49 4D 45 07 E3 04 15 0B 07 23 2F 30 CA B5 00 00 00.tIME.......#/0.....
0000004C19 74 45 58 74 43 6F 6D 6D 65 6E 74 00 43 72 65 61 74 65.tEXtComment.Create
0000005F64 20 77 69 74 68 20 47 49 4D 50 57 81 0E 17 00 00 20 00d with GIMPW..... .
0000007249 44 41 54 78 DA EC BD 67 57 23 4B D3 B4 DB 80 10 F2 16IDATx....gW#K.......
0000008533 E6 FE FF FF EB 59 7B 06 23 EF 91 40 C0 F9 70 1D E2 4D3.....Y{.#..@..p..M
00000098AA BA 1B 01 12 23 53 F9 81 C5 66 6B 40 EA EE AA 8A 8C 8C.....#S...fk@......
000000AB8C 3C 7A 79 79 89 42 84 08 11 22 44 88 10 21 42 84 08 B1.<zyy.B..."D..!B...
```

Signed 8 bit: 23 Signed 32 bit: 675554071 Hexadecimal: 17

#1 Fix the image.

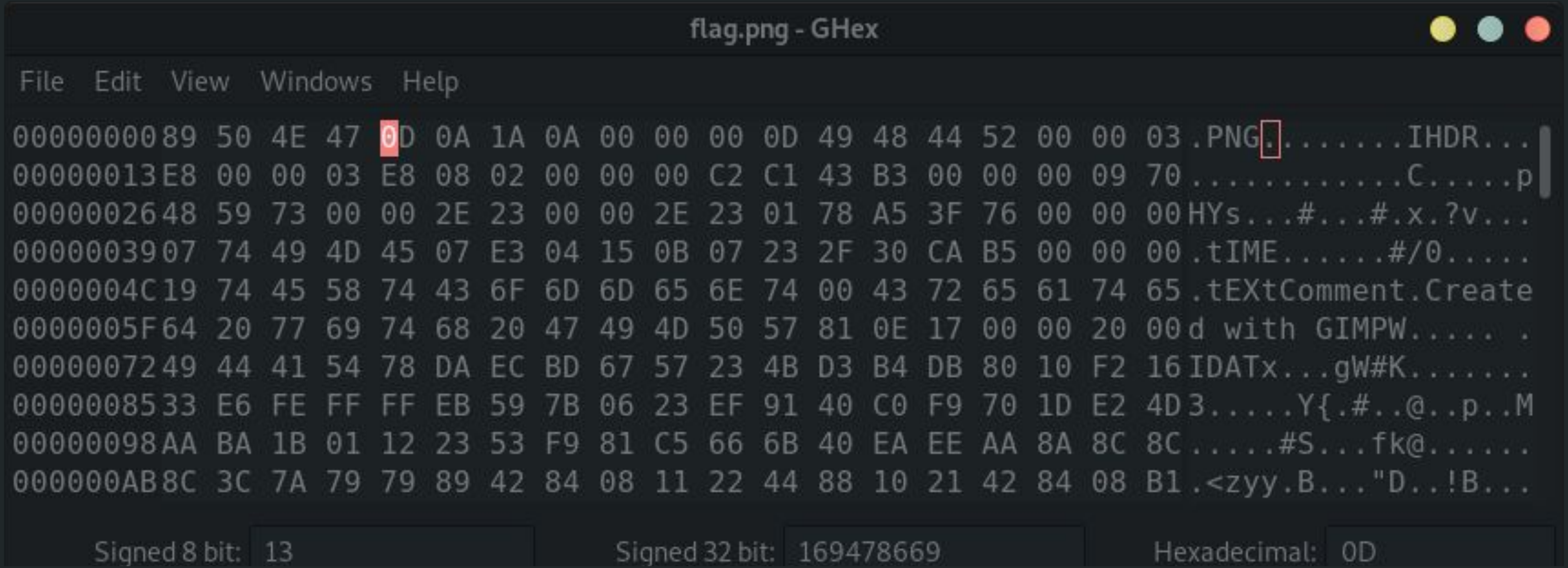
Lookup the file signature for png and compare it with:
17 23 44 28 0D 0A 1A ..



89 50 4E 47 0D 0A 1A 0A	.PNG....	0	png	Image encoded in the Portable Network Graphics format[13]

This doesn't match.
Edit the hex on flag.png to match, and then save it.

#1 Fix the image.



```
0000000089 50 4E 47 0D 0A 1A 0A 00 00 00 0D 49 48 44 52 00 00 03.PNG........IHDR...
00000013E8 00 00 03 E8 08 02 00 00 00 C2 C1 43 B3 00 00 00 09 70............C.....p
0000002648 59 73 00 00 2E 23 00 00 2E 23 01 78 A5 3F 76 00 00 00HYs...#...#.x.?v...
0000003907 74 49 4D 45 07 E3 04 15 0B 07 23 2F 30 CA B5 00 00 00.tIME......#/0.....
0000004C19 74 45 58 74 43 6F 6D 6D 65 6E 74 00 43 72 65 61 74 65.tEXtComment.Create
0000005F64 20 77 69 74 68 20 47 49 4D 50 57 81 0E 17 00 00 20 00d with GIMPW..... .
0000007249 44 41 54 78 DA EC BD 67 57 23 4B D3 B4 DB 80 10 F2 16IDATx...gW#K.......
0000008533 E6 FE FF FF EB 59 7B 06 23 EF 91 40 C0 F9 70 1D E2 4D3.....Y{.#..@..p..M
00000098AA BA 1B 01 12 23 53 F9 81 C5 66 6B 40 EA EE AA 8A 8C 8C.....#S...fk@......
000000AB8C 3C 7A 79 79 89 42 84 08 11 22 44 88 10 21 42 84 08 B1.<zyy.B..."D..!B...
```

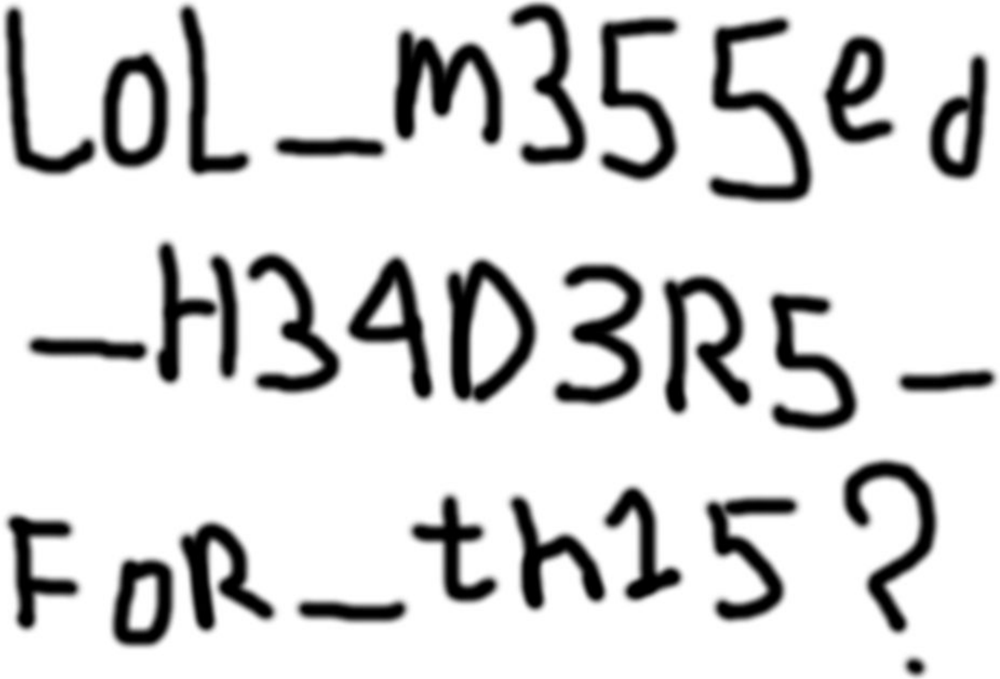
Signed 8 bit: 13   Signed 32 bit: 169478669   Hexadecimal: 0D

#1 Fix the image.

The fixed image is:

And the flag is:

tryhackme{LoL_m355ed_H34D3R5_FoR_th15?}



[Part 6] #1 Audio?!

#1 Audio?!

Download flag.wav

Hint:

HACKER1: FBI is onto me that is why I am sending you a hidden message in an audio file.

HACKER2: What? Audio file...how the hell is that safe.

H1: It is because audio has nothing to do with it.

H2: So how can I see it.

H1: Just check the spectro......

-----DISCONNECTED--------------------

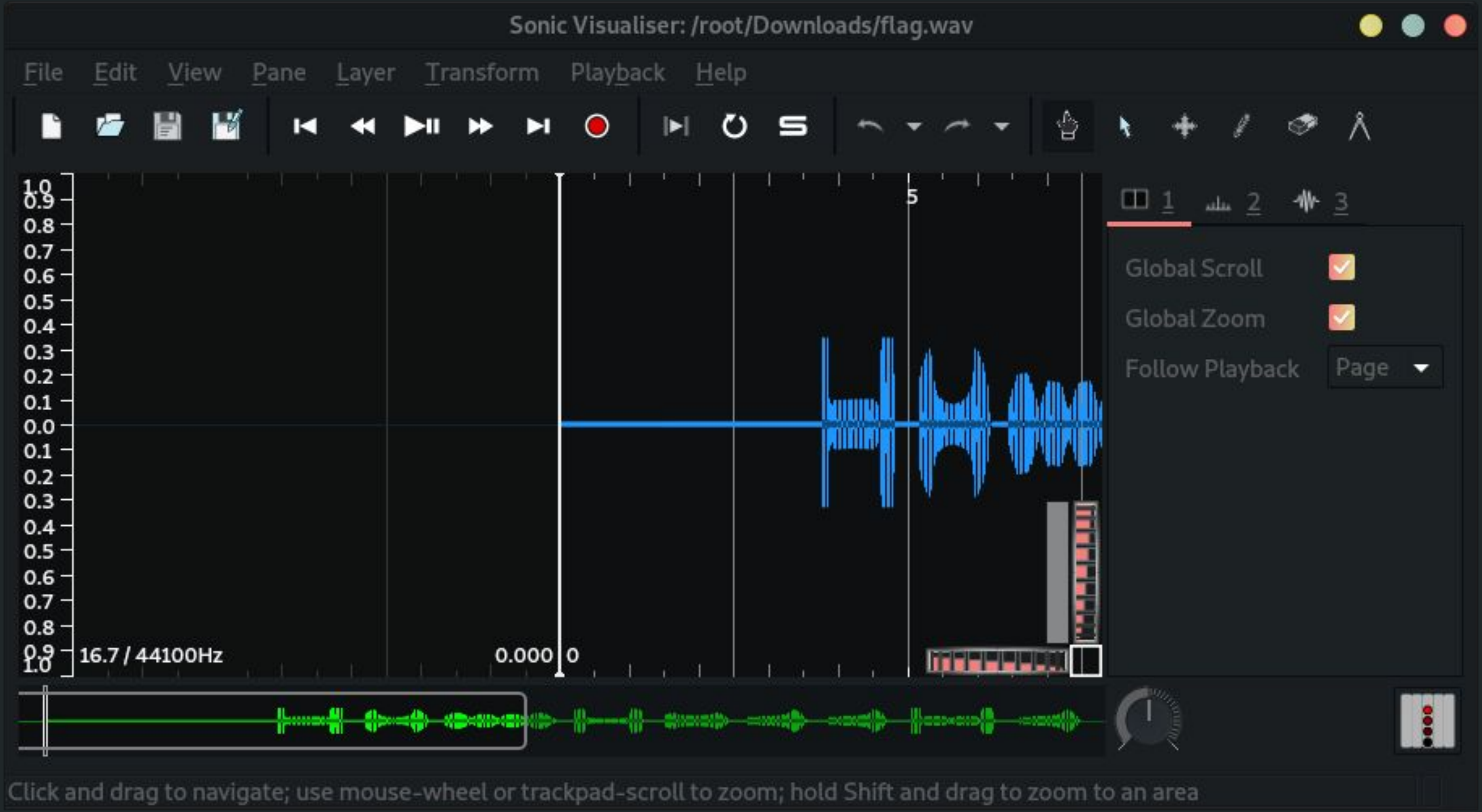
This was the conversation intercepted by FBI between two hackers. FBI has provided you with the audio file can you help then find the message?

#1 Audio?!

In the hint it says check the spectro
After much searching I found a tool:

```
sonic-visualiser/kali-rolling 4.0-1 amd64
  viewing and analysing the contents of music audio files
```

Downloaded it and opened the file



Global Scroll ☑

Global Zoom ☑

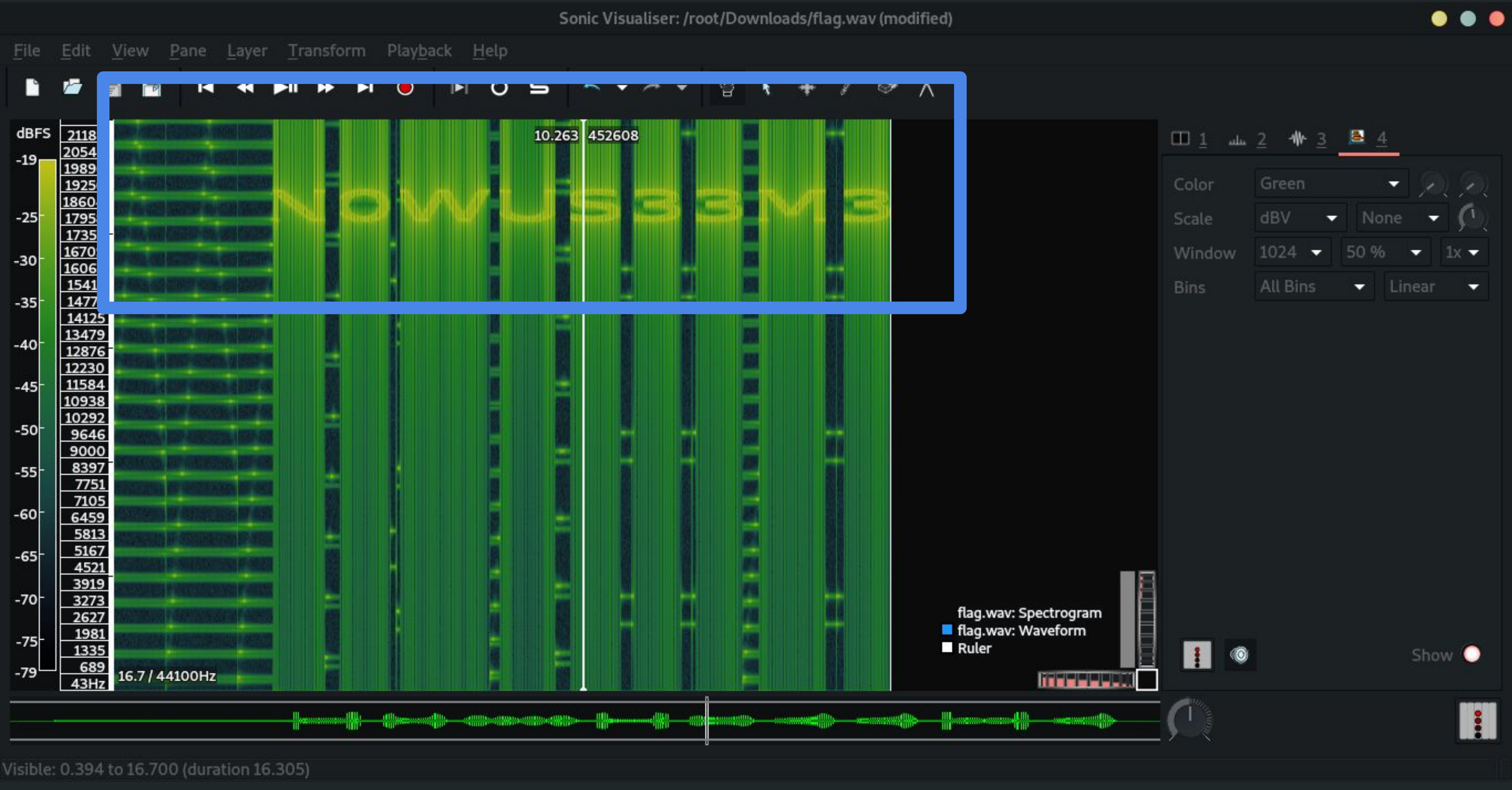
Follow Playback: Page

Click and drag to navigate; use mouse-wheel or trackpad-scroll to zoom; hold Shift and drag to zoom to an area

#1 Audio?!

To reveal the flag:

Layer > Add Spectrogram



Sonic Visualiser: /root/Downloads/flag.wav (modified)

Color: Green
Scale: dBV, None
Window: 1024, 50 %, 1x
Bins: All Bins, Linear

flag.wav: Spectrogram
flag.wav: Waveform
Ruler

Visible: 0.394 to 16.700 (duration 16.305)

#1 Audio?!

That is so cool...

Flag is:

tryhackme{NOWUS33M3}

[Part 7] #1 Let's start with the basic

#1 Let's start with the basic

Let's start with the basic:

Aopz pz h Jhlzhy jpwoly zopmalk zlclu wvzpapvuz zv h pz lxbpchslua av o huk zv vu.

Doha fvb ullk pz h mshn ypnoa ayfohjrtl{Uv_jhlzhy_Uv_Jyfwav}

#1 Let's start with the basic

Text has been shifted. We have to figure out how much.

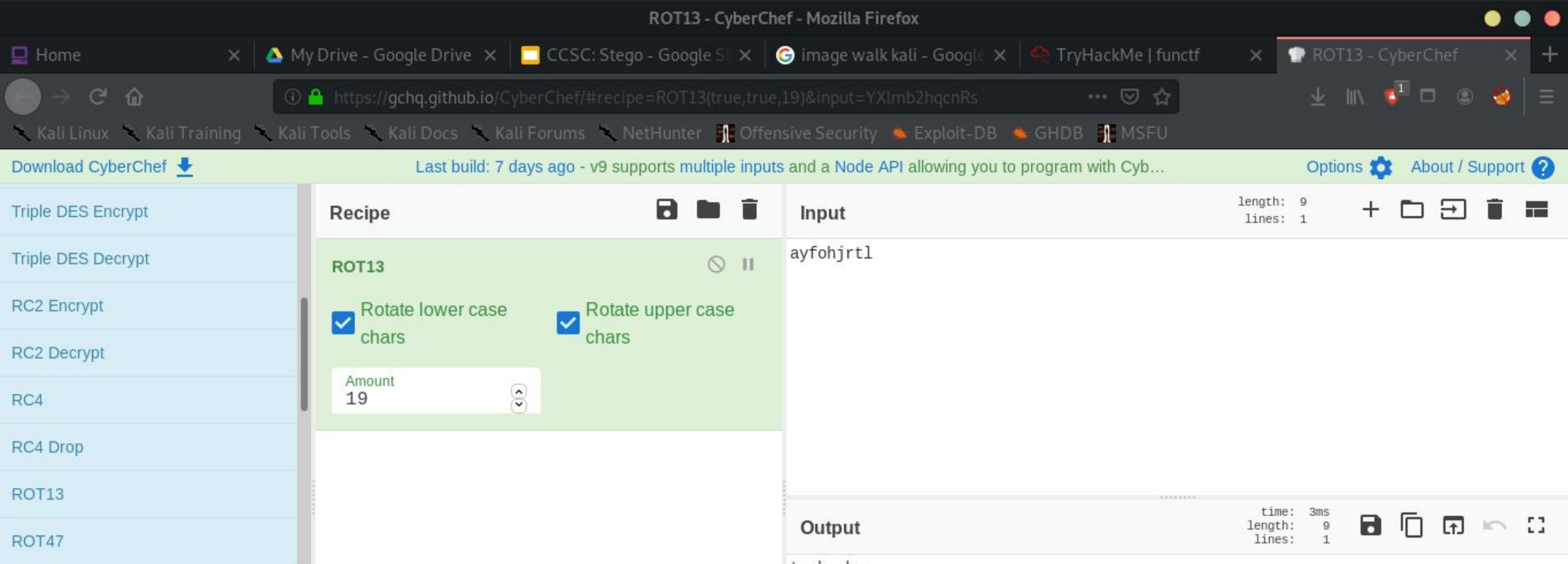
The last bit in the phrase is obviously the flag:

ayfohjrtl{Uv_jhlzhy_Uv_Jyfwav}

ayfohjrtl == tryhackme

#1 Let's start with the basic

Using ROT13



ROT13 - CyberChef - Mozilla Firefox

https://gchq.github.io/CyberChef/#recipe=ROT13(true,true,19)&input=YXlmb2hqcnRs

Last build: 7 days ago - v9 supports multiple inputs and a Node API allowing you to program with Cyb…

Recipe

ROT13

☑ Rotate lower case chars

☑ Rotate upper case chars

Amount
19

Input

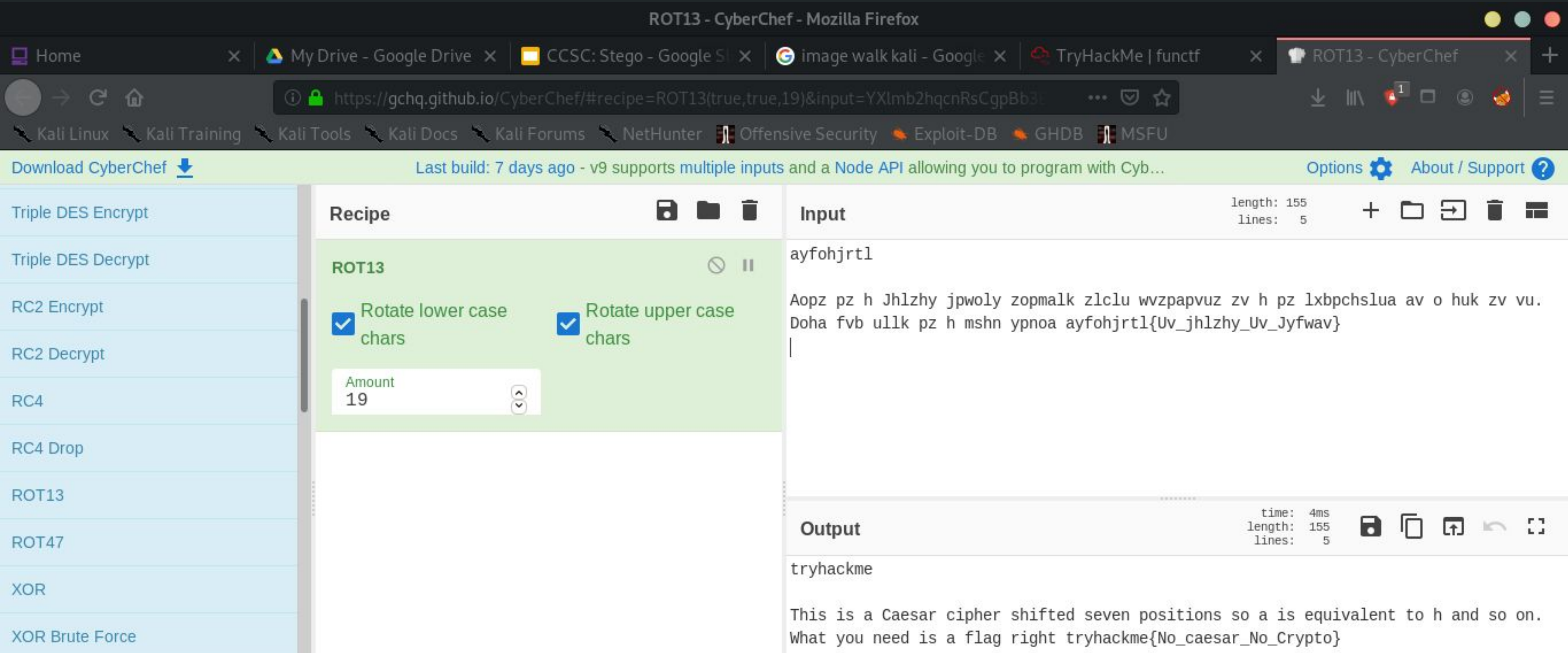
```
length: 9
 lines: 1
```

```
ayfohjrtl
```

Output

```
  time:  3ms
length:    9
 lines:    1
```

#1 Let's start with the basic



ROT13

☑ Rotate lower case chars ☑ Rotate upper case chars

Amount 19

Input — length: 155 lines: 5

```
ayfohjrtl

Aopz pz h Jhlzhy jpwoly zopmalk zlclu wvzpapvuz zv h pz lxbpchslua av o huk zv vu.
Doha fvb ullk pz h mshn ypnoa ayfohjrtl{Uv_jhlzhy_Uv_Jyfwav}
```

Output — time: 4ms length: 155 lines: 5

```
tryhackme

This is a Caesar cipher shifted seven positions so a is equivalent to h and so on.
What you need is a flag right tryhackme{No_caesar_No_Crypto}
```

[Part 7] #2 Let's start with the basic

#2 Let's start with the basic

Guvf gvzr gurl ner fuvsgrq guvegrra cbfvgvbaf gung vf jul vg'f pnyyrq EBG guvegrra.

SYNT: gelunpxzr{ebg_guvegrra_vf_nyfb_pnrfne_pvcure}

#2 Let's start with the basic

Focusing on this

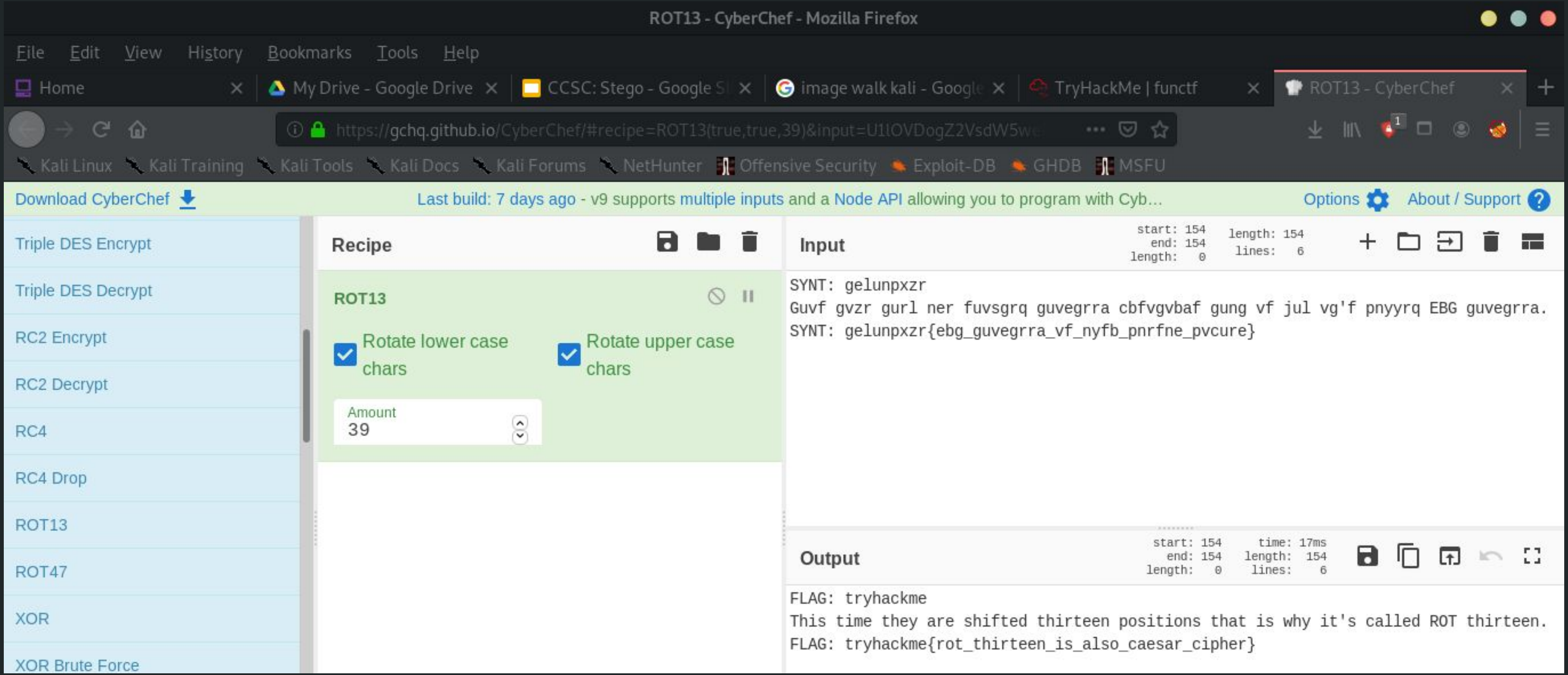
SYNT: gelunpxzr{ebg_guvegrra_vf_nyfb_pnrfne_pvcure}

SYNT is hint maybe?

→ (Its actually flag)

→ Using the same technique

#1 Let's start with the basic



ROT13

☑ Rotate lower case chars ☑ Rotate upper case chars

Amount 39

Input

```
start: 154   length: 154
  end: 154    lines:   6
length:   0
```

```
SYNT: gelunpxzr
Guvf gvzr gurl ner fuvsgrq guvegrra cbfvgvbaf gung vf jul vg'f pnyyrq EBG guvegrra.
SYNT: gelunpxzr{ebg_guvegrra_vf_nyfb_pnrfne_pvcure}
```

Output

```
start: 154     time: 17ms
  end: 154   length: 154
length:   0   lines:   6
```

```
FLAG: tryhackme
This time they are shifted thirteen positions that is why it's called ROT thirteen.
FLAG: tryhackme{rot_thirteen_is_also_caesar_cipher}
```

[Part 7] #3 What the hell is this?

#3 What the hell is this?

(@29]]]H:== E9:D 6G6C DE@An x >62? H6 42? ;FDE D9:7E E@ 2?J 2>@F?E @7 A@D:E:@?D H:E9 H926G6C 492C24E6C D6E]

ECJ924<>6Lu=2v0xD0p==0x0Hp?E0?@0q$N

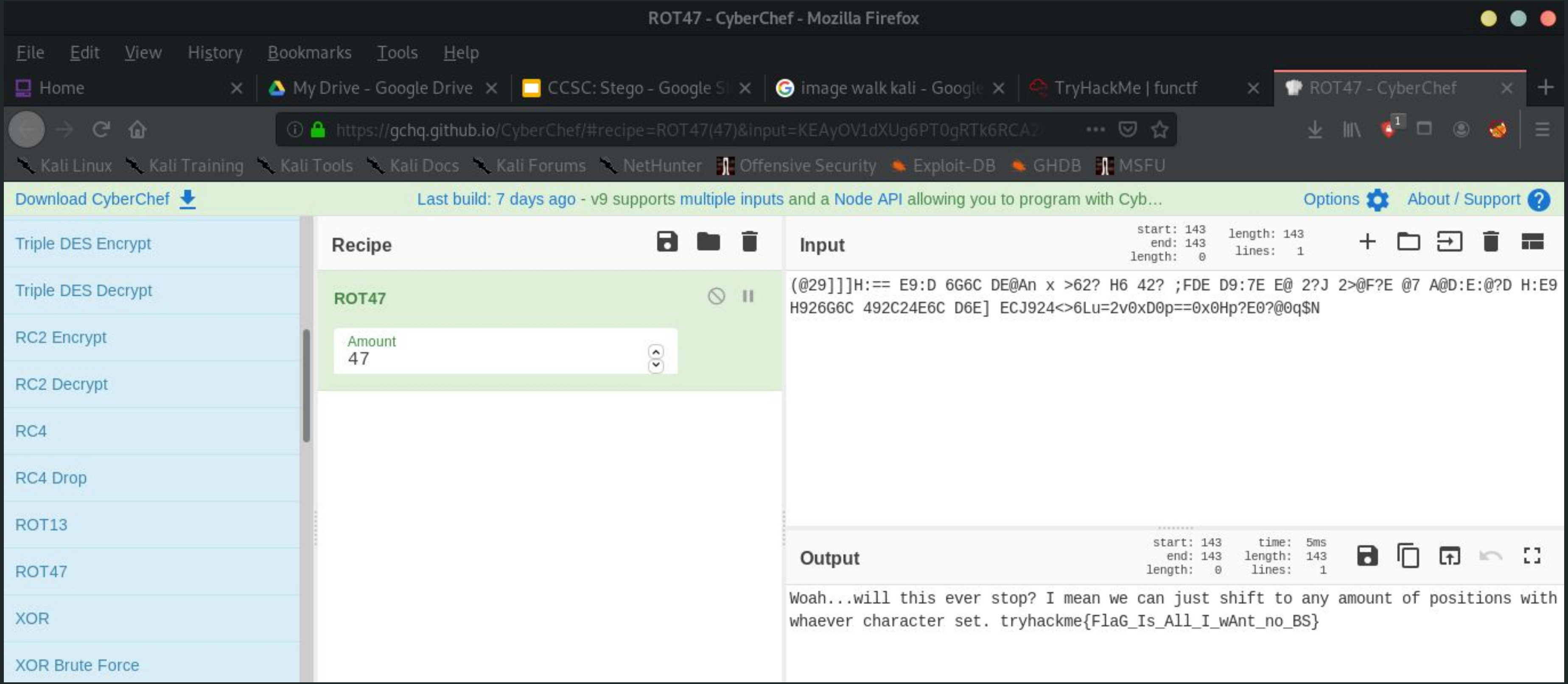
#3 What the hell is this?

(@29]]]H:== E9:D 6G6C DE@An x >62? H6 42? ;FDE D9:7E E@ 2?J 2>@F?E @7 A@D:E:@?D H:E9 H926G6C 492C24E6C D6E]
ECJ924<>6Lu=2v0xD0p==0x0Hp?E0?@0q$N

➜ They have probably shifted more than just letters

⇒ The encoding that does that is ROT47

#3 What the hell is this?



ROT47

Amount 47

Input

```
start: 143   length: 143
  end: 143    lines:   1
length:   0
```

```
(@29]]]H:== E9:D 6G6C DE@An x >62? H6 42? ;FDE D9:7E E@ 2?J 2>@F?E @7 A@D:E:@?D H:E9
H926G6C 492C24E6C D6E] ECJ924<>6Lu=2v0xD0p==0x0Hp?E0?@0q$N
```

Output

```
start: 143     time:   5ms
  end: 143   length:   143
length:   0    lines:     1
```

```
Woah...will this ever stop? I mean we can just shift to any amount of positions with
whaever character set. tryhackme{FlaG_Is_All_I_wAnt_no_BS}
```

[Part 7] #4 What the hell is this?

#4 What the hell is this?

Fmeorcbi gc rmd gyowyb sp sw gd. Afy gybiq gi hewr geld xfo jjkk rbcfkgiwi{TsKcxipo_gGzLcB_mQ_MeCcep_mmNrlp}

P.S: Don't forget to use your brain ;)

#4 What the hell is this?

Keeping only the flag part:

Three different amounts that were shifted by:

tdehmikyk{VuMezkrq_iIbNeD_oS_OgEegr_ooPtKr}

hrsvawymy{JiAsnyfe_wWpBsR_cG_CuSsuf_ccDhYf}

nxybgcese{PoGytelk_cCvHyX_iM_IaYyal_iiJnEl}

--> You can see that together they make tryhackme

#4 What the hell is this?

--> Replace the characters that are in the wrong position with # to make clearer

t##h##k##{V##e##r#_#I##e#r_#s_##E#r_##P##r}

#r##a##m#{#i##s##e_##p##R_##_C##s##_c##h##}

##y##c##e{##G##e##_c##H##_i#_#a##a#_#i##E#}

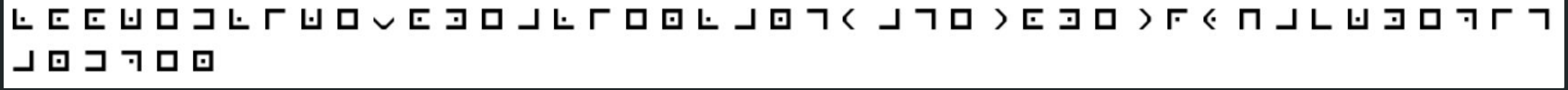
the flag is:

tryhackme{ViGesere_cIpHeR_iS_CaEsar_ciPhEr}

--> I figured this out with pencil and paper I am sure there is faster way

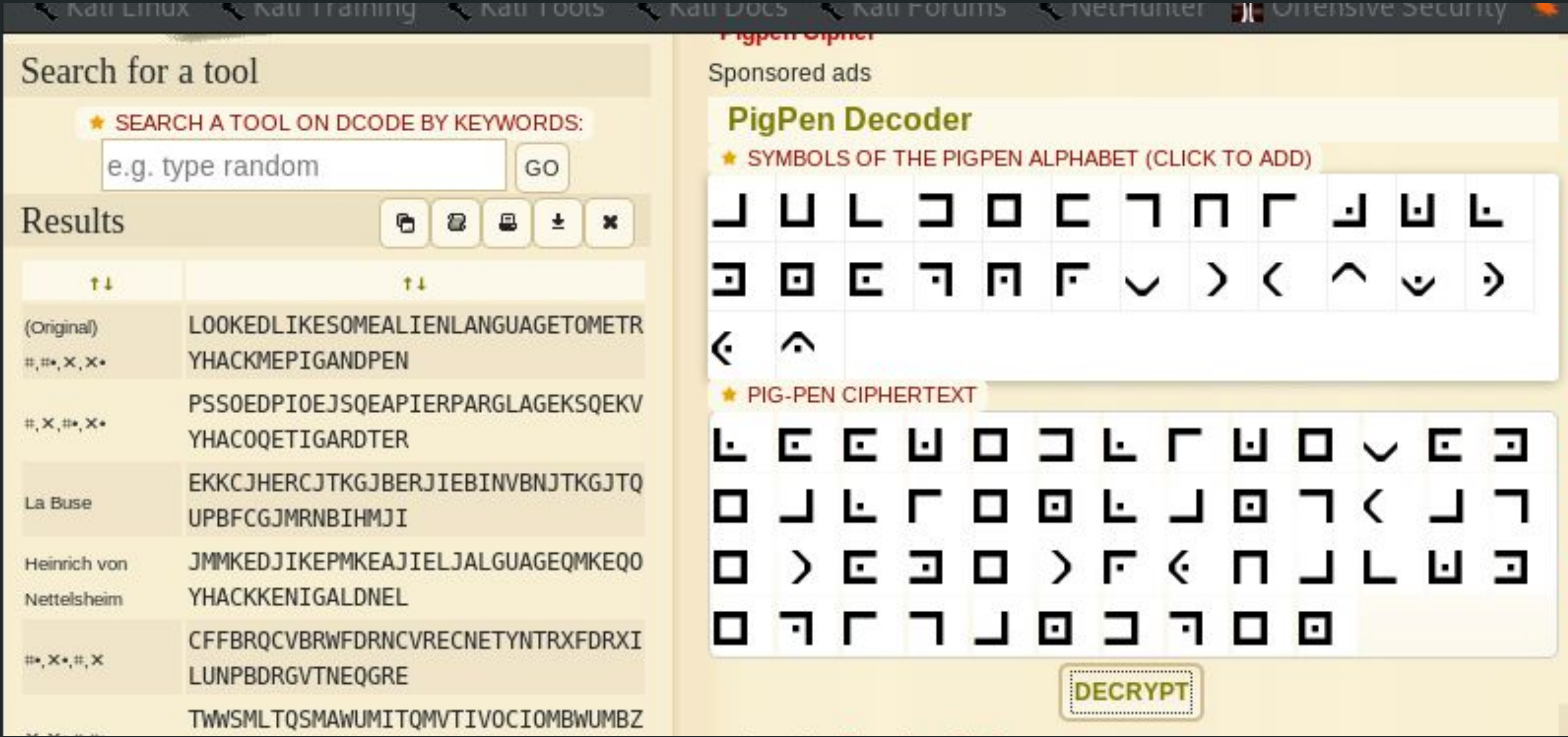
[Part 8] #1 Ancient Times

#1 Ancient Times



#1 Ancient Times

- Looked around until I found out what it was
- Pigpen cipher
- Found a site to decode it



↑↓	↑↓
(Original) #,#•,×,×•	LOOKEDLIKESOMEALIENLANGUAGETOMETR YHACKMEPIGANDPEN
#,×,#•,×•	PSSOEDPIOEJSQEAPIERPARGLAGEKSQEKV YHACOQETIGARDTER
La Buse	EKKCJHERCJTKGJBERJIEBINVBNJTKGJTQ UPBFCGJMRNBIHMJI
Heinrich von Nettelsheim	JMMKEDJIKEPMKEAJIELJALGUAGEQMKEQO YHACKKENIGALDNEL
#•,×•,#,×	CFFBRQCVBRWFDRNCVRECNETYNTRXFDRXI LUNPBDRGVTNEQGRE
	TWWSMLTQSMAWUMITQMVTIVOCIOMBWUMBZ

[Part 9]: #1 Genetics

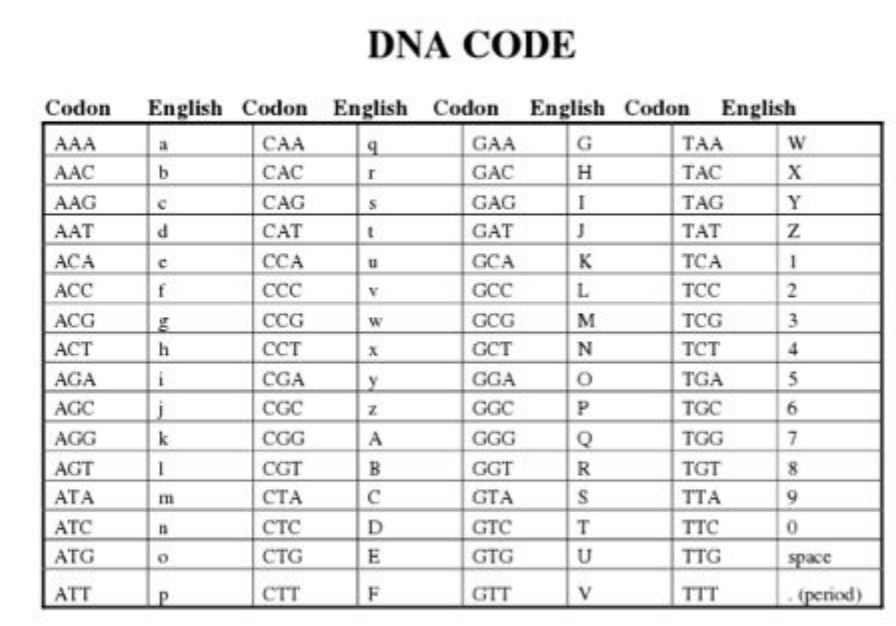
#1 Genetics

I heard scientist found ways to hide data in DNA and stuff. Is it really true?

CTCAAAATAATCTTGATTACAATGATTAGTACATTGAAACACACATTGCCACAG
AGAATCACGTTGAAAATCCGACATACTAGAATCACGTTGCATATGTTGATAAAA
AGGACATTGCATACTACAAGACACTTGATAACACAGCAGAAAACGACATTGCA
GACAAAGCCACACACATTTTTGTAAACAAGTAGTTTGCCGACTATGTTGAAGAA
ACACACACAGTTGGAGTTGAGCCCACAGCATTTGATCACAACAAATTTGATAC
GATTGAATAAAATAATCTTGACCAGTAAAACGTTGGAAACAGCTTCGCATTCAA
AGTGAGACCCAGAC

#1 Genetics

(This actually took me a long time to figure out)
First I looked up dna code to english and scrolled until I found this promising table:

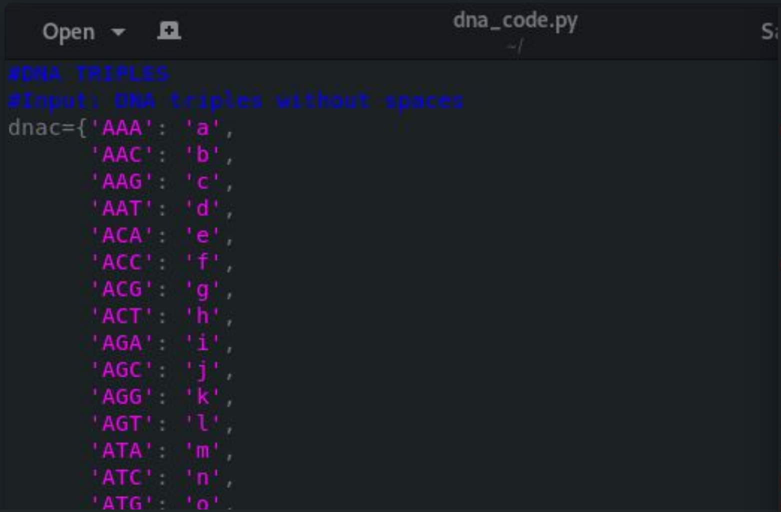


DNA CODE

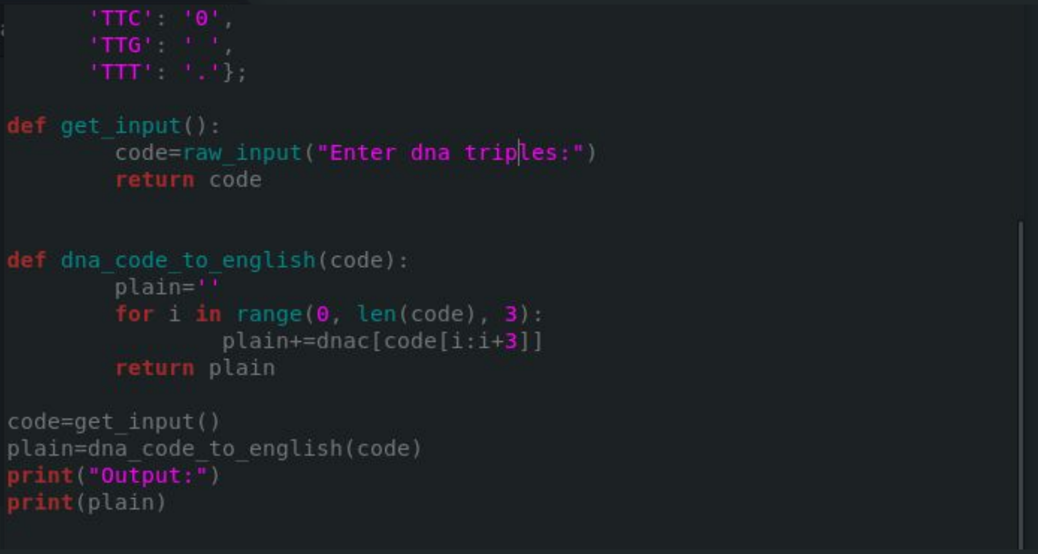
Codon	English	Codon	English	Codon	English	Codon	English
AAA	a	CAA	q	GAA	G	TAA	W
AAC	b	CAC	r	GAC	H	TAC	X
AAG	c	CAG	s	GAG	I	TAG	Y
AAT	d	CAT	t	GAT	J	TAT	Z
ACA	e	CCA	u	GCA	K	TCA	1
ACC	f	CCC	v	GCC	L	TCC	2
ACG	g	CCG	w	GCG	M	TCG	3
ACT	h	CCT	x	GCT	N	TCT	4
AGA	i	CGA	y	GGA	O	TGA	5
AGC	j	CGC	z	GGC	P	TGC	6
AGG	k	CGG	A	GGG	Q	TGG	7
AGT	l	CGT	B	GGT	R	TGT	8
ATA	m	CTA	C	GTA	S	TTA	9
ATC	n	CTC	D	GTC	T	TTC	0
ATG	o	CTG	E	GTG	U	TTG	space
ATT	p	CTT	F	GTT	V	TTT	. (period)

#1 Genetics

I started manually translating it, but then gave up and wrote a program to do it for me.



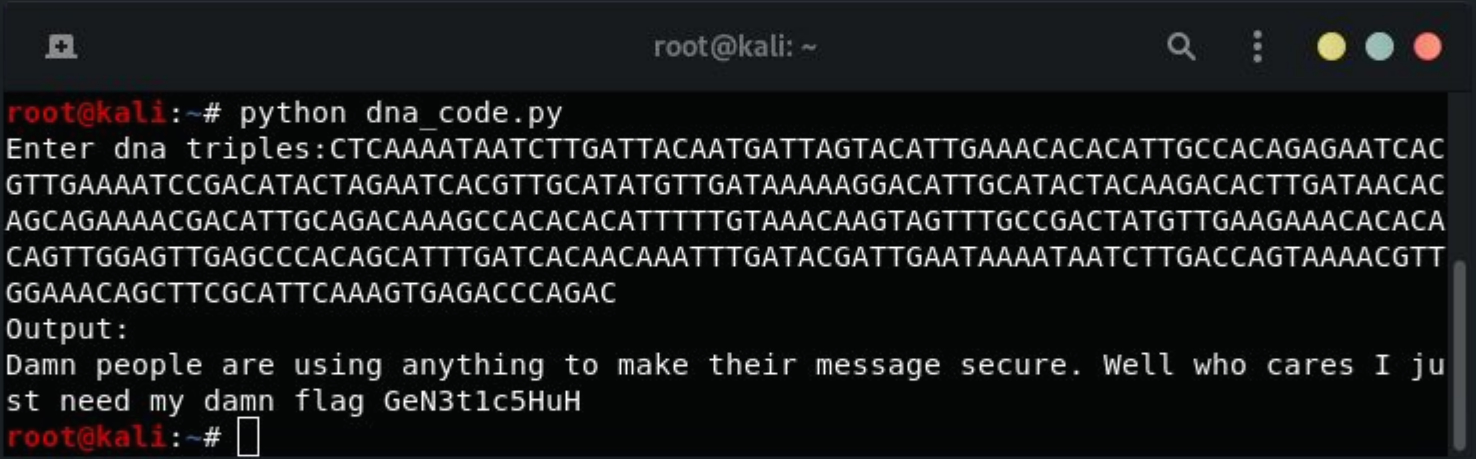
```
#DNA TRIPLES
#Input: DNA triples without spaces
dnac={'AAA': 'a',
      'AAC': 'b',
      'AAG': 'c',
      'AAT': 'd',
      'ACA': 'e',
      'ACC': 'f',
      'ACG': 'g',
      'ACT': 'h',
      'AGA': 'i',
      'AGC': 'j',
      'AGG': 'k',
      'AGT': 'l',
      'ATA': 'm',
      'ATC': 'n',
      'ATG': 'o',
```



```
      'TTC': '0',
      'TTG': ' ',
      'TTT': '.'};

def get_input():
        code=raw_input("Enter dna triples:")
        return code


def dna_code_to_english(code):
        plain=''
        for i in range(0, len(code), 3):
                plain+=dnac[code[i:i+3]]
        return plain

code=get_input()
plain=dna_code_to_english(code)
print("Output:")
print(plain)
```

#1 Genetics

Program Output:



```
root@kali:~# python dna_code.py
Enter dna triples:CTCAAAATAATCTTGATTACAATGATTAGTACATTGAAACACACATTGCCACAGAGAATCAC
GTTGAAAATCCGACATACTAGAATCACGTTGCATATGTTGATAAAAAGGACATTGCATACTACAAGACACTTGATAACAC
AGCAGAAAACGACATTGCAGACAAAGCCACACACATTTTTGTAAACAAGTAGTTTGCCGACTATGTTGAAGAAACACACA
CAGTTGGAGTTGAGCCCACAGCATTTGATCACAACAAATTTGATACGATTGAATAAAATAATCTTGACCAGTAAAACGTT
GGAAACAGCTTCGCATTCAAAGTGAGACCCAGAC
Output:
Damn people are using anything to make their message secure. Well who cares I ju
st need my damn flag GeN3t1c5HuH
root@kali:~# 
```

Flag: tryhackme{GeN3t1c5HuH}

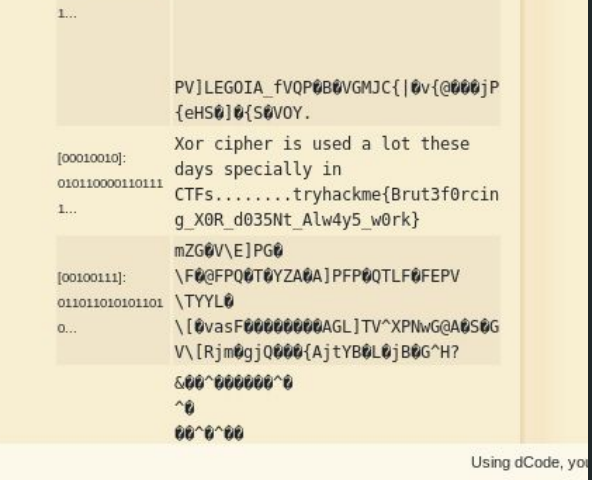
[Part 10] #1 Exclusive Or Random

#1 Exclusive Or Random

Hint: You know you can take any two beautiful messages(strings) and mesh them together and they'll come out complete random.

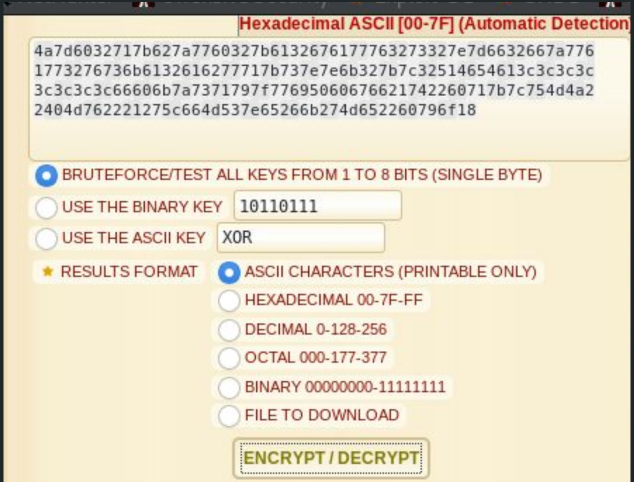
```
4a7d6032717b627a7760327b6132676177763273327e7d6632667a7761773276
736b6132616277717b737e7e6b327b7c32514654613c3c3c3c3c3c3c66606b7
a7371797f776950606766217422260717b7c754d4a22404d762221275c664d537
e65266b274d652260796f18
```

#1 Exclusive Or Random



```
1...                  PV]LEGOIA_fVQP�B�VGMJC{|�v{@���jP
                      {eHS�]�{S�VOY.
[00010010]:           Xor cipher is used a lot these
0101100001101111...   days specially in
                      CTFs........tryhackme{Brut3f0rcin
                      g_X0R_d035Nt_Alw4y5_w0rk}
                      mZG�V\E]PG�
[00100111]:           \F�@FPQ�T�YZA�A]PFP�QTLF�FEPV
0110110101011010...   \TYYL�
                      \[�vasF��������AGL]TV^XPNwG@A�S�G
                      V\[Rjm�gjQ���{AjtYB�L�jB�G^H?
                      &��^������^�
                      ^�
                      ��^�^��
```

Using dCode, yo



Hexadecimal ASCII [00-7F] (Automatic Detection)

```
4a7d6032717b627a7760327b61326761777632733 27e7d6632667a776
1773276736b6132616277717b737e7e6b327b7c32514654613c3c3c3c
3c3c3c3c66606b7a7371797f776950606766217422607 17b7c754d4a2
2404d762221275c664d537e65266b274d652260796f18
```

● BRUTEFORCE/TEST ALL KEYS FROM 1 TO 8 BITS (SINGLE BYTE)

○ USE THE BINARY KEY 10110111

○ USE THE ASCII KEY XOR

RESULTS FORMAT
● ASCII CHARACTERS (PRINTABLE ONLY)
○ HEXADECIMAL 00-7F-FF
○ DECIMAL 0-128-256
○ OCTAL 000-177-377
○ BINARY 00000000-11111111
○ FILE TO DOWNLOAD

You have to brute force it:

[00010010]: 0101100001101111...Xor cipher is used a lot these days specially in CTFs........tryhackme{Brut3f0rcing_X0R_d035Nt_Alw4y5_w0rk}

[Part 11] #1 Morse Code

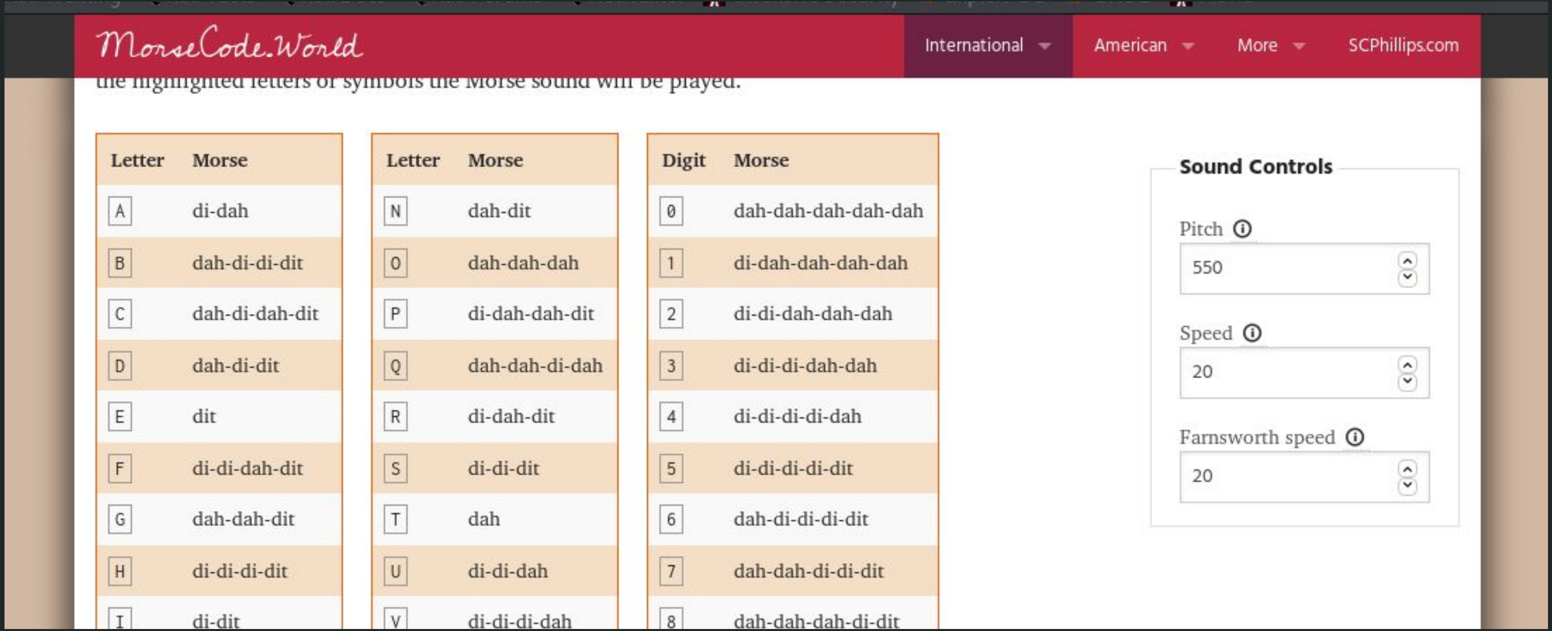
#1 Morse Code

Download the file: morse.txt

Hint: Morse code is being used for a very long time. And since then there has been a lot of versions like using your eyebrows, flashing torches, tapping etc.

#1 Morse Code

Found a table that had the conversion:



the highlighted letters or symbols the Morse sound will be played.

Letter	Morse	Letter	Morse	Digit	Morse
A	di-dah	N	dah-dit	0	dah-dah-dah-dah-dah
B	dah-di-di-dit	O	dah-dah-dah	1	di-dah-dah-dah-dah
C	dah-di-dah-dit	P	di-dah-dah-dit	2	di-di-dah-dah-dah
D	dah-di-dit	Q	dah-dah-di-dah	3	di-di-di-dah-dah
E	dit	R	di-dah-dit	4	di-di-di-di-dah
F	di-di-dah-dit	S	di-di-dit	5	di-di-di-di-dit
G	dah-dah-dit	T	dah	6	dah-di-di-di-dit
H	di-di-di-dit	U	di-di-dah	7	dah-dah-di-di-dit
I	di-dit	V	di-di-di-dah	8	dah-dah-dah-di-dit

Sound Controls

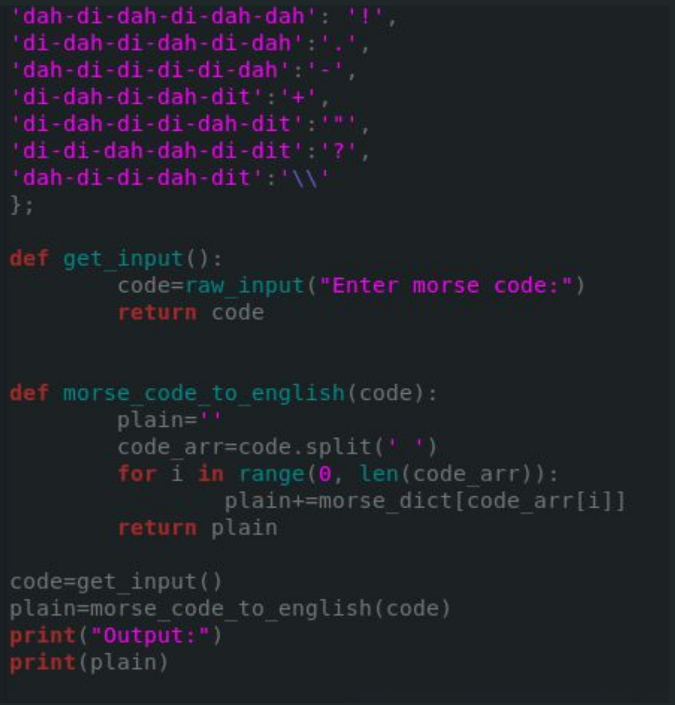
Pitch: 550

Speed: 20

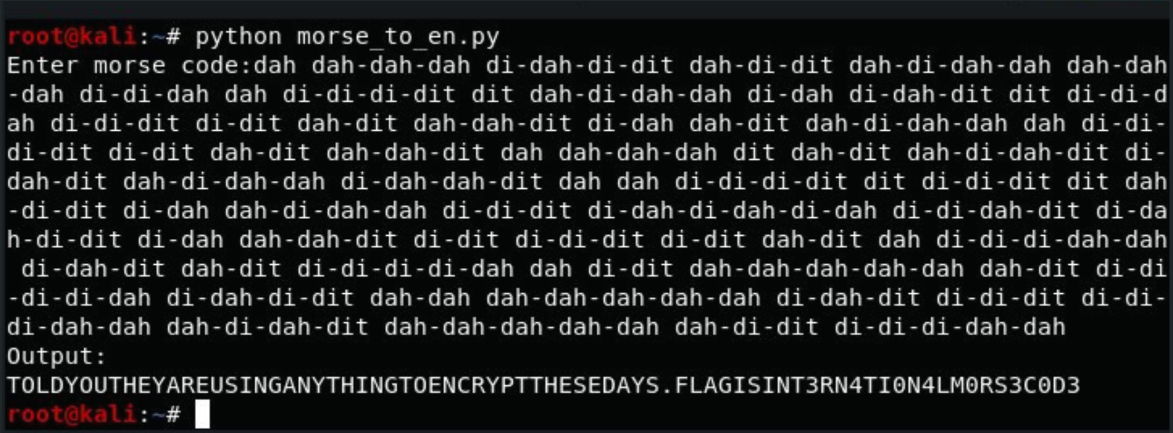
Farnsworth speed: 20

#1 Morse Code

Wrote a program to perform the decoding:



```
'dah-di-dah-di-dah-dah': '!',
'di-dah-di-dah-di-dah':'.',
'dah-di-di-di-di-dah':'-',
'di-dah-di-dah-dit':'+',
'di-dah-di-di-dah-dit':'"',
'di-di-dah-dah-di-dit':'?',
'dah-di-di-dah-dit':'\\'
};

def get_input():
        code=raw_input("Enter morse code:")
        return code


def morse_code_to_english(code):
        plain=''
        code_arr=code.split(' ')
        for i in range(0, len(code_arr)):
                plain+=morse_dict[code_arr[i]]
        return plain

code=get_input()
plain=morse_code_to_english(code)
print("Output:")
print(plain)
```



```
root@kali:~# python morse_to_en.py
Enter morse code:dah dah-dah-dah di-dah-di-dit dah-di-dit dah-di-dah-dah dah-dah
-dah di-di-dah dah di-di-di-dit dit dah-di-dah-dah di-dah di-dah-dit dit di-di-d
ah di-di-dit di-dit dah-dit dah-dah-dit di-dah dah-dit dah-di-dah-dah dah di-di-
di-dit di-dit dah-dit dah-dah-dit dah dah-dah-dah dit dah-dit dah-di-dah-dit di-
dah-dit dah-di-dah-dah di-dah-dah-dit dah dah di-di-di-dit dit di-di-dit dit dah
-di-dit di-dah dah-di-dah-dah di-di-dit di-dah-di-dah-di-dah di-di-dah-dit di-da
h-di-dit di-dah dah-dah-dit di-dit di-di-dit di-dit dah-dit dah di-di-di-dah-dah
 di-dah-dit dah-dit di-di-di-di-dah dah di-dit dah-dah-dah-dah-dah dah-dit di-di
-di-di-dah di-dah-di-dit dah-dah dah-dah-dah-dah-dah di-dah-dit di-di-dit di-di-
di-dah-dah dah-di-dah-dit dah-dah-dah-dah-dah dah-di-dit di-di-di-dah-dah
Output:
TOLDYOUTHEYAREUSINGANYTHINGTOENCRYPTTHESEDAYS.FLAGISINT3RN4TI0N4LM0RS3C0D3
root@kali:~# 
```

End.

There are about 5 more. But they are about reverse engineering, and are unrelated to the topic.

There are a ton of challenges similar to these on this site, and hack the box.

I used to do these challenges in first year:

https://cryptopals.com/sets/1/challenges/

https://www.mysterytwisterc3.org/en/challenges/