c4ptur3-th3-fl4g

it is capture the flag room in TRYHACKME.

https://tryhackme.com/room/c4ptur3th3fl4g

Task - 1 Translation & Shifting:

c4n y0u c4p7u23 7h3 f149?

by seeing this we can understand the flag

c4n - can

y0u - you

c4p7u23 - capture

7h3 - the

f149 - flag

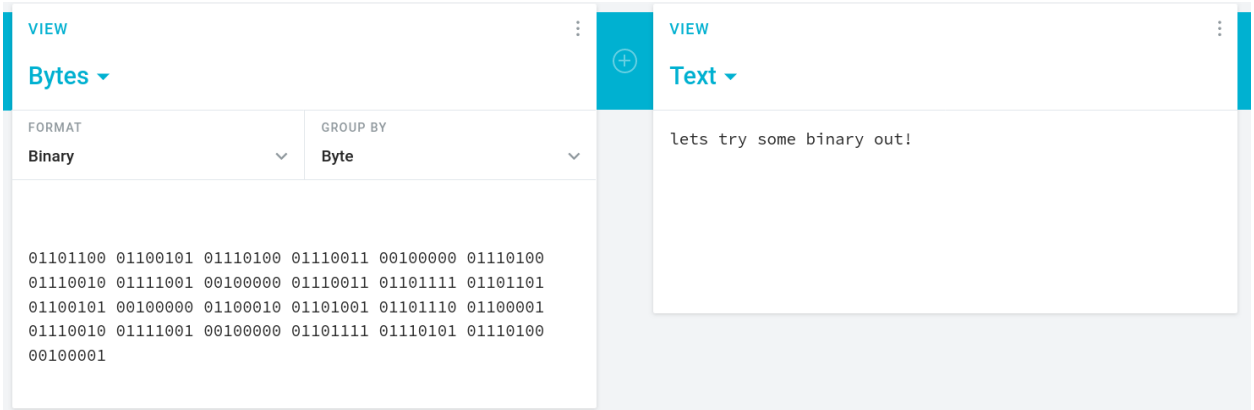
ans: **can you capture the flag**

01101100 01100101 01110100 01110011 00100000 01110100 01110010 01111001 00100000 01110011 01101111 01101101 01100101 00100000 01100010 01101001 01101110 01100001 01110010 01111001 00100000 01101111 01110101 01110100 00100001

what i had done was, i copied and pasted the text in the google so it had given me some online decoders like “cryptii” “convertbiniary” and some other

i had opened one i.e cryptii and pasted my binary in that.

the decode message was



so the answer was

ans: **lets try some binary out!**

MJQXGZJTGIQGS4ZAON2XAZLSEBRW63LNN5XCA2LOEBBVIRRHOM======

this encoding was “BASE32” we can decode that by using online decoder like “cryptii”, “github online tools” and so on.

or

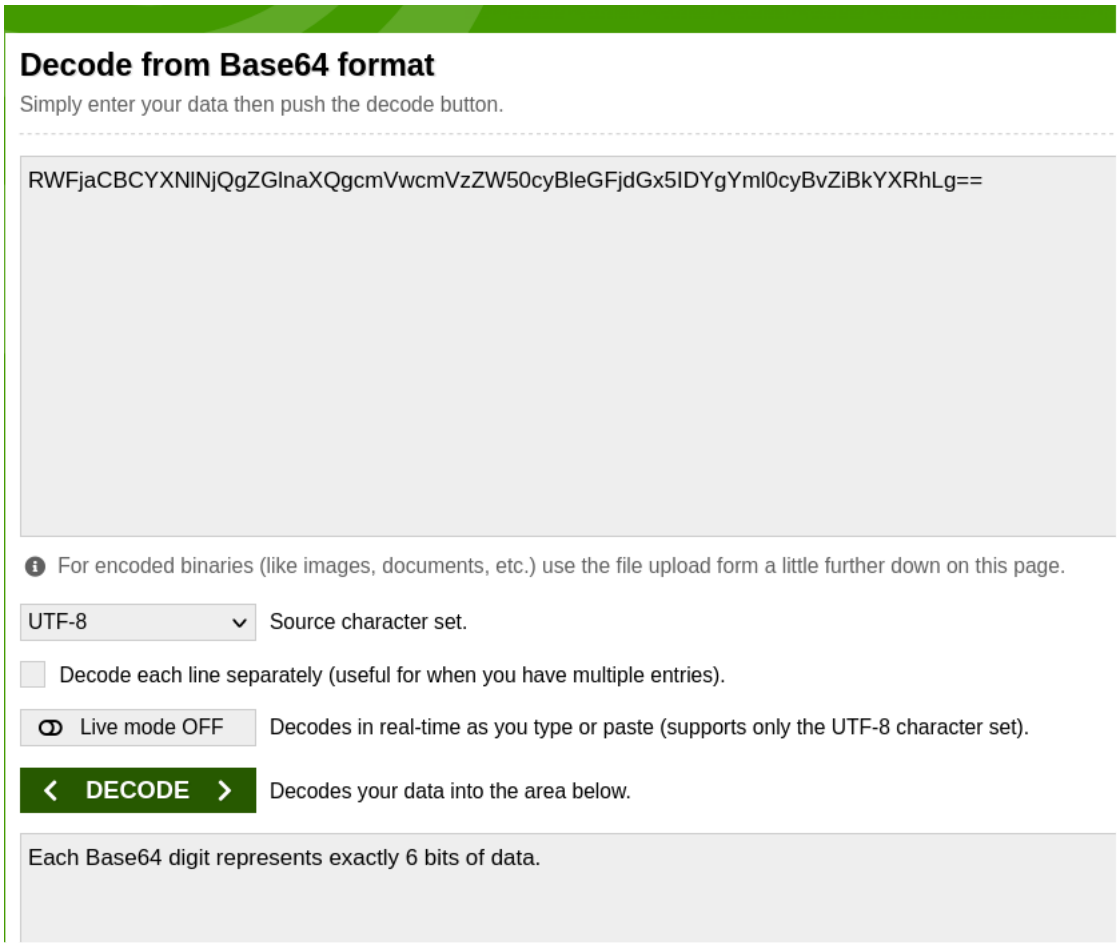
we can decode it by using **terminal.**

second method was

$ echo "MJQXGZJTGIQGS4ZAON2XAZLSEBRW63LNN5XCA2LOEBBVIRRHOM======" | base32 -d
base32 is super common in CTF's

RWFjaCBCYXNlNjQgZGlnaXQgcmVwcmVzZW50cyBleGFjdGx5IDYgYml0cyBvZiBkYXRhLg==

we can solve this in many ways like using of terminal or using of web tools like “base64_decoder”



and the second method was

echo "RWFjaCBCYXNlNjQgZGlnaXQgcmVwcmVzZW50cyBleGFjdGx5IDYgYml0cyBvZiBkYXRhLg==" | base64 --decode
Each Base64 digit represents exactly 6 bits of data.

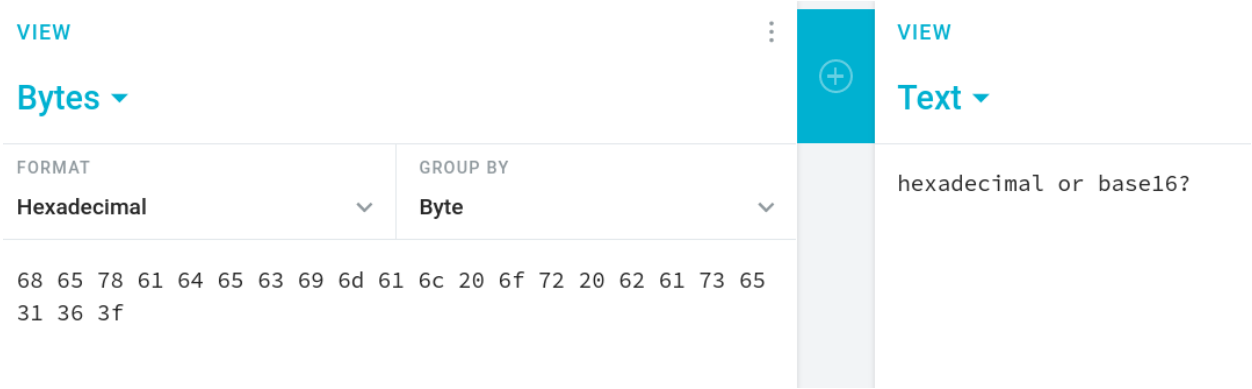
---

68 65 78 61 64 65 63 69 6d 61 6c 20 6f 72 20 62 61 73 65 31 36 3f

the given encoded text is hexadecimal we can use online editor and also we can use terminal.

IN TERMINAL WE HAVE MANY METHODS TO DO THAT.

online decoder method



terminal methods

echo "68 65 78 61 64 65 63 69 6d 61 6c 20 6f 72 20 62 61 73 65 31 36 3f" | xxd -r -p "here -r for retrive and -p for plain"
hexadecimal or base16?

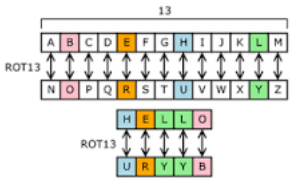
echo "68 65 78 61 64 65 63 69 6d 61 6c 20 6f 72 20 62 61 73 65 31 36 3f" | perl -ne 'print pack("H*", $_)'

hexadecimal or base16?

Ebgngr zr 13 cynprf!

this encryption was ROT13

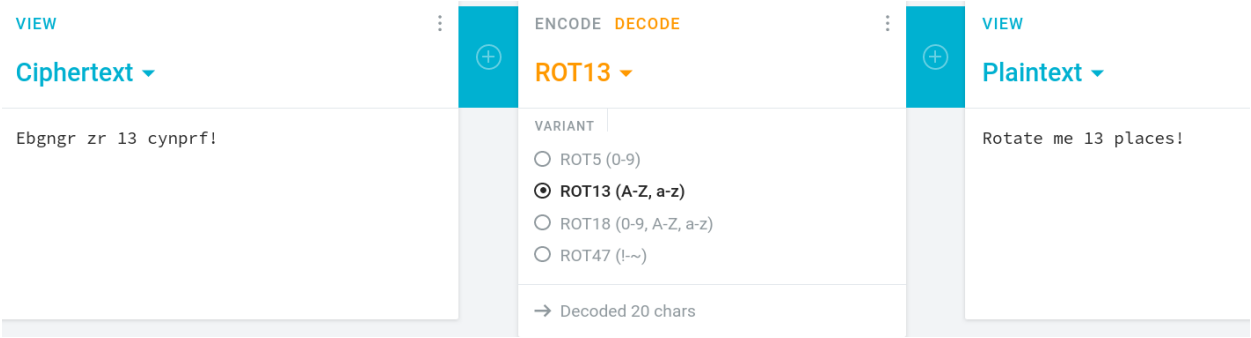
[



ROT13 ("rotate by 13 places", usually hyphenated ROT-13) is a simple Caesar cipher used for obscuring text by **replacing each letter with the letter thirteen places down the alphabet**.

}

we can decode this in both ways online method and terminal method



the second method was terminal method

echo "Ebgngr zr 13 cynprf!" | tr 'A-Za-z' 'N-ZA-Mn-za-m'

Rotate me 13 places

The tr command translates characters using the specified rules.
'A-Za-z' specifies all uppercase and lowercase letters of the alphabet.
'N-ZA-Mn-za-m' specifies the ROT13 mapping, where each letter is shifted 13 positions ahead in the alphabet (A -> N, B -> O, C -> P, ..., Z -> M, a -> n, b -> o, c -> p, ..., z -> m).

***@F DA:? >6 C:89E C@F?5 323J C:89E C@F?5 Wcf E:>6DX**

this encryption was ROT47 the best way to use this is using was to use of online tool “DCODE”.

we can also do with terminla

echo "*@F DA:? >6 C:89E C@F?5 323J C:89E C@F?5 Wcf E:>6DX" | tr '!-~' 'P-~!-O'

You spin me right round baby right round (47 times)

- **. .-.. . -.-. --- -- -- ..- -. .. -.-. .- - .. --- -.**

. -. -.-. --- -.. .. -. --.

this is morse code encryption

we can use online web tool to decrypt this

Morse Code Translater

Note: When translating Morse to Text: For best results, each grouping of morse symbols that translates to a letter should be separated with a space, and each word with two spaces. One space will be eliminated between each morse character, so this will yield natural looking english.

https://mattfedder.com/blog/ham/MorseTranslater

Morse Code Translater

Enter text (or morse code) that you would like to translate, and click 'translate' below.

TELECOMMUNICATION

ENCODING

Translate

Note: When translating Morse to Text: For best results, each grouping of morse symbols that translates to a letter should be separated spaces. One space will be eliminated between each morse character, so this will yield natural looking english.

For example, SOS SINKING should be entered as:

ans: **TELECOMMUNICATION ENCODING**

---

85 110 112 97 99 107 32 116 104 105 115 32 66 67 68

this is decimal encoding

we can decrypt this using online decoder like

Decimal to Text

cryptii v2 is an archived OpenSource web application published under the MIT license where you can convert, encode and decode content between different formats.

https://v2.cryptii.com/decimal/text

MORSECODE - .... . / --.- ..

LEETSPEAK +h3 qv|©|{ br°w2 ƒc

ATBASH ROMAN GSV JFRXP YILDM

CAESAR CIPHER QEB NRFZH YOLTK

ans: **Unpack this BCD**

LS0tLS0gLi0tLS0gLi0tLS0gLS0tLS0gLS0tLS0gLi0tLS0gLi0tLS0gLS0tLS0KLS0tLS0gLi0tLS0gLi0tLS0gLS0tLS0gLS0tLS0gLi

this question was little be trickier

we have to go in a path to get the flag

base64>morse>binary>ROT47>decimal

base64:

- ---- .---- .---- ----- ----- .---- .---- -----
  ----- .---- .---- ----- ----- .---- ----- .----
  ----- ----- .---- ----- ----- ----- ----- -----
  ----- .---- .---- ----- ----- ----- ----- -----
  ----- .---- ----- .---- .---- .---- .---- .----
  ----- .---- .---- ----- ----- ----- ----- -----
  ----- ----- .---- ----- ----- ----- ----- -----
  ----- .---- .---- ----- ----- ----- ----- -----
  ----- .---- .---- ----- ----- ----- ----- -----
  ----- .---- .---- ----- ----- .---- ----- .----
  ----- ----- .---- ----- ----- ----- ----- -----
  ----- .---- .---- ----- ----- ----- .---- -----
  ----- .---- .---- ----- .---- ----- ----- -----
  ----- ----- .---- ----- ----- ----- ----- -----
  ----- .---- .---- ----- ----- ----- ----- -----
  ----- .---- .---- ----- ----- ----- ----- -----
  ----- .---- .---- ----- ----- .---- ----- -----
  ----- ----- .---- ----- ----- ----- ----- -----
  ----- .---- .---- ----- ----- ----- .---- -----
  ----- .---- .---- ----- ----- ----- ----- .----
  ----- ----- .---- ----- ----- ----- ----- -----

----- .---- .---- ----- ----- ----- ----- -----
----- .---- ----- .---- .---- .---- .---- .----
----- .---- .---- ----- .---- ----- ----- -----
----- ----- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- .---- ----- ----- -----
----- .---- .---- ----- ----- .---- .---- -----
----- ----- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- ----- .---- .---- .---- .---- .----
----- .---- .---- ----- ----- .---- .---- -----
----- ----- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- ----- .---- .---- .---- .---- .----
----- .---- .---- ----- ----- ----- ----- -----
----- ----- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- ----- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- .----
----- ----- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- ----- .---- .---- .---- .---- .----
----- .---- .---- ----- ----- ----- ----- -----
----- ----- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- ----- .---- .---- .---- .---- .----
----- .---- .---- ----- ----- ----- ----- -----
----- ----- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- .----
----- ----- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- .---- ----- ----- -----
----- .---- .---- ----- ----- .---- .---- -----
----- ----- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- .----
----- ----- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- .---- .---- .----
----- ----- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- ----- .---- .---- .---- .---- .----
----- .---- .---- ----- ----- .---- ----- -----
----- ----- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- .----
----- ----- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- .---- ----- ----- -----
----- .---- .---- ----- ----- .---- .---- -----
----- ----- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- .----
----- ----- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- ----- .---- .---- .---- .---- .----
----- .---- .---- ----- ----- .---- ----- -----
----- ----- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- .---- ----- .----
----- ----- .---- ----- ----- ----- ----- -----

----- .---- .---- ----- ----- ----- .---- -----
----- .---- .---- ----- ----- ----- ----- .----
----- ----- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- .---- ----- .----
----- ----- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- .---- .----
----- ----- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- ----- .---- .---- .---- .---- .----
----- .---- .---- ----- ----- .---- ----- -----
----- ----- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- .---- ----- ----- -----
----- .---- .---- ----- .---- ----- ----- -----
----- ----- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- ----- .---- .---- .---- .---- .----
----- .---- .---- ----- ----- .---- ----- -----
----- ----- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- ----- .---- .---- .---- .---- .----
----- .---- .---- ----- ----- .---- ----- -----
----- ----- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- ----- .---- .---- .---- .---- .----
----- .---- .---- ----- ----- ----- ----- -----
----- ----- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- .---- .----
----- ----- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- .---- .----
----- .---- .---- ----- ----- .---- ----- .----
----- ----- .---- ----- ----- ----- ----- -----
----- .---- .---- ----- ----- ----- .---- .----
----- .---- .---- ----- ----- .---- ----- .----
----- .---- .---- ----- ----- .---- ----- .----

morse:

01100110
01100101
00100000
01100000
01011111
01100000
00100000
01100000

01100000
01100101
00100000
01100010
01101000
00100000
01100000
01100000
01100100
00100000
01100010
01100001
00100000
01100000
01011111
01101000
00100000
01101000
01100110
00100000
01100000
01011111
01100110
00100000
01100000
01011111
01100000
00100000
01100010
01100001
00100000
01100000
01100000
01100101
00100000
01100000
01011111
01100011
00100000
01100000
01011111
01100100
00100000
01100000
01100000
01100100
00100000
01100010
01100001
00100000
01101000
01100110
00100000
01100010

01100001
00100000
01101000
01100111
00100000
01100000
01011111
01100100
00100000
01100000
01100000
01100101
00100000
01100010
01100001
00100000
01100000
01100000
01100101
00100000
01100000
01100000
01100011
00100000
01100000
01011111
01100100
00100000
01101000
01101000
00100000
01100000
01011111
01100110
00100000
01100000
01011111
01100100
00100000
01100000
01011111
01100000
00100000
01100000
01100000
01100011
00100000
01100011
01100101
00100000
01100011
01100101
00100000

01100011
01100101

binary:

fe `_` `e bh` d ba `_h hf` _f `_` ba `e `_c `_d` d ba hf ba hg `_d ``e ba ``e ``c` _d hh `_f` _d `_` ``c ce ce ce

ROT47:

76 101 116 39 115 32 109 97 107 101 32 116 104 105 115 32 97 32 98 105 116 32 116 114 105 99 107 105 101 114 46 46 46

Decimal:

Let's make this a bit trickier…

Task - 2

spectrogram:

A spectrogram is a visual representation of the spectrum of frequencies of a signal as it varies with time. When applied to an audio signal, spectrograms are sometimes called sonographs, voiceprints, or voicegrams. When the data is represented in a 3D plot they may be called waterfalls.

secretaudio.wav

hint given was "Audacity"

the best practice is to use online tools

Spectrum Analyzer | Academo.org - Free, interactive, education.

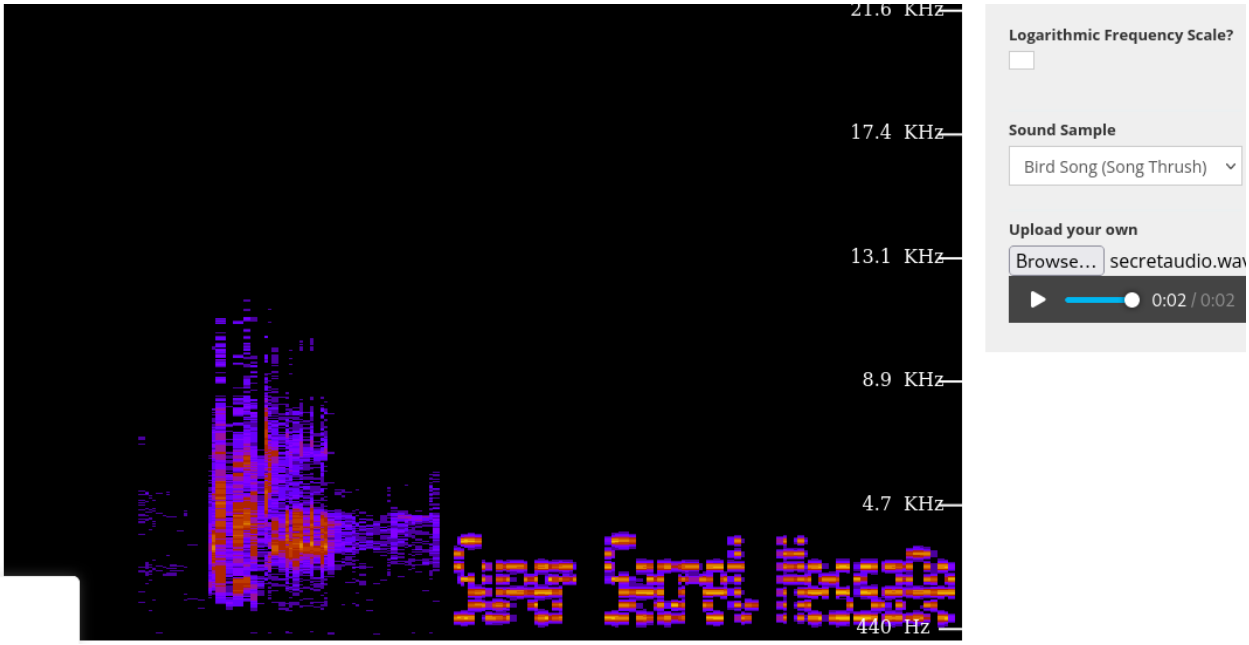
This audio spectrum analyzer enables you to see the frequencies present in audio recordings.

https://academo.org/demos/spectrum-analyzer/

Spectrum Analyzer - Spectrogram - Online Audio File Spectral Analysis

Tool to perform spectral analysis of audio files (WAV, MP3, etc.) and display any hidden data in sound frequencies and their visualization.

https://www.dcode.fr/spectral-analysis



ans: **super secret message**

Task - 3

Steganography:

Steganography is the practice of concealing a file, message, image, or video within another file, message, image, or video.

as it comes under steganography there are many things we have to try like **Zsteg, Steghide, Outguess, ExifTool, Foremost, Strings, PngCheck**

the best method to do steganography is to use online tool

Aperi'Solve

Aperi'Solve is an online platform which performs layer analysis on image. The platform also uses zsteg, steghide, outguess, exiftool, binwalk, foremost and strings for deeper steganography analysis. The platform supports the following images format: .png, .jpg, .gif, .bmp, .jpeg, .jfif, .jpe, .tiff...

https://www.aperisolve.com/

this gonna do all the things which comes under steganography. its the better option.

ans: **SpaghettiSteg**

Task - 4

Security through obscurity:

Security through obscurity is the reliance in security engineering on the secrecy of the design or implementation as the main method of providing security for a system or component of a system.

meme.jpg

getting this answer was so simple just we have to use **strings** tool to retrive the answer.

$ strings meme.jpg

so the answers are

1. Download and get 'inside' the file. What is the first filename & extension?

A) **hackerchat.png**

2. Get inside the archive and inspect the file carefully. Find the hidden text?

A) **AHH_YOU_FOUND_ME!**