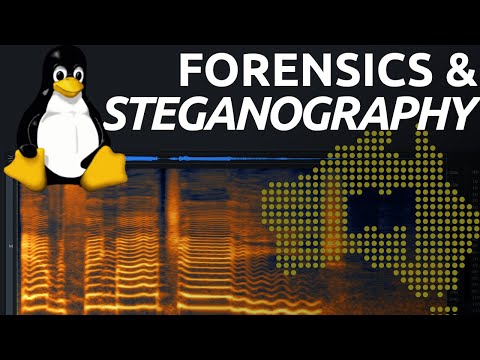
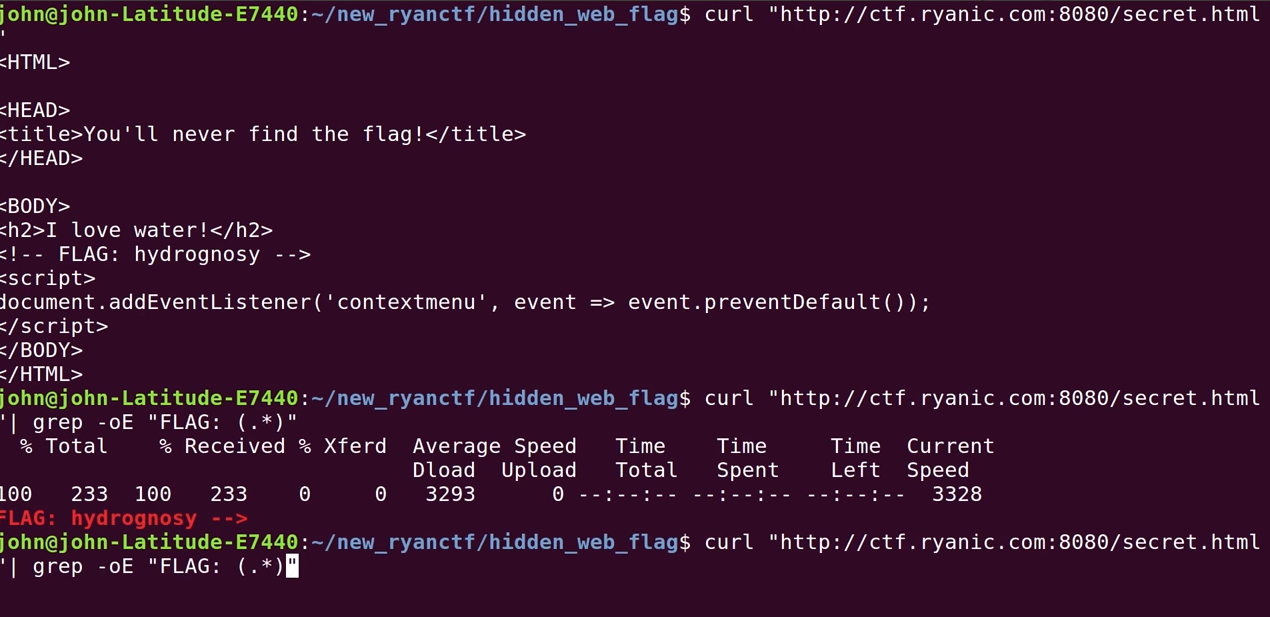
Website (youtube video)

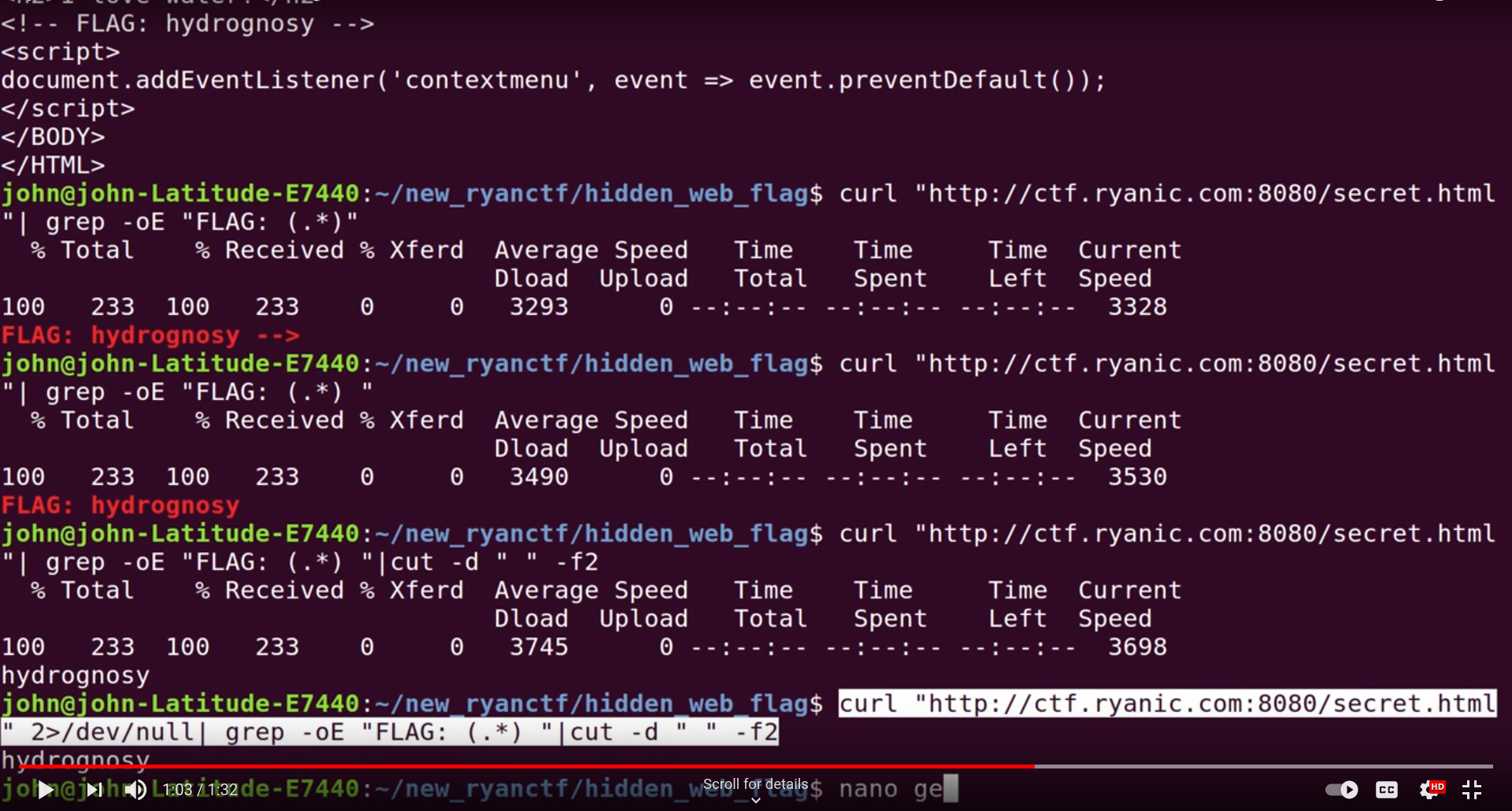
[Steganography (1/2) BsidesCT CTF 2018](https://www.youtube.com/watch?v=sChN6o06jDM)

[Audio Spectrogram & StegCracker Passwords - DownUnderCTF](https://www.youtube.com/watch?v=rAGkm4pv44s&t=433s)









**Image**

**Open image using command line**

sudo apt-get install fim

**fim cat.jpg**

**eog the\_number.png**

file

Exif/ exiftool

xxd

**Janam Kundali**

[Aperi'Solve](https://aperisolve.fr/)

[**https://aperisolve.fr**](https://aperisolve.fr)

binwalk (To see if some file is hidden )

Craig Heffner, ReFirmLabs

<https://github.com/ReFirmLabs/binwalk>

Usage: binwalk [OPTIONS] [FILE1] [FILE2] [FILE3] ...

Signature Scan Options:

-B, --signature Scan target file(s) for common file signatures

-R, --raw=<str> Scan target file(s) for the specified sequence of bytes

-A, --opcodes Scan target file(s) for common executable opcode signatures

-m, --magic=<file> Specify a custom magic file to use

-b, --dumb Disable smart signature keywords

-I, --invalid Show results marked as invalid

-x, --exclude=<str> Exclude results that match <str>

-y, --include=<str> Only show results that match <str>

Extraction Options:

-e, --extract Automatically extract known file types

-D, --dd=<type:ext:cmd> Extract <type> signatures, give the files an extension of <ext>, and execute <cmd>

-M, --matryoshka Recursively scan extracted files

-d, --depth=<int> Limit matryoshka recursion depth (default: 8 levels deep)

-C, --directory=<str> Extract files/folders to a custom directory (default: current working directory)

-j, --size=<int> Limit the size of each extracted file

-n, --count=<int> Limit the number of extracted files

-r, --rm Delete carved files after extraction

-z, --carve Carve data from files, but don't execute extraction utilities

-V, --subdirs Extract into sub-directories named by the offset

Entropy Options:

-E, --entropy Calculate file entropy

-F, --fast Use faster, but less detailed, entropy analysis

-J, --save Save plot as a PNG

-Q, --nlegend Omit the legend from the entropy plot graph

-N, --nplot Do not generate an entropy plot graph

-H, --high=<float> Set the rising edge entropy trigger threshold (default: 0.95)

-L, --low=<float> Set the falling edge entropy trigger threshold (default: 0.85)

Binary Diffing Options:

-W, --hexdump Perform a hexdump / diff of a file or files

-G, --green Only show lines containing bytes that are the same among all files

-i, --red Only show lines containing bytes that are different among all files

-U, --blue Only show lines containing bytes that are different among some files

-w, --terse Diff all files, but only display a hex dump of the first file

Raw Compression Options:

-X, --deflate Scan for raw deflate compression streams

-Z, --lzma Scan for raw LZMA compression streams

-P, --partial Perform a superficial, but faster, scan

-S, --stop Stop after the first result

General Options:

-l, --length=<int> Number of bytes to scan

-o, --offset=<int> Start scan at this file offset

-O, --base=<int> Add a base address to all printed offsets

-K, --block=<int> Set file block size

-g, --swap=<int> Reverse every n bytes before scanning

-f, --log=<file> Log results to file

-c, --csv Log results to file in CSV format

-t, --term Format output to fit the terminal window

-q, --quiet Suppress output to stdout

-v, --verbose Enable verbose output

-h, --help Show help output

-a, --finclude=<str> Only scan files whose names match this regex

-p, --fexclude=<str> Do not scan files whose names match this regex

-s, --status=<int> Enable the status server on the specified port

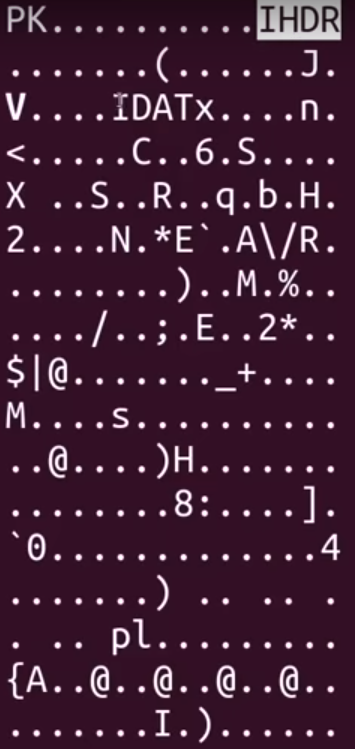
**When you fail to extract using binwalk –e**

**binwalk - - dd=”.\* ” filename.jpg**

**Forced**

**binwalk - M - -dd=.\***

**How to identify png file ??**

r

*PK, IDAT*

foremost

foremost filename.jpg

**Meta data**

identify –verbose filename

Zsteg (ONLY FOR PNG)

zsteg garden2.png

zsteg –a filename.png

strings

Strings filename |less

Change type of file

cp Orignal.jpeg Newfile.png

Site: metapicz, imago

<http://metapicz.com/#landing>

Gcode to graph/photo

<https://nraynaud.github.io/webgcode/>

Photo edit

<https://stegonline.georgeom.net/>

“There’s something called **8 bit plane**”

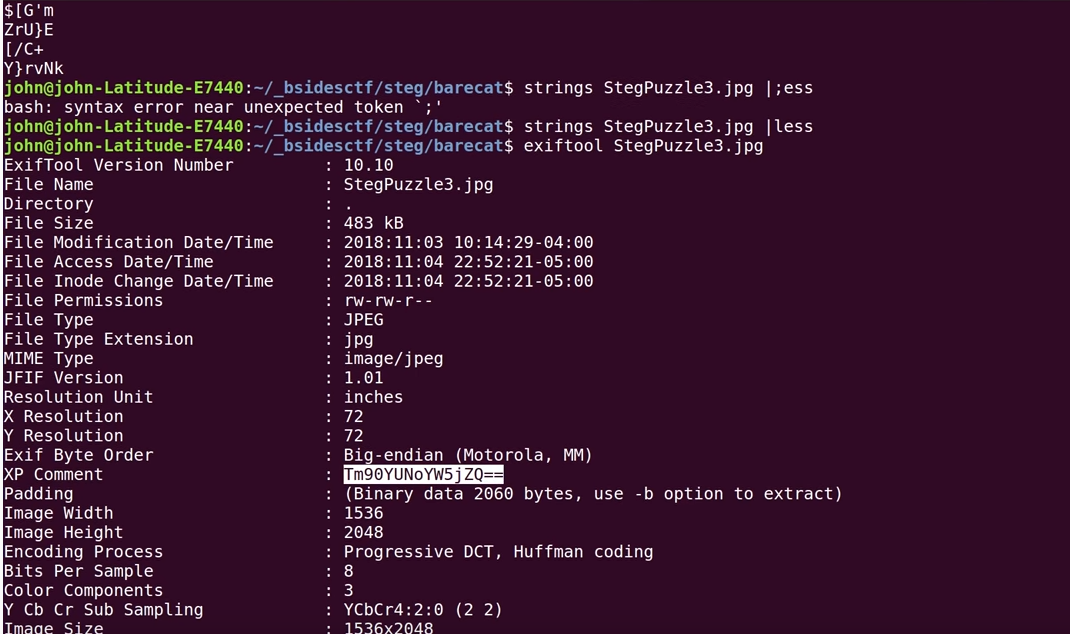
* [stegsolve](https://github.com/zardus/ctf-tools/blob/master/stegsolve/install)
  + wget "https://github.com/eugenekolo/sec-tools/raw/master/stego/stegsolve/stegsolve/stegsolve.jar"
  + java -jar stegsolve.jar

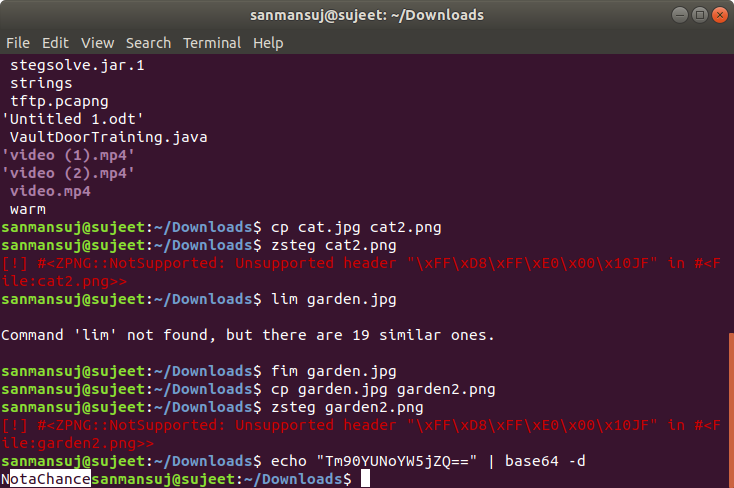
locate stegsolve

cp /home/sanmansuj/bin/stegsolve.jar .

java -jar stegsolve.jar

XP COMMENT





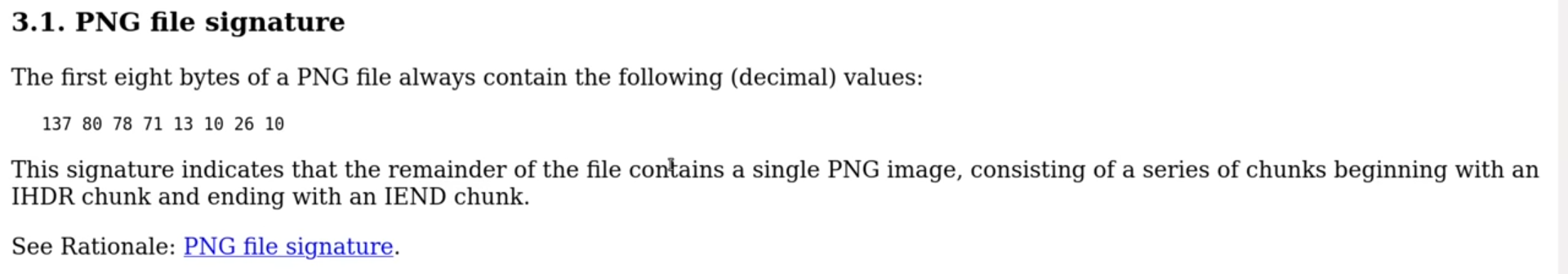
steghide extract –sf filename.jpeg

Enter passphrase :

steghide extract -sf imageFilename.jpeg -p EnterYourPassword

**hexedit**

# Magic Number and png


WEBSITE

INSPECT ELEMENT – html, css, js

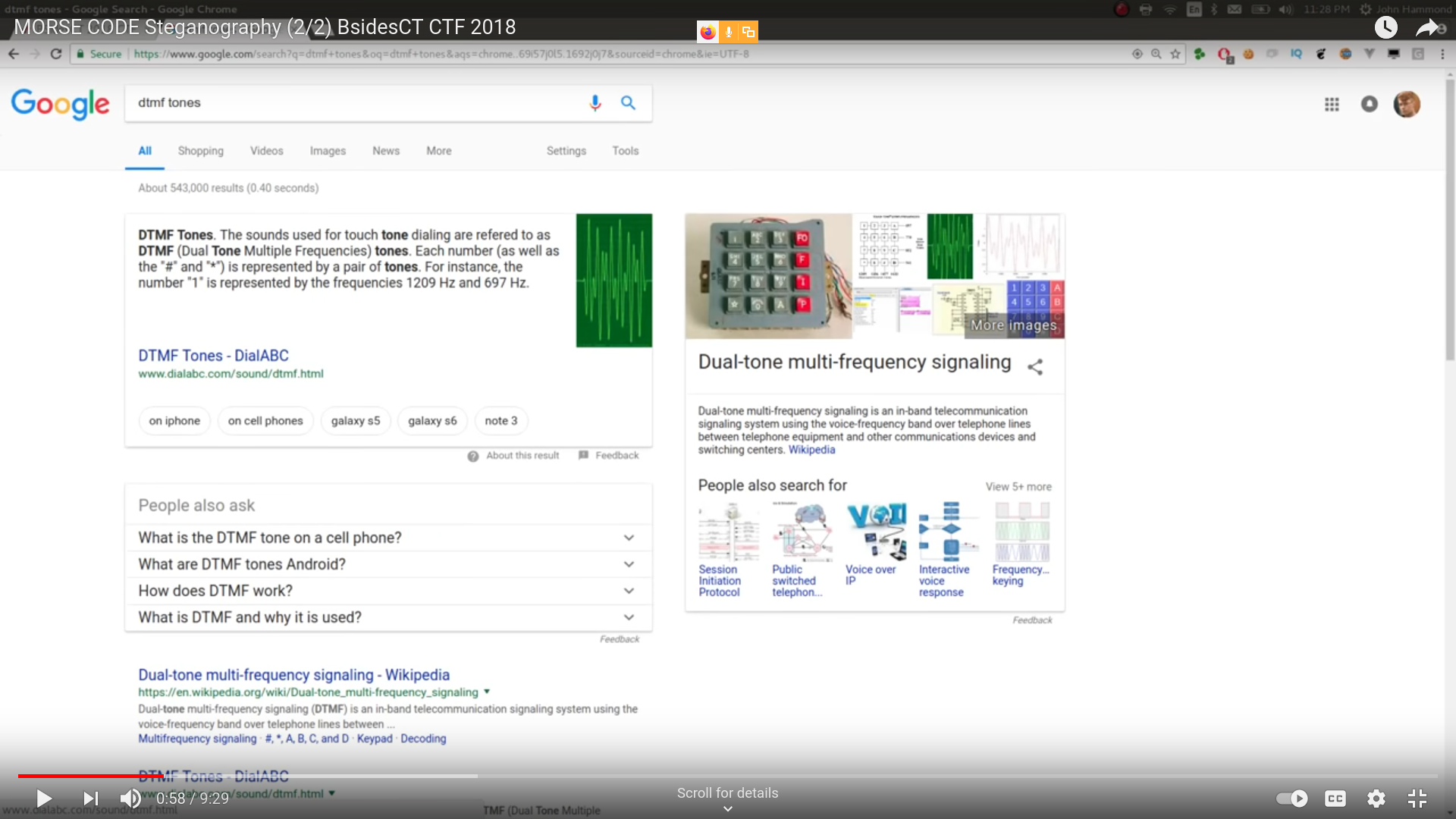
Source code- urllink

‘curl’ command

AUDIO

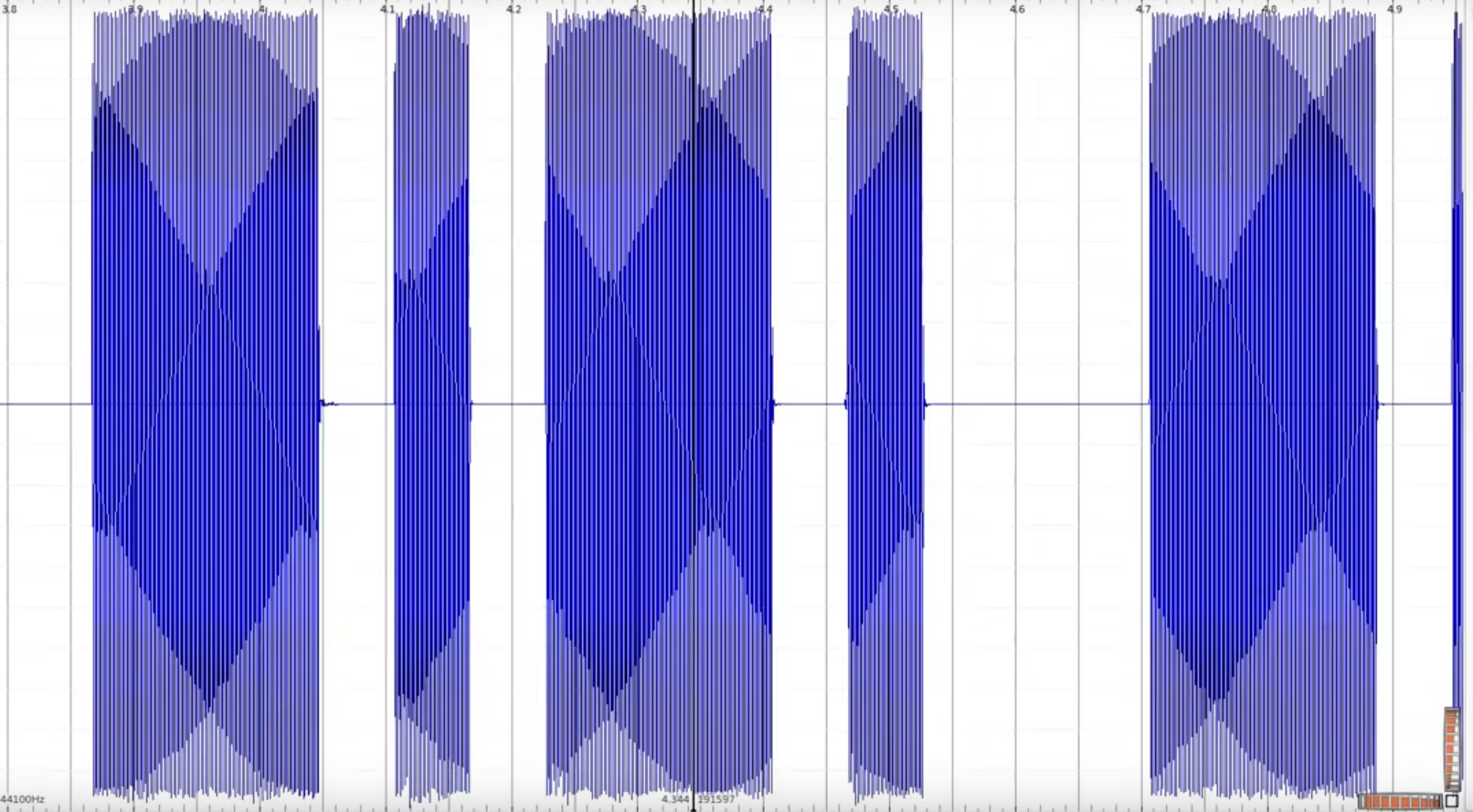
mplayer filename.mp3

Touch tone/ dial pad



**Morse code**- ( analysed using the sonic visualizer)

* some waves – long, some- short;
* same pattern within
* Same tone beeping over and over
* Use a decoder after that



<https://morsecode.world/international/decoder/audio-decoder-adaptive.html>



<BT>- Break or ‘=’

**Encryption and decryption**

**Find**

[**https://www.bjoxentriq.com/code-breaking/cryptogram**](https://www.bjoxentriq.com/code-breaking/cryptogram)

[Decrypt a Message - Cipher Identifier - Online Code Recognizer (dcode.fr)](https://www.dcode.fr/cipher-identifier)

Chinese to Unicode

<https://pages.ucsd.edu/~dkjordan/resources/unicodemaker.html>

Hex to ascii/unicode/binary/ etc.

<https://www.rapidtables.com/convert/number/hex-to-ascii.html>

Hex to ascii

<https://onlinehextools.com/convert-hex-to-ascii>

**Base 32 Identification**

**Lots of cap letter**

**To convert base 64 to ascii-**

**echo "Tm90YUNoYW5jZQ==" | base64 -d**

**ROT 47- has alphabets, digits, special characters(common symbols)**

<BT> (break) or ‘=’

**Command line**, {use PYTHON}

hex(int ('100010',2))

bytearray.fromhex("7368616b6564").decode()

bytes.fromhex('7368616b6564').decode('utf-8')

Slice [START, -1]

Find flag in text

strings strings | grep pico

Code With Rockstars

<https://codewithrockstar.com/>

File

Make it executeable

chmod +x file name

Extract

7z e something.7z

Wireshark (pcap)

strings file.pcap | grep -i “key”

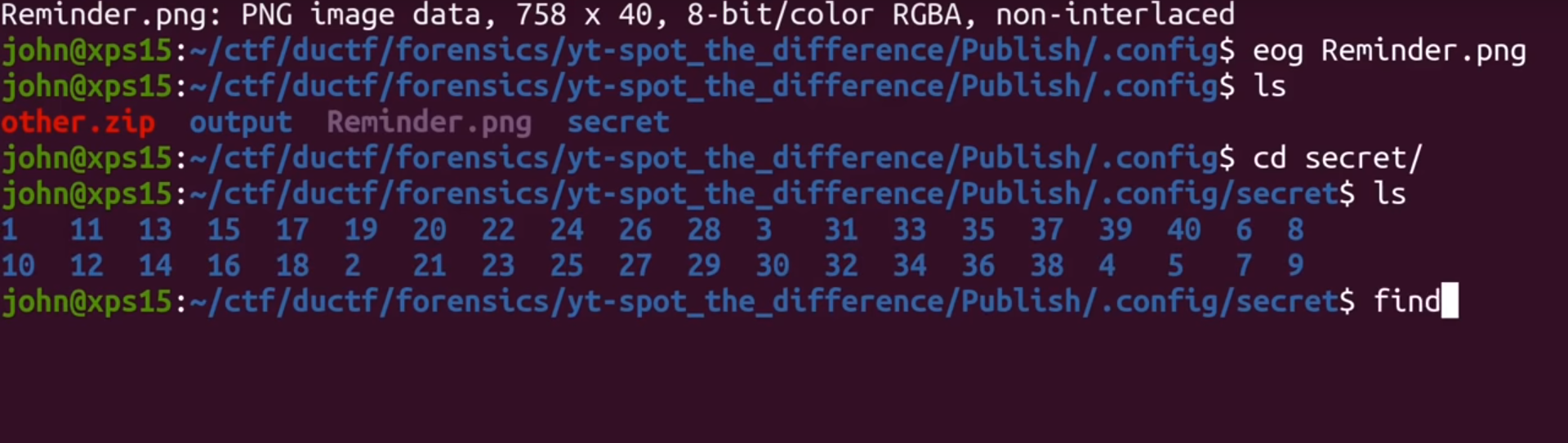
search a word

egrep 'mellon' mysampledata.txt

“**imacros**”

# To see files Nested Directory

find



find | xargs cat

find | xargs cat | grep StringYouWantToFind

# BruteForce

stegcracker <file> [<wordlist>]

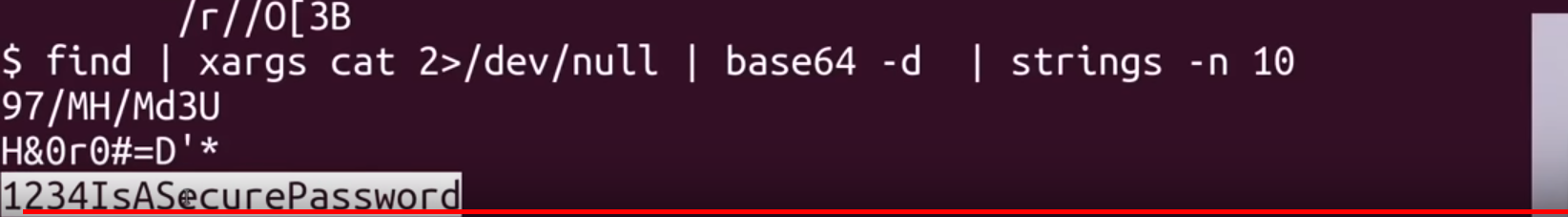
Bash command

basic while loop to read names of all the files in a directory

done

end of the loop





**Resources to be looked up later:**

<https://github.com/JohnHammond/ctf-katana>

**Wire Shark**

Use wireshark

In filters you can put [ tcp contains “picoCTF” to search for flag]

Right click on ‘stream’ and then ‘follow’ and then ‘streams’ to access streams for a particular packet

Check for the last letter, See the destination and source

Capture using python scapy

#Import all the functions from this lib

from scapy.all import \*

pcap = rdpcap ("capture.pcap")

print (pcap[UDP])

flag=""

for p in pcap[UDP]:

try:

if(p[IP].src == "10.0.0.2" and p[IP].dst == "10.0.0.12"):

p[IP].show()

flag += (p[Raw].load)

except IndexError:

continue

print (flag)

**RSA**

Pow(x,y,z)=> z: modulus

Example code: angstom ctf(RSA)

from Crypto.Util.number import inverse

n=113138904645172037883970365829067951997230612719077573521906183509830180342554841790268134999423971247602095979484887092205889453631416247856139838680189062511282674134361726455828113825651055263796576482555849771303361415911103661873954509376979834006775895197929252775133737380642752081153063469135950168223

p=11556895667671057477200219387242513875610589005594481832449286005570409920461121505578566298354611080750154513073654150580136639937876904687126793459819369

q=9789731420840260962289569924638041579833494812169162102854947552459243338614590024836083625245719375467053459789947717068410632082598060778090631475194567

c=108644851584756918977851425216398363307810002101894230112870917234519516101802838576315116490794790271121303531868519534061050530562981420826020638383979983010271660175506402389504477695184339442431370630019572693659580322499801215041535132565595864123113626239232420183378765229045037108065155299178074809432

phi = ( p - 1) \* ( q - 1 )

e=65537

d = inverse(e,phi)

m= pow ( c, d, n)

print(m)

print('hi')

print(hex(m)[2:-1].decode('hex'))

Open the current directory using cmd

nautilus .

Binary Explotataion

Shellstrorm