TOP STEGANOGRAPHY METHODS

These modern methods offer new avenues for covert communication

the art of concealing messages within other messages or media

HADESS

WWW.HADESS.IO

YOUR HEART, A BIRD'S HEART,
BUT YOUR SKIN, A LION'S SKIN,
LET GO OF YOUR LIVING BODY,
O BIRD, SPREAD YOUR WINGS.

POET IRAJ JANNATI ATAI

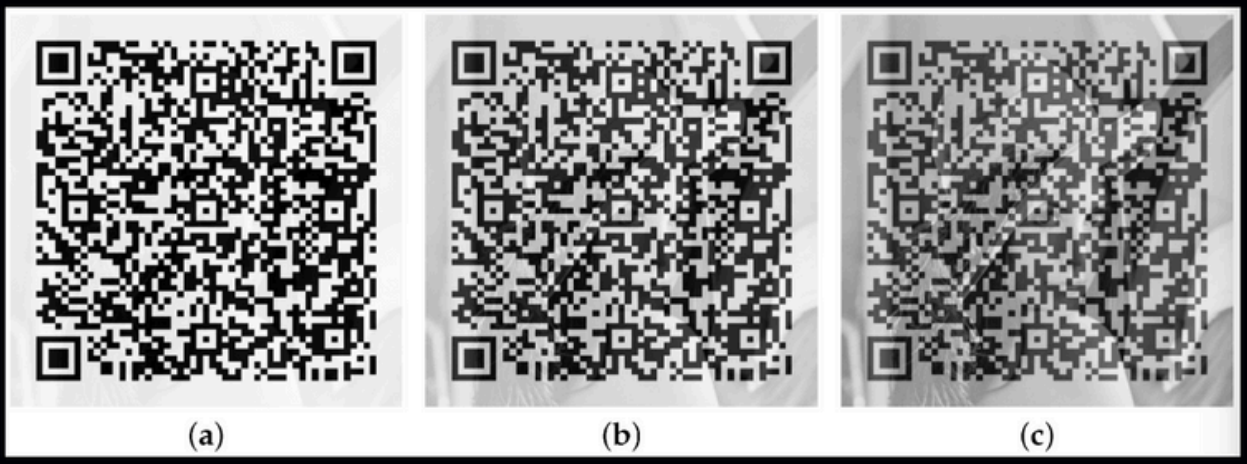
TABLE OF CONTENT

Top Stenography Methods

Steganography, the art of concealing messages within other messages or media, encompasses a variety of techniques both ancient and modern. Historical methods include invisible ink, microdots, and null ciphers, all designed to hide information in plain sight. These methods often relied on physical manipulation of materials or clever linguistic tricks to encode and decode messages, serving purposes ranging from espionage to personal communication.

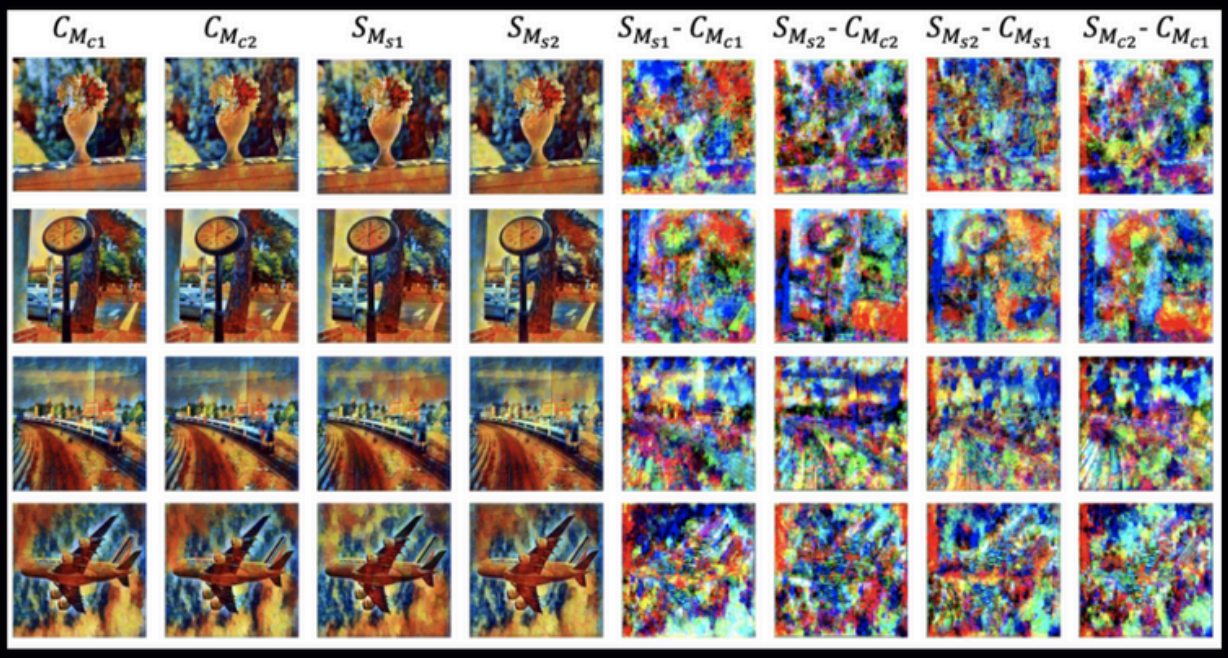
In the digital age, steganography has adapted to the realm of digital files, with techniques such as digital steganography and acoustic steganography. Digital steganography hides messages within images, audio, or video files by subtly altering their data, while acoustic steganography conceals information within audio signals. These modern methods offer new avenues for covert communication, digital watermarking, and copyright protection in an era where information security is paramount.

QR CODES



To convert binary codes to QR codes, you can use online tools such as dCode and Inlite Research's Online Barcode Reader. First, visit the dCode website at https://www.dcode.fr/binary-image and input your binary data. Then, follow the instructions to generate a QR code containing your binary message. Similarly, you can use the Online Barcode Reader provided by Inlite Research at https://online-barcode-reader.inliteresearch.com/. Upload your binary image to the website and let the tool decode it into a QR code for you. These tools offer convenient ways to convert binary data into QR codes, facilitating easy sharing and scanning of information.

Image Transformations



To perform image transformations and steganography operations on images, you can use the "Stegsolve" tool, which is a Java-based application. To use it, first, ensure you have Java installed on your system. Then, download the "Stegsolve.jar" file. Once downloaded, navigate to the directory where the "Stegsolve.jar" file is located using the command line. Then, execute the following command:

```
java -jar Stegsolve.jar
```

This command will launch the Stegsolve application, allowing you to apply various transformations and analyze images for hidden information. Stegsolve provides a graphical interface where you can load images and apply different color plane transformations, bit plane slicing, and other operations to uncover hidden data. It's a powerful tool often used in digital forensics, cryptography, and steganography analysis, offering insights into the structure and content of images beyond what is visible to the naked eye.

Files Strings

```
┌──[✗]─[root@parrot]─[~/Desktop]
└──╼ #binwalk -e patrick.jpg

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
0             0x0             PC bitmap, Windows 3.x format,, 450 x 599 x 24

WARNING: Extractor.execute failed to run external extractor 'unstuff '%e'': [Errno 2] No such file or directory: 'unstuff': 'unstuff'
451386        0x6E33A         StuffIt Deluxe Segment (data): fN`aEWXDVWCUVASTP_aN]_MZ\;HJ>JLGSUAMOANPDQS@MO8EG>KMDQS<IK?LN8EG:GI7DF8EG9FH3@B6CE5CB6DC
7ED3A@2>>7CC7CC.::/;;0<<1==1==0::.88-77-

WARNING: Extractor.execute failed to run external extractor 'unstuff '%e'': [Errno 2] No such file or directory: 'unstuff': 'unstuff'
493337        0x78719         StuffIt Deluxe Segment (data): f?QR;MNDSUHWYAPR>MOAPRETVDSU?NP@MOBOQANP;HJCQP<JI=KJ=KJ5CB5CB:HG9GF8DD4@@5AA1==5AA4@@8DD
```

To analyze a file for hidden content or embedded data, you can use the command-line tools "binwalk" and "strings." These tools are particularly useful for examining binary files, firmware images, and executable files. Here's how you can use them:

1. **Binwalk**: Binwalk is a tool designed to search binary files for embedded files and executable code. It can identify and extract various types of files embedded within the input file. To use binwalk, open a terminal and navigate to the directory containing the file you want to analyze. Then, run the following command:

```
binwalk -e <file>
```

Replace `<file>` with the path to the file you want to analyze. The `-e` option tells binwalk to extract any files it finds within the input file. After running the command, binwalk will scan the file for embedded content and extract any identified files to a directory named `"_<filename>.extracted_"` in the current directory.

2. **Strings**: The "strings" command extracts printable characters from a binary file. It's useful for identifying human-readable text, such as strings, within a binary file. To use "strings," open a terminal and navigate to the directory containing the file you want to analyze. Then, run the following command:

```
strings <file>
```

Replace `<file>` with the path to the file you want to analyze. This command will print any printable strings found within the file to the terminal. It's helpful for quickly identifying text-based content embedded within a binary file, such as plaintext passwords, error messages, or other informative strings.

WAV/* Steg and Bruteforce

```
root@raypzvm:~/stegbrute# stegbrute -f image.jpg -w wordlist.txt -x results.txt
============================================================
    ____  _                ____                _
   / ___|| |_ ___  __ _  |  _ \ _ __ _   _| |_ ___
   \___ \| __/ _ \/ _` | | |_) | '__| | | | __/ _ \
    ___) | ||  __/ (_| | |  _ <| |  | |_| | ||  __/
   |____/ \__\___|\__, | |_| \_\_|   \__,_|\__\___|
                  |___/

StegBrute v0.1.1 - By R4yan
https://github.com/R4yGM/StegBrute

exist
Bruteforcing the file 'image.jpg' with the wordlist 'wordlist.txt' using 3 threads
(thread-0) Failed to crack the file, finished the passwords 202.69ms
(thread-2) Failed to crack the file, finished the passwords 307.14ms
password try: cool123 - Success
File extracted!
Password: cool123
Results written in: results.txt
Tried passwords : 61
Successfully cracked in 530.10ms
============================================================
```

To guess the password for a file embedded within another file, you can use the steg_brute.py script along with a wordlist. Here's how you can do it:

1. **Download and Navigate**: First, download the steg_brute.py script or ensure it's available on your system. Then, navigate to the directory containing the script and the wordlist you want to use.

2. **Run the Command**: Open a terminal and run the following command:

```
./steg_brute.py -b -d /usr/share/wordlists/rockyou.txt -f
../meow.wav
```

or

```
java -jar turgen.jar
```

File in File

```
root@raypzvm:~/stegbrute# stegbrute -f image.jpg -w wordlist.txt -x results.txt
============================================================
    ____  _             ____             _
   / ___|| |_ ___  __ _| __ ) _ __ _   _| |_ ___
   \___ \| __/ _ \/ _` |  _ \| '__| | | | __/ _ \
    ___) | ||  __/ (_| | |_) | |  | |_| | ||  __/
   |____/ \__\___|\__, |____/|_|   \__,_|\__\___|
                  |___/

StegBrute v0.1.1 - By R4yan
https://github.com/R4yGM/StegBrute

exist
Bruteforcing the file 'image.jpg' with the wordlist 'wordlist.txt' using 3 threads
(thread-0) Failed to crack the file, finished the passwords 202.69ms
(thread-2) Failed to crack the file, finished the passwords 307.14ms
password try: cool123 - Success
File extracted!
Password: cool123
Results written in: results.txt
Tried passwords : 61
Successfully cracked in 530.10ms
============================================================
```

To extract a file embedded within another file, you can use the "steghide" tool along with appropriate commands. Here's how you can do it:

1. **Check Embedded File Information**: Before extracting the file, you may want to gather information about the embedded file, such as its size and encryption details. To do this, run the following command in the terminal:

```
steghide info <filename> -p <password>
```

Replace `<filename>` with the name of the carrier file containing the embedded file, and `<password>` with the password required to extract it. This command will provide you with information about the embedded file, such as its size and encryption method.

2. **Extract the Embedded File**: Once you have the necessary information, you can proceed to extract the embedded file. Use the following command:

```
steghide extract -sf <filename> -p <password>
```

Braille

Line 1:	
Line 2:	
Line 3:	
Line 4:	
Line 5:	
Line 6:	
Line 7:	

TABLE 1: 63 Braille Characters With Dots.

Braille steganography is a technique that involves concealing messages within Braille characters. Braille, a tactile writing system used by people who are visually impaired, consists of raised dots arranged in a grid. By manipulating the arrangement of these dots, it's possible to encode hidden messages within Braille text.

To encode and decode messages using Braille steganography, you can use online Braille translators such as the one available at https://www.branah.com/braille-translator. Here's how you can use it:

1. Encoding:

   * Open the Braille translator tool in your web browser.
   * Type or paste the message you want to encode into the input box.
   * The tool will automatically convert the text into Braille characters.
   * To encode a hidden message, you can manipulate the arrangement of dots within the Braille characters manually. For example, you can slightly alter the position or spacing of dots to represent a binary code that encodes the hidden message.
   * Once you've encoded the message, you can copy the modified Braille text for further use.

TTF

TTF-Steganography

A pure-python module to apply Least Significant Bit(s) Steganography to TrueType font files

Hide binary data to a file:

```
ttf_file = LSB(ttf_path)
ttf_file.hide(your_binary_data, path_to_hide)
```

Recover binary data from a file:

```
ttf_file = LSB(ttf_path)
your_binary_data = ttf_file.recover()
```

The module uses LSB to hide the data, further methods like Value Differencing will be added soon. If you want to hide more than one bit per coordinate, use the parameter 'change':

```
ttf_file.hide(your_binary_data, path_to_hide, change=3)
```

Now the last three bits will be changed, which shouldn't be detectable to the human eye.

If you need to hide/recover more than 1MB (or 2^23 bits), set META_DATA_LENGTH higher.

To perform Fast Fourier Transform (FFT) on an image, and potentially extract information from it, you can use the online tool provided at http://bigwww.epfl.ch/demo/ip/demos/FFT/. This tool enables you to analyze the frequency components of an image, which can sometimes reveal hidden information or patterns.

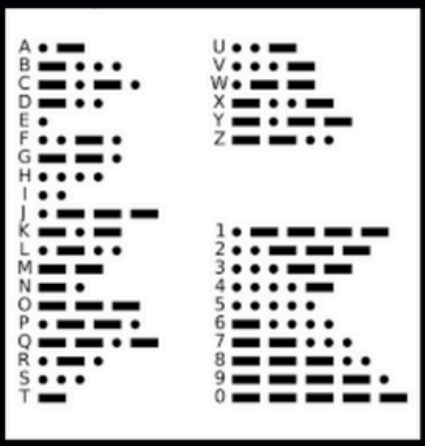
Here's how you can utilize it:

1. Access the Tool:
    * Open your web browser and navigate to the provided URL: http://bigwww.epfl.ch/demo/ip/demos/FFT/.
2. Upload the Image:
    * Click on the "Choose File" or similar button to upload the image you want to analyze.

BRAINFUCK

TO TRANSLATE BRAINFUCK CODE, YOU CAN USE THE ONLINE BRAINFUCK INTERPRETER PROVIDED AT HTTPS://WWW.DCODE.FR/BRAINFUCK-LANGUAGE. THIS TOOL ALLOWS YOU TO WRITE BRAINFUCK CODE, OR INPUT EXISTING CODE, AND EXECUTE OR TRANSLATE IT TO OBTAIN THE CORRESPONDING OUTPUT.

Morse Code

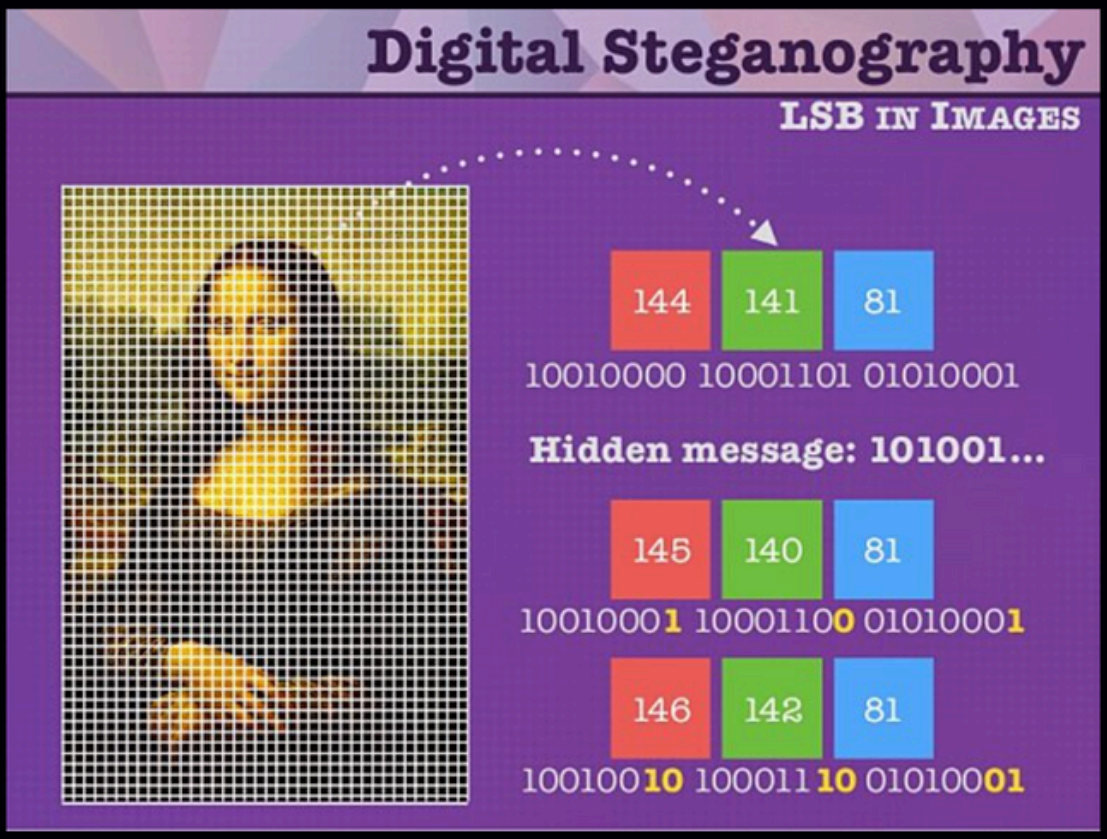


To translate Morse code, you can use the online Morse code translator provided at https://www.boxentriq.com/code-breaking/morse-code. This tool allows you to input Morse code and obtain the corresponding text, or input text and obtain the Morse code representation.

Here's how you can use it:

1. Access the Tool:
   * Open your web browser and navigate to the provided URL: https://www.boxentriq.com/code-breaking/morse-code.
2. Translate Morse Code to Text:
   * To translate Morse code to text, enter the Morse code sequence into the provided input box.
   * Once entered, the tool will automatically translate the Morse code into readable text.
3. Translate Text to Morse Code:
   * Alternatively, if you want to translate text into Morse code, input the text into the provided input box.
   * The tool will then convert the text into its corresponding Morse code representation.
4. Analyze Results:
   * After translation, review the output to ensure accuracy and completeness.
   * Pay attention to any messages or information conveyed by the Morse code translation.

LSB HALF

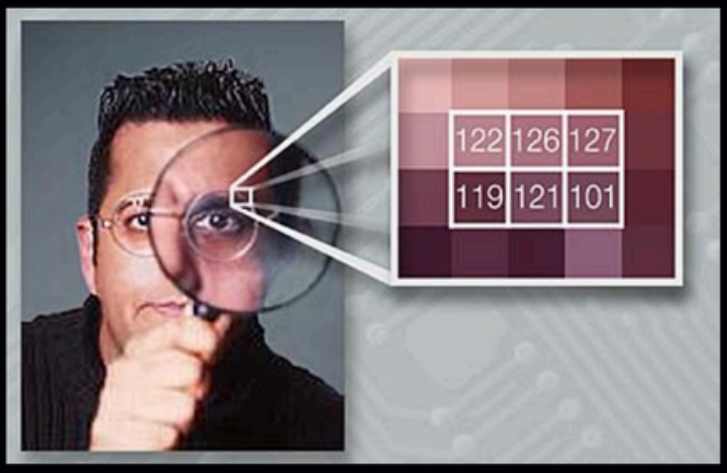


To view the LSB (Least Significant Bit) HALF mode of an image, you can use the online tool available at https://georgeom.net/StegOnline/image. This tool allows you to analyze the least significant bits of an image, which can reveal hidden information or alterations made to the image through steganography techniques.

Here's how you can use it:

1. Access the Tool:
   * Open your web browser and navigate to the provided URL: https://georgeom.net/StegOnline/image.
2. Upload the Image:
   * Click on the "Choose File" or similar button to upload the image you want to analyze.
3. Select LSB HALF Mode:
   * Once the image is uploaded, locate the option to select the LSB mode.
   * Choose the "LSB HALF" mode from the available options.

Digital Watermarking



OpenStego offers two primary modes of operation: data hiding and watermarking.

Data Hiding: In this mode, users can embed files within images or extract hidden data from images. The process is straightforward and involves concealing sensitive information within the pixels of an image, making it imperceptible to the human eye.

Watermarking (beta): Watermarking mode allows users to embed invisible signatures within images for authentication and ownership verification purposes. Users can generate signature files and use them to watermark images or verify watermarks later. OpenStego ensures robust digital watermarking capabilities, making the watermark resistant to common image modifications like resizing or cropping.

Digital watermarking is the process of embedding covert markers within noise-tolerant signals, such as image data, to identify ownership or authenticate the signal's integrity. These markers, though imperceptible to humans, serve to trace copyright infringements and authenticate documents like banknotes. OpenStego implements digital watermarking with robustness, ensuring that the watermark remains intact even after minor modifications to the watermarked image.

For more information and to access OpenStego, visit https://www.openstego.com/ .

Cryptography

Cryptography in steganography involves the use of cryptographic techniques to enhance the security and concealment of hidden information within cover files. While steganography alone aims to hide the existence of a secret message, cryptography ensures that even if the hidden message is discovered, it remains unintelligible without the appropriate decryption key.

By combining steganography with cryptography, practitioners can achieve higher levels of security and privacy for their communications. Here's how cryptography is applied in steganography:

1. Encryption of Hidden Data:
    * Before embedding the data into a cover file, it is encrypted using cryptographic algorithms such as AES (Advanced Encryption Standard), DES (Data Encryption Standard), or RSA (Rivest-Shamir-Adleman).
    * Encryption ensures that the hidden message remains secure even if the cover file is intercepted or subjected to unauthorized access.
2. Decryption Key:
    * To extract the hidden data from the cover file, the recipient needs the decryption key.
    * The decryption key is a piece of secret information that is used to decrypt the encrypted data and reveal the original message.
3. Key Management:
    * Effective key management practices, such as key generation, distribution, and storage, are crucial for ensuring the security of the hidden data.
    * Key management protocols may include techniques like key exchange using asymmetric cryptography or key derivation from a passphrase.
4. Authentication and Integrity:
    * Cryptographic techniques can also be used to ensure the authenticity and integrity of the hidden data.
    * Digital signatures or message authentication codes (MACs) can be embedded within the cover file to verify the origin and integrity of the hidden message.
5. Hybrid Approach:
    * A hybrid approach combining steganography and cryptography leverages the strengths of both techniques to achieve a balance between security and stealthiness.
    * By embedding encrypted data within cover files using steganographic methods, practitioners can achieve high levels of security while maintaining covert communication channels.

Splited Files

List of All Tools

List of all tools available online on dCode, classified by category (click to expand). To be used in addition to the search bar which allows you to find tools by keywords.
Reminder: dCode offers an encryption identifier !

- ▶ Games and Solvers
- ▼ Cryptography
  - ▶ Cryptanalysis
  - ▶ Modern Cryptography
  - ▶ Poly-Alphabetic Cipher
  - ▼ Transposition encryption
    - Digit ADFGVX
    - ADFGX cipher
    - AMSCO figure
    - Caesar's Square Cipher
    - Double Transposition Cipher
    - Figure Rail Fence (Zig-Zag)
    - Redefence Figure
    - Scytale Cipher
    - Ubchi cipher
    - Cipher by Spiral Writing
    - Skip Cipher

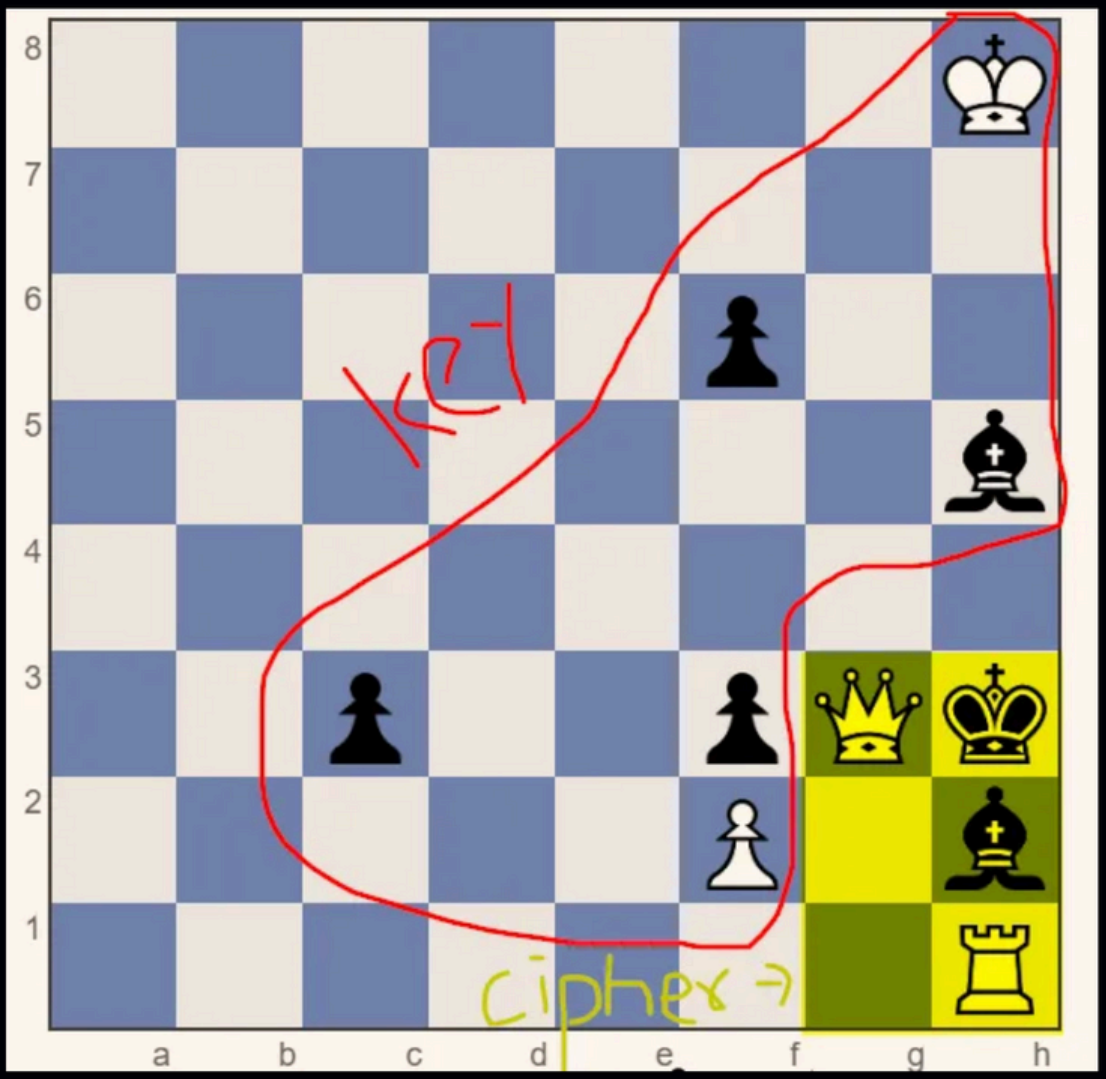
Splitting files into smaller parts, also known as "file fragmentation" or "file splitting," is a technique often used in steganography to conceal data within cover files more effectively. By splitting a large file into smaller chunks, the steganographer can distribute the hidden data across multiple locations within the cover file, making it more difficult for unauthorized parties to detect or extract the concealed information.

Here's how you can implement file splitting techniques in steganography:

1. Splitting the Data:

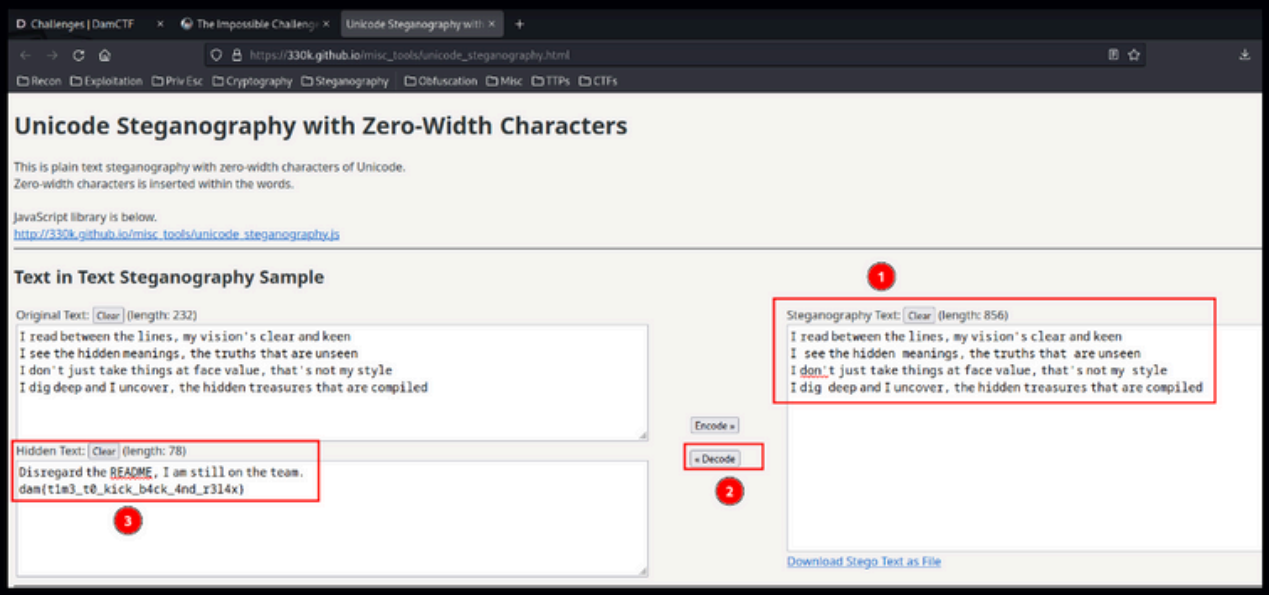
   * Before embedding the data into the cover file, the original file is split into smaller parts using file splitting algorithms or tools.
   * The splitting process divides the data into chunks of equal or variable sizes, depending on the specific requirements and constraints of the steganographic application.

Key and Cipher alongside



Steganography often employs cryptographic techniques alongside key and cipher systems to enhance the security and concealment of hidden messages within cover files. This integration allows for a more robust approach to hiding information, making it challenging for unauthorized parties to detect or decipher the concealed data. Let's explore how key and cipher systems are utilized alongside steganography techniques:

Unicode



NICODE IS A UNIVERSAL CHARACTER ENCODING STANDARD THAT ASSIGNS A UNIQUE NUMBER TO EVERY CHARACTER ACROSS DIFFERENT WRITING SYSTEMS AND LANGUAGES. THIS VAST CHARACTER SET INCLUDES NOT ONLY LETTERS, DIGITS, AND SYMBOLS BUT ALSO VARIOUS CONTROL CHARACTERS AND SPECIAL FORMATTING ELEMENTS.

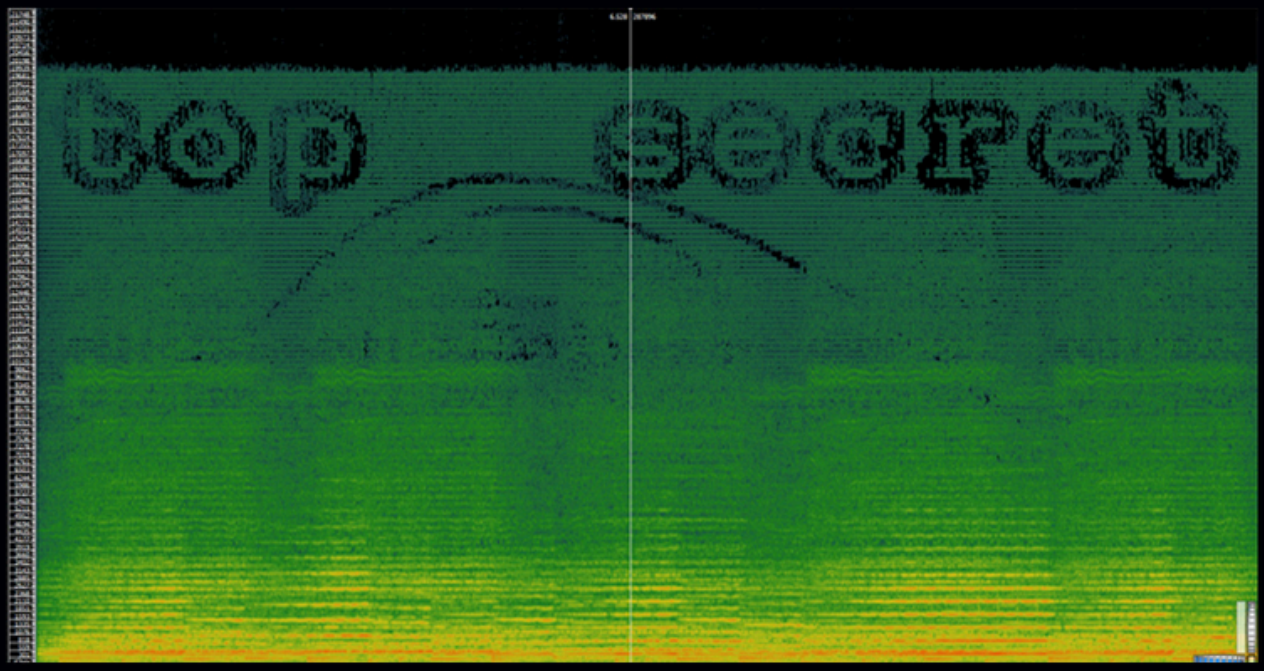
Unicode Steganography

UNICODE STEGANOGRAPHY INVOLVES EMBEDDING SECRET MESSAGES WITHIN THE UNICODE CHARACTER SET ITSELF. BY EXPLOITING LESS COMMON OR VISUALLY SIMILAR CHARACTERS, COVERT MESSAGES CAN BE CONCEALED WITHIN PLAIN TEXT, MAKING THEM VIRTUALLY UNDETECTABLE TO THE NAKED EYE.

Exploring the Challenge

LET'S DELVE INTO A PRACTICAL EXAMPLE TO ILLUSTRATE UNICODE STEGANOGRAPHY IN ACTION. CONSIDER A SCENARIO WHERE WE ENCOUNTER A SUSPICIOUS MESSAGE CONCEALED WITHIN A SEEMINGLY INNOCENT TEXT FILE. THROUGH CAREFUL EXAMINATION AND THE USE OF COMMAND-LINE TOOLS, WE'LL UNCOVER THE HIDDEN MESSAGE ENCODED WITHIN UNICODE CHARACTERS.

SPECTOGRAM



A SPECTROGRAM IS A VISUAL REPRESENTATION OF THE SPECTRUM OF FREQUENCIES IN A SIGNAL AS IT VARIES WITH TIME. IT DISPLAYS THE INTENSITY OF THE FREQUENCIES PRESENT IN AN AUDIO SIGNAL OVER TIME, ALLOWING FOR DETAILED ANALYSIS AND VISUALIZATION OF SOUND CHARACTERISTICS. SPECTROGRAMS ARE COMMONLY USED IN FIELDS SUCH AS MUSIC ANALYSIS, SPEECH RECOGNITION, AND AUDIO PROCESSING.

SPECTROGRAM STEGANOGRAPHY

SPECTROGRAM STEGANOGRAPHY INVOLVES EMBEDDING SECRET MESSAGES WITHIN THE FREQUENCY DOMAIN OF AN AUDIO SIGNAL. BY MANIPULATING SPECIFIC FREQUENCY COMPONENTS OR AMPLITUDE VARIATIONS, COVERT INFORMATION CAN BE CONCEALED WITHIN THE AUDIO FILE WITHOUT PERCEPTIBLE DISTORTION TO THE HUMAN EAR. SPECTROGRAM STEGANOGRAPHY OFFERS A HIGH LEVEL OF CONCEALMENT, AS THE HIDDEN MESSAGE IS IMPERCEPTIBLE WHEN LISTENING TO THE AUDIO FILE.

Sound pattern of thing like dial Number

Dial tone steganography leverages the familiar sounds produced by telephone keypads to encode secret messages. Each digit on a phone keypad generates a distinct dual-tone multi-frequency (DTMF) signal consisting of two simultaneous frequencies. By mapping these frequencies to characters or symbols, messages can be encoded and decoded using dialing sequences.

Creating a Steganography Flag

Here's a step-by-step guide on how to create a steganography flag using dial tone sounds:

Mapping Characters to Frequencies: Define a mapping between characters (e.g., letters, numbers, symbols) and the corresponding DTMF frequencies generated by a phone keypad. For example:

Dial tone steganography leverages the familiar sounds produced by telephone keypads to encode secret messages. Each digit on a phone keypad generates a distinct dual-tone multi-frequency (DTMF) signal consisting of two simultaneous frequencies. By mapping these frequencies to characters or symbols, messages can be encoded and decoded using dialing sequences.

Creating a Steganography Flag

Here's a step-by-step guide on how to create a steganography flag using dial tone sounds:

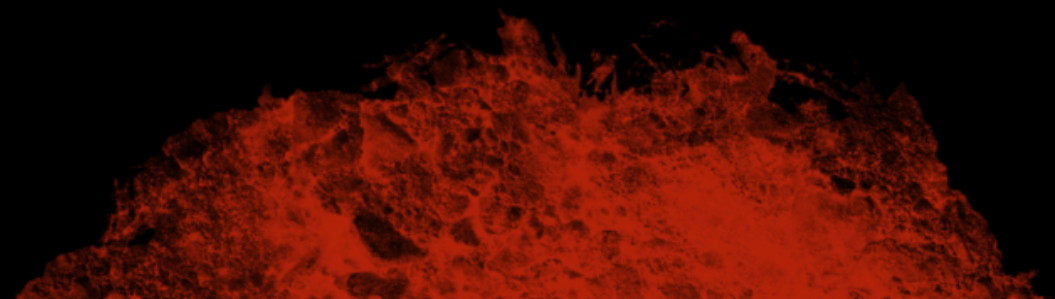
Mapping Characters to Frequencies: Define a mapping between characters (e.g., letters, numbers, symbols) and the corresponding DTMF frequencies generated by a phone keypad. For example:

1. Encoding the Flag: Convert the desired flag (message) into a sequence of characters. Then, for each character, determine the corresponding DTMF frequencies and generate the corresponding dial tones. You can use software or hardware capable of generating DTMF tones for this purpose.

2. Recording the Dial Tones: Record the dial tones generated for each character to create the audio representation of the steganography flag. Ensure that each dial tone is distinct and clearly audible.

References

* https://redteamguides.com/
* https://github.com/openpreserve/format-corpus
* https://github.com/openpreserve/format-corpus/tree/master



cat ~/.hadess

"Hadess" is a cybersecurity company focused on safeguarding digital assets and creating a secure digital ecosystem. Our mission involves punishing hackers and fortifying clients' defenses through innovation and expert cybersecurity services.

Website:
WWW.HADESS.IO

Email
MARKETING@HADESS.IO

To be the vanguard of cybersecurity, Hadess envisions a world where digital assets are safeguarded from malicious actors. We strive to create a secure digital ecosystem, where businesses and individuals can thrive with confidence, knowing that their data is protected. Through relentless innovation and unwavering dedication, we aim to establish Hadess as a symbol of trust, resilience, and retribution in the fight against cyber threats.