DIGITAL SIGNAL PROCESSING

DSP 2020-2021 |24/6

CONTENTS

I. INTRODUCTION

1. Topic

Audio Steganography: text is embedded into audio file without loss of audio quality, and while extraction can get the embedded text back.

2. Members of group

Hoàng Hữu Huy – BI10-077 (30%)

Lý Anh Kiệt – BI10-092 (25%)

Phạm Hoàng Việt – BI10-192 (20%)

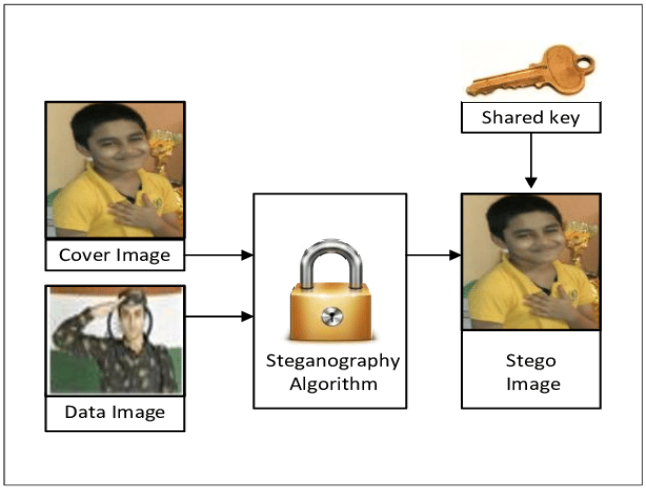
Phạm Đức Thắng – BI10-159 (25%)

II. AUDIO STEGANOGRAPHY

1. Definition

Steganography comes from the Greek word. It means covered or secret message. Steganography is the practice of concealing a message within another message or a physical object.

It is utilized to shroud important private messages from a third party inside different innocuous messages.

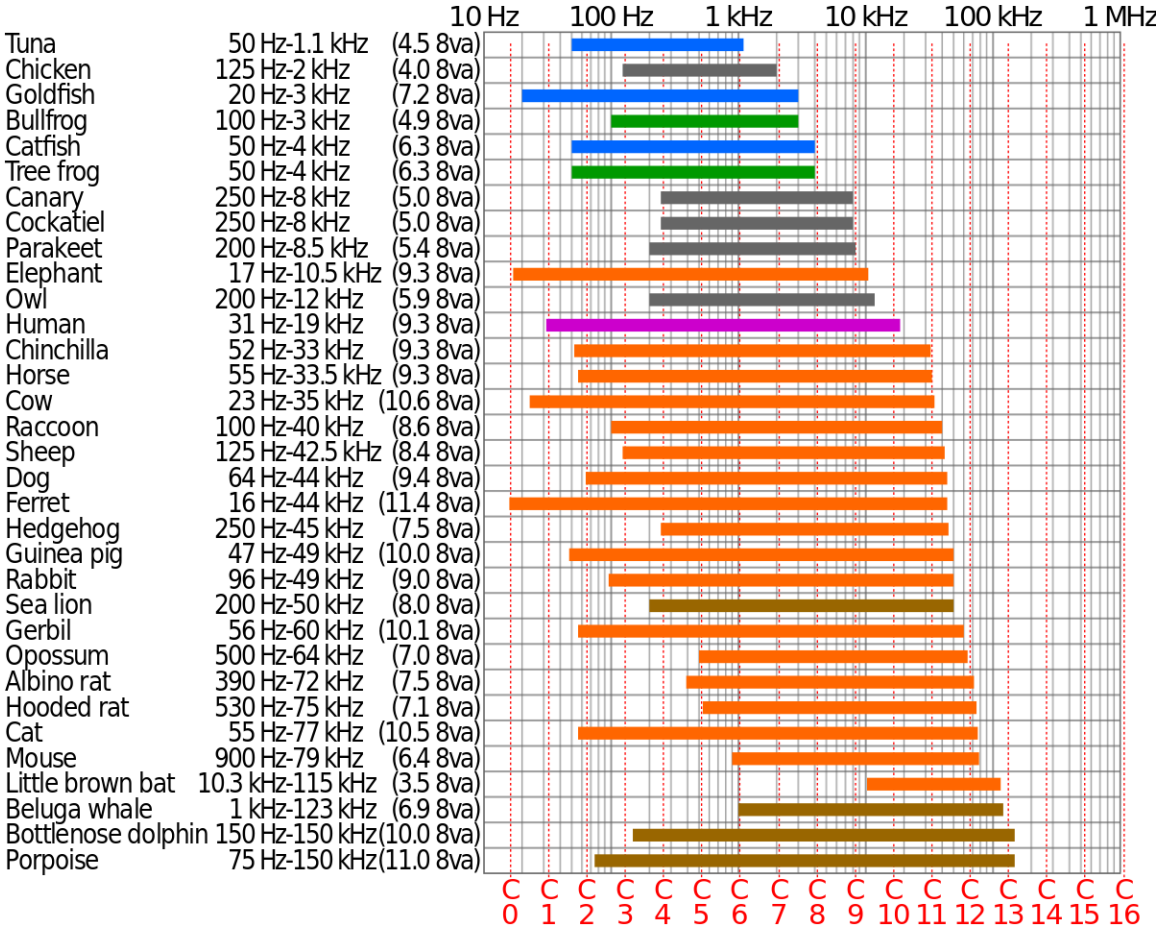


This may be accomplished by concealing the presence of data inside apparently safe carriers or cover.

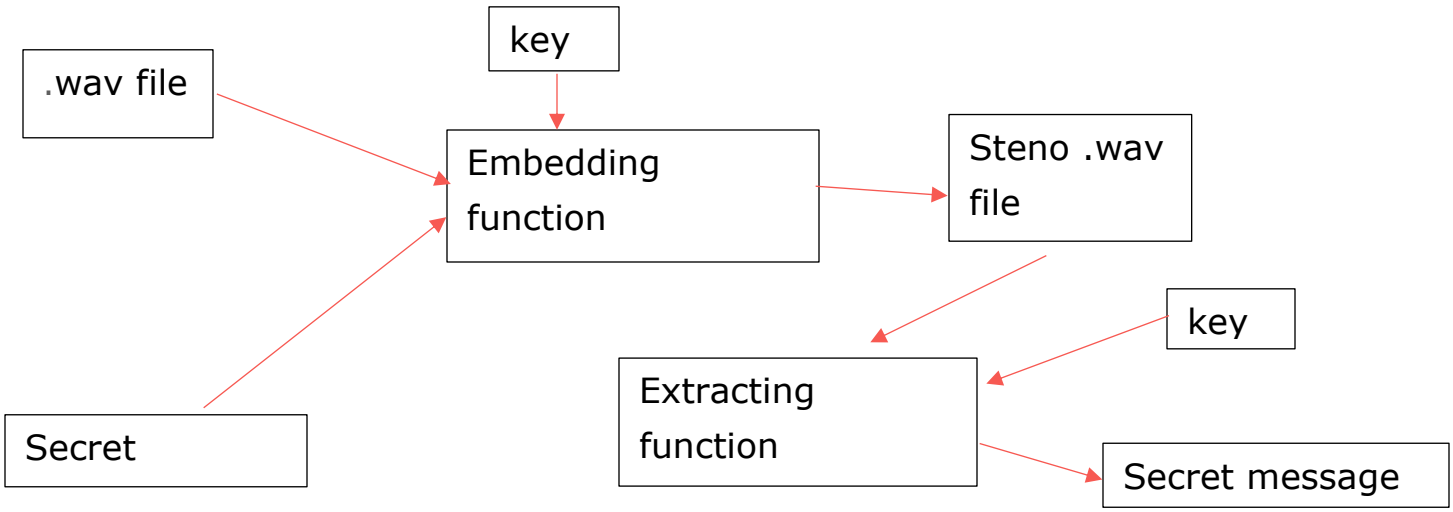
2. Audio steganography

Information hiding technique is a new kind of secret communication technology. The majority of today's information hiding systems uses multimedia objects like image, audio, video. Audio Steganography (AS), it is a technique used to transmit hidden information by modifying an audio signal in an imperceptible manner. It is the science of hiding some secret text or audio information in a host message.

The key to all of the methods is to exploit the Human Auditory System (HAS). The human ear can choose up the vibrations of a layer between the recurrence run of 20 Hz and 20 kHz.



So one way to realize the Audio steganography would be to utilize Infrasound or Ultrasound extend to transmit our “secret” message, which would go with by a “public” sound information being played on the capable of being heard recurrence extend, in arrange to misdirect the unintended recipient.



The idea we use is Whereas implanting the stealthy information the arrange has got to be beyond any doubt so that that header portion of the wave record (to begin with 44 bytes) ought to be untouched since in case the header gets debased, the audio file will too degenerate as within the figure. The moment thought that ought to be made isn't to insert information into the quiet zone as that might cause undesirable to alter to the sound record.

There are a lot of way to hide text on audio file, such as:

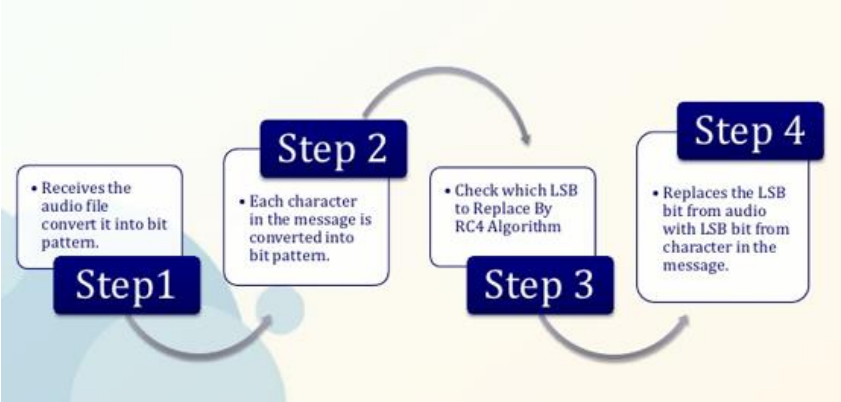
- Least Significant Bit (LSB) Coding
- Echo Data Hiding
- Parity Coding
- Phase Coding
- Spread Spectrum

In this topic, we work with the technique Least Significant Bit.

3. LSB technique

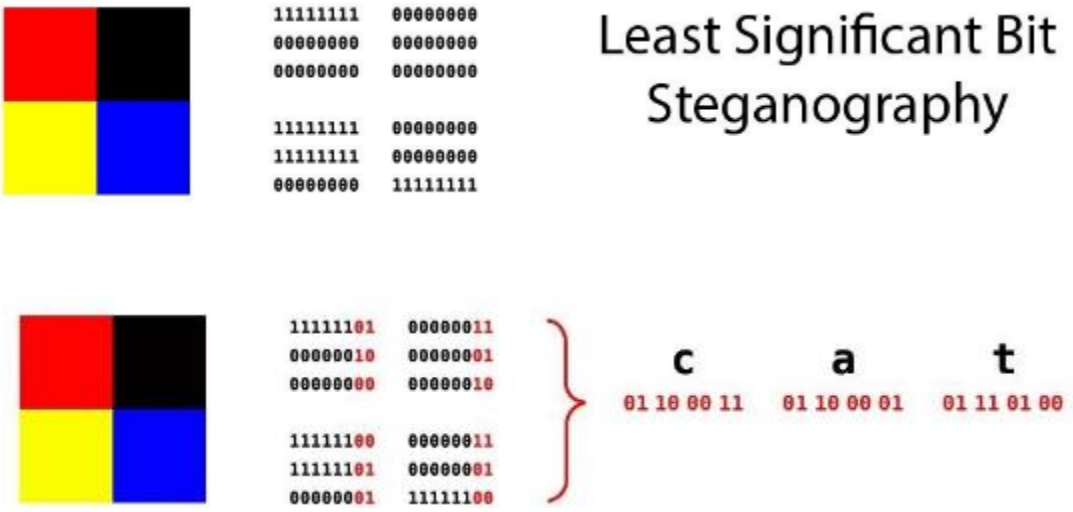
First, we need to understand about Least Significant Bit or LSB for short. The slightest critical bit is the least bit in an arrangement of numbers in double. It is either the furthest left or furthest right bit in a twofold number, depending on the computer's engineering. If the LSB is corrected, the design is called "little-endian." In case the LSB is on the cleared out, the design is called "big-endian." For illustration, in a little-endian design, the LSB of parallel number 00000001 is 1.

LSB calculation may be a classic Steganography strategy utilized to conceal the presence of mystery information interior a “public” cover. The LSB or “Least Critical Bit”, in computing terms, represents the bit at the unit’s put within the double representation of a number.

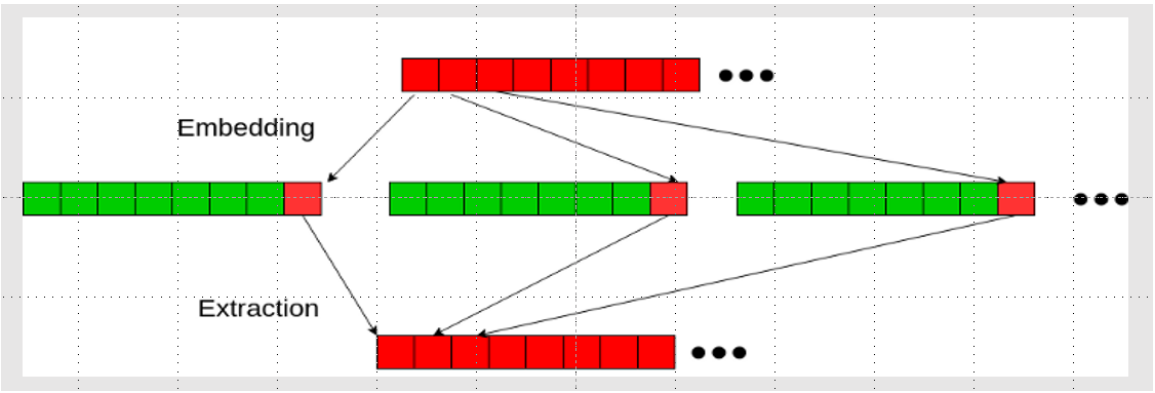


4. How it works?

From the idea above, now we move how LSB works. For the LSB, we will be interested in the Least Significant bit. suppose with 8 bits 10001001, last 4 bits 1001 can be replaced by another bit. The audio file can be converted to a matrix of binary codes, and for each text, converted to binary code. Next, the binary code representing the text is replaced with the Least Significant bit of the audio file. From there, you can hide the text. For each switch, there will be its own key.



Next is how to extract.



For command, the length of the message to be encoded is smaller than the total samples of an audio file. If it is larger than, it can be affected the audio quality. This affects security. If we hide too much in the audio file, the attacker can realize the difference and attack it. To decode, we will use the key to encode

and reverse use it to get the LSB. After that, we cover the bit to become the text. This will make the text in the receiver as same as the text hide.

Finally, how to propose the LSB technique? In our proposed show we take consecutive two bits from the mystery message and instep of changing a single bit in a test we alter two bits (4th and 3rd position) of the test. In the event that there's alter in these two bits we flip the rest of the LSB something else there's no alter.

5. Advantages and disadvantages

The advantages of Least-Significant-Bit (LSB) steganographic data embedding are that it is simple to understand, easy to implement, and it results in stego-audio that contain hidden data yet appear to be of high sound fidelity. This strategy included security, capacity, and vigor, the three required perspectives of steganography that makes it valuable in covering up a trade of data through content reports and setting up a secret communication

But it also has disadvantages about the security, the fatal drawback of LSB embedding is the existence of detectable artifacts in the form of pairs of values (PoVs). If the attacker can have a key, he can get the hiding texts. So the secure problem still not a good answer.

III, IMPLEMENT IN PYTHON

```
def Embedding():
        audio = wave.open(root.input, mode='rb')
        bytes_covert = bytearray(list(audio.readframes(audio.getnframes())))
        string = message.get("1.0",'end-1c')
        string = string + int((len(bytes_covert) - (len(string) * 8 * 8)) / 8) * '#'
        bits = list(map(int, ''.join([bin(ord(i)).lstrip('0b').rjust(8, '0') for i in s
        for i, bit in enumerate(bits):
            bytes_covert[i] = (bytes_covert[i] & 254) | bit
        frame_modified = bytes(bytes_covert)
        for i in range(0, 10):
            print(bytes_covert[i])
        name_output = os.path.join(root.output, outputname.get() + ".wav")
        output = wave.open(name_output, 'wb')
        output.setparams(audio.getparams())
        output.writeframes(frame_modified)

        output.close()
        audio.close()
```

Here is the embedding code in Python.

```
ef Extracting():
      Mess.delete('1.0', END)

      audio = wave.open(root.input, mode='rb')
      frame_bytes = bytearray(list(audio.readframes(audio.getnframes())))
      extracted = [frame_bytes[i] & 1 for i in range(len(frame_bytes))]
      string = "".join(chr(int("".join(map(str, extracted[i:i + 8])), 2)) for i in
      hidden_message = string.split("###")[0]
      Mess.insert(tkinter.END, hidden_message)
      audio.close()
```

And here is code that use to extract message.

IV, REFERENCES

https://www.ijcsmc.com/docs/papers/April2014/V3I4201468.pdf

https://null-byte.wonderhowto.com/how-to/steganography-hide-secret-data-inside-image-audio-file-seconds-0180936/

https://sumit-arora.medium.com/audio-steganography-the-art-of-hiding-secrets-within-earshot-part-1-of-2-6a3bbd706e15

https://sumit-arora.medium.com/audio-steganography-the-art-of-hiding-secrets-within-earshot-part-2-of-2-c76b1be719b3

https://en.wikipedia.org/wiki/Steganography

https://www.computerhope.com/jargon/l/leastsb.htm