

MIT WORLD PEACE UNIVERSITY

Digital Forensics and Investigation  
Third Year B. Tech, Semester 5

---

---

AUDIO AND IMAGE FILE INTEGRITY ANALYSIS

---

---

LAB ASSIGNMENT 6

Prepared By

Krishnaraj Thadesar  
Cyber Security and Forensics  
Batch A1, PA 20

November 1, 2023

# Contents

<b>1 Aim</b>	<b>1</b>
<b>2 Objectives</b>	<b>1</b>
<b>3 Theory</b>	<b>1</b>
3.1 How to Secure Audio Files? . . . . .	1
3.2 What is Steganography . . . . .	1
3.3 Hashing and Methods to check Integrity . . . . .	2
<b>4 Analysis and Experiment</b>	<b>2</b>
4.1 Performing Steganography for Audio and Image Files . . . . .	2
4.1.1 Audio Files . . . . .	2
4.1.2 Encrypting Data in the Audio File. . . . .	3
4.1.3 Text files involved . . . . .	4
4.1.4 Image File . . . . .	4
4.2 Analysing the Differences using Software . . . . .	4
<b>5 Platform</b>	<b>5</b>
<b>6 Code</b>	<b>5</b>
<b>7 Conclusion</b>	<b>7</b>
<b>References</b>	<b>8</b>

## **1 Aim**

Write a program for identifying tampering in either image or voice data. Use big data as input.

## **2 Objectives**

1. To learn about audio and image file integrity.
2. To learn ways to secure and make sure integrity of audio and image files is maintained.

## **3 Theory**

### **3.1 How to Secure Audio Files?**

1. Digital Rights Management (DRM) - This is a technology that controls the use and distribution of digital files, such as audio files. DRM can be used to restrict the number of times a file can be played, or to prevent the file from being copied or shared.
2. Encryption - This is a method of converting the audio file into a code that can only be accessed by those who have the decryption key. This method can be used to protect the file from unauthorized access or copying.
3. Watermarking - This is a method of adding a unique identifying mark, such as a logo or text, to the audio file. Watermarking can be used to trace the origin of an unauthorized copy of the file.
4. Copyright Notice - This is a legal notice that informs others that the audio file is protected by copyright law and that unauthorized use of the file is prohibited.
5. Limited distribution - Keep the distribution of the audio file limited to a specific group or individuals, this can be achieved by sending the audio file via email or cloud storage with restricted access.
6. Use of a Licensing agreement- Having a licensing agreement in place can set specific terms and conditions on how the audio file can be used, this will help you protect your audio file from unauthorized use.

It is still worth noting that none of these methods can completely prevent unauthorized access or distribution of an audio file, but they can make it more difficult and less attractive for people to do so. It's also important to register your audio file with the Copyright Office for additional legal protection.

### **3.2 What is Steganography**

Steganography is the practice of concealing information within other non-secret data, often within media files such as images, audio, or video. This covert technique allows you to hide a message or data within a seemingly innocuous carrier file.

### **3.3 Hashing and Methods to check Integrity**

Hashing is a process of converting data into a fixed-size string of characters, which is typically a hexadecimal number. It is used to verify data integrity, detect tampering, and ensure data consistency.

1. **Checksums:** Checksums are simple hashes used to verify the integrity of data. Common examples include CRC32 and MD5. For example, you can generate an MD5 checksum for a file and compare it with the original checksum to detect changes.
2. **Cryptographic Hash Functions:** Cryptographic hash functions, like SHA-256 (Secure Hash Algorithm 256-bit), are widely used for secure data verification. They produce unique, fixed-size hashes. For instance, SHA-256 is often used in blockchain technology to validate transactions.
3. **HMAC (Hash-Based Message Authentication Code):** HMAC is a technique that combines a cryptographic key with a hash function. It's used for data authentication. An example is HMAC-SHA-256, which adds a secret key to the hashing process.
4. **Digital Signatures:** Digital signatures use asymmetric cryptography to hash data and encrypt the hash with a private key. The recipient can decrypt the hash with the sender's public key and compare it to a freshly computed hash. This ensures both integrity and authenticity.
5. **File Verification Tools:** Various software tools, such as the 'sha256sum' command in Unix-based systems, calculate and compare hashes for file integrity.

## **4 Analysis and Experiment**

### **4.1 Performing Steganography for Audio and Image Files**

#### **4.1.1 Audio Files**

Shown here is the audio file 'Alesis-Fusion-Clean-Guitar-C3.wav' that was used to perform this analysis.

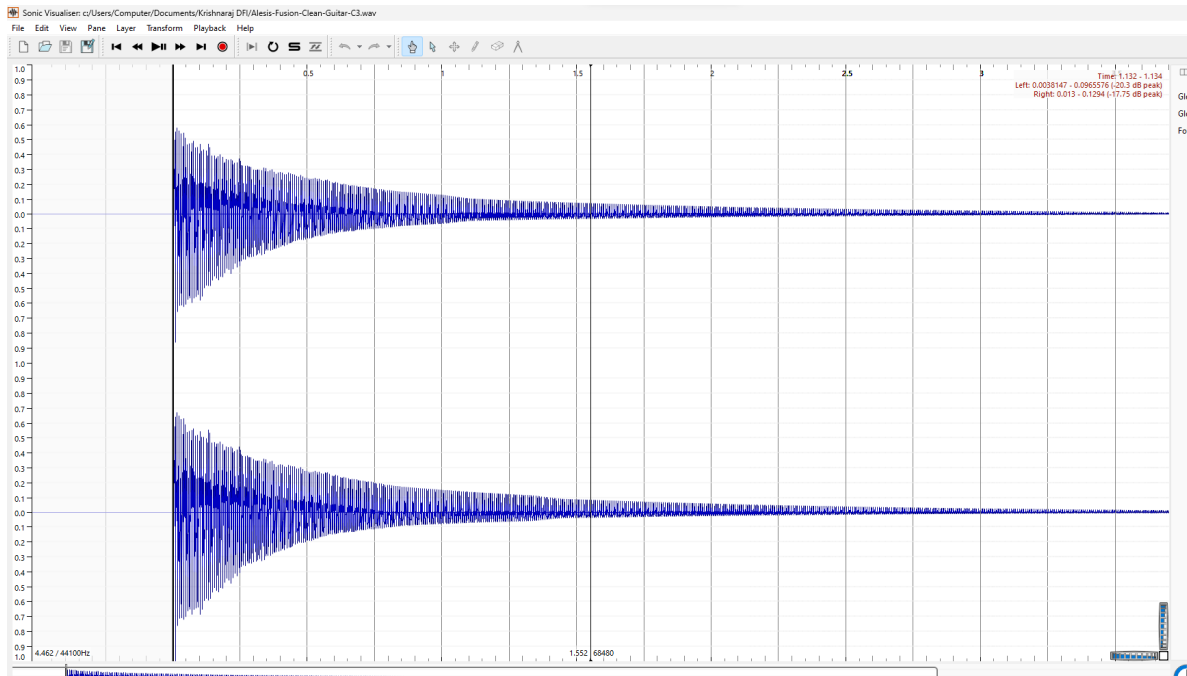


Figure 1: Audio File as seen on Sonic Visualizer.

#### 4.1.2 Encrypting Data in the Audio File.

This was done using a python package called 'stegolsb'.

```
PS C:\Users\MTECH\Downloads> & C:/Users/MTECH/AppData/Local/Programs/Python/Python37/pyt
c:/Users/MTECH/Downloads/sha.py
Welcome to DFI Assignment 5, Krishnaraj PT. PA10. CSF
File Modified Detection
First Music File without Tampering:
classical_guitar.mp3
Hash is: 059df4238eb2f90db772d60177df456e92858d93
Music File after deleting portion of it :
Hash is: c08d765cae3380bd101c8e5d6cb8b7412487e56c
classical_guitar_modified.mp3
=====
Image File
Hash of the file is:
b91bf2a1dd825725f788a7e204cfd38d0a4badd1bf3745ec2efceeb4e3cab591
Modified the File, added hidden data to it.
Hash of the file now is:
d02d021cc9c7771339327216c82540b1796ce1f2d715797907c97ee7c24788e8
```

Figure 2: Encrypting a Message using Stegolsb

```

PS C:\Users\Computer\Documents\Krishnaraj DFI> stegolsb wavsteg -r -i sound_steg.
wav -o output.txt -n 1 -b 1000
Files read                               in 0.00s
Recovered 1000 bytes                     in 0.00s
Written output file                       in 0.00s
PS C:\Users\Computer\Documents\Krishnaraj DFI> stegolsb wavsteg -h -i .\Alesis-Fu
sion-Clean-Guitar-C3.wav -s .\file.txt -o sound_steg.wav -n 1
Using 1 LSBs, we can hide 49195 bytes
Files read                               in 0.00s
93 bytes hidden                           in 0.00s
Output wav written                        in 0.00s
PS C:\Users\Computer\Documents\Krishnaraj DFI> |

```

Figure 3: Decrypting a Message using Stegolsb

#### 4.1.3 Text files involved

Shown here are 2 files.

1. The First file shows the file.txt that was used to encrypt the message.
2. The second file, output.txt is where the text output was given.

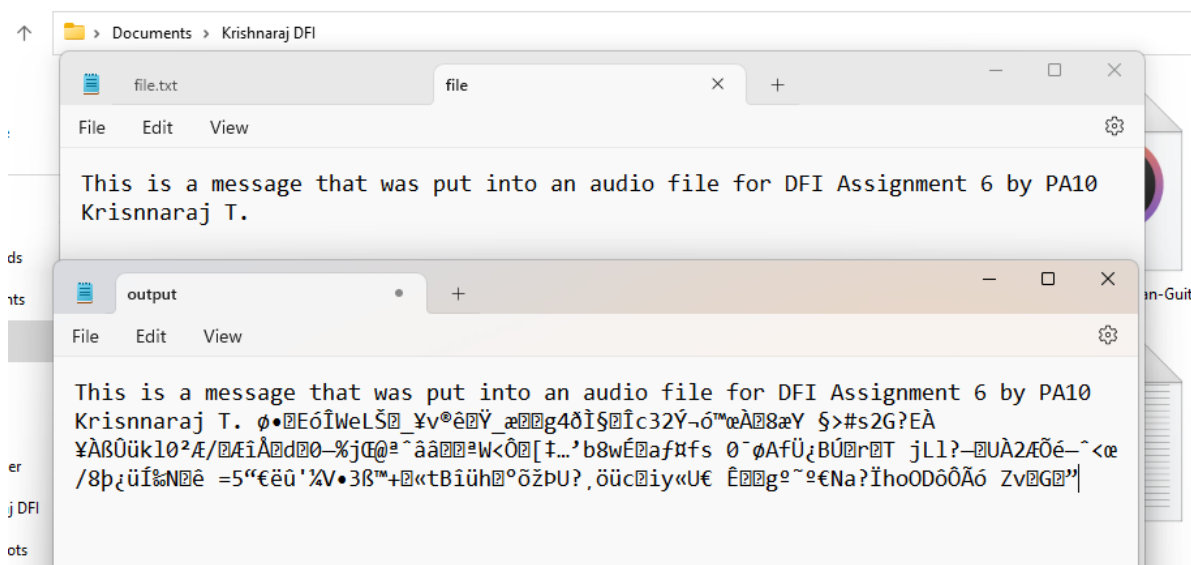


Figure 4: Decrypting a Message using Stegolsb

#### 4.1.4 Image File

### 4.2 Analysing the Differences using Software

Shown below is the when both the files were opened side by side in sonic visualizer, and no visible difference between the 2 files was noted. The changes were present when checked with hashes, but not visible visually.

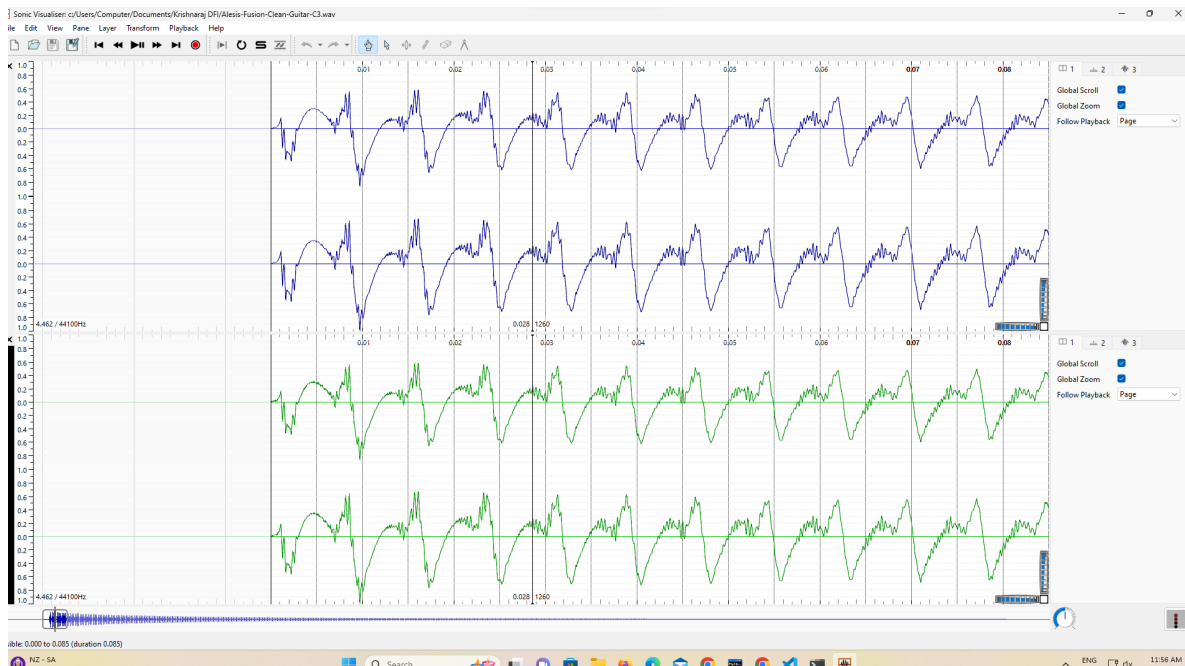


Figure 5: This shows that there is no change in both files visually when seen side by side.

## 5 Platform

**Operating System:** Arch Linux x86-64

**IDEs or Text Editors Used:** Visual Studio Code

**Compilers or Interpreters:** Python 3.12

## 6 Code

```

1 import hashlib
2
3 def hash_file(filename):
4     """This function returns the SHA-1 hash
5     of the file passed into it"""
6
7     # make a hash object
8     h = hashlib.sha1()
9
10    # open file for reading in binary mode
11    with open(filename, 'rb') as file:
12
13        # loop till the end of the file
14        chunk = 0
15        while chunk != b'':
16            # read only 1024 bytes at a time
17            chunk = file.read(1024)
18            h.update(chunk)
19
20    # return the hex representation of digest

```

```

21     return h.hexdigest()
22
23 print("Welcome to DFI Assignment 6, Krishnaraj PT. PA10. CSF")
24 print("File Modified Detection")
25
26 file_name = "Alesis-Fusion-Clean-Guitar-C3.wav"
27 file_name_modified = "sound_steg.wav"
28
29 print("First Music File without Tampering: ")
30 print(file_name)
31 print("Hash is: ", hash_file(file_name))
32
33 print("Music File after deleting portion of it : ")
34 print("Hash is: ", hash_file(file_name_modified))
35 print(file_name_modified)
36
37
38 print("====" * 21)
39
40 print("Image File")
41
42 from hashlib import sha256
43 import os
44
45 file_location = os.path.join(os.path.dirname(__file__), "tony.jpg")
46 # Read image file
47
48 print("Hash of the file is: ")
49 # Convert to byte array
50 with open(file_location, "rb") as image_file:
51     f = image_file.read()
52     b = bytearray(f)
53
54 # Print the result
55 print(sha256(b).hexdigest())
56
57 print("Modified the File, added hidden data to it.")
58 print("Hash of the file now is: ")
59
60 # run this command to modify the file
61
62 file_location = os.path.join(os.path.dirname(__file__), "tony_changed.jpg")
63 # Read image file
64
65 # Convert to byte array
66 with open(file_location, "rb") as image_file:
67     f = image_file.read()
68     b = bytearray(f)
69
70 # Print the result
71 print(sha256(b).hexdigest())

```

Listing 1: "Python Script to Verify Hash"

```

1 Welcome to DFI Assignment 6, Krishnaraj PT. PA10. CSF
2 File Modified Detection
3 First Music File without Tampering:
4 Alesis-Fusion-Clean-Guitar-C3.wav
5 Hash is: 242fff54feface4f9ac8b51887deb4eee052f3d7

```



```
6 Music File after deleting portion of it :
7 Hash is: 89755f2b3b0438750818c1ae5fd26a93624cece8
8 sound_steg.wav
9 =====
10 Image File
11 Hash of the file is:
12 b91bf2a1dd825725f788a7e204cfd38d0a4badd1bf3745ec2efceeb4e3cab591
13 Modified the File, added hidden data to it.
14 Hash of the file now is:
15 d02d021cc9c7771339327216c82540b1796ce1f2d715797907c97ee7c24788e8
```

Listing 2: Output

## 7 Conclusion

Thus, we have successfully Analysed the differences between the audio and image file when it is modified.

1. It was found that when an audio file is changed using steganography, the Hash changes.
2. A Similar change was noticed when a message was encrypted in an image file as well.
3. It was also seen that the wave file of the audio was also modified. The changes were also visually invisible. This was done using sonic visualizer.

## **References**

- [1] [A Novel Steganography Approach for Audio Files](#)  
Sazeen T. Abdulrazzaq, Mohammed M. Siddeq, Marcos A. Rodrigues
- [2] [Sonic Visualizer](#)
- [3] [Audio Steganography : The art of hiding secrets within earshot](#)  
Sumit Kumar Arora
- [4] [stego-lsb package from python.](#)