<u>Information Security Project Report</u>

"Encryption and Decryption of Audio Files using AES and RSA Techniques"

Jaypee Institute of Information Technology

Submitted by

Ananya Kapoor	*20103104*	*B4*
Dhairya Sachdeva	*20103098*	*B4*
Sparsh Celly	*20103102*	*B4*
Aakarsh Srivastava	*20103096*	*B4*

Submitted To

Dr. Arpita Jadhav Bhatt

DECLARATION

We hereby declare that the project titled *“Encryption and Decryption of Audio Files using AES and RSA Techniques”* submitted for the degree of Bachelors in Technology is a bonafide record submitted to Jaypee Institute of Information Technology, under the guidance of Dr. Arpita Jadhav Bhatt, has been carried out by our own efforts and is a record of our original work.

1. Abstract

One of the most important methods to protect and verify information that are exchanged over public communication channels in the existence of third party called antagonists is encryption. The stored or transmitted message is transformed in the encryption process to unreadable or gibberish form. The reverse process in which the intendent recipient can reveal the encrypted message content is called decryption. The encryption and decryption processes are achieved using secret keys that are exclusively exchanged between the sender and recipient. This method can be applied to any form of message such as audio, video, image or text data. The current work applies the well-known RSA and AES algorithm for audio signal encryption and decryption.

2. Introduction

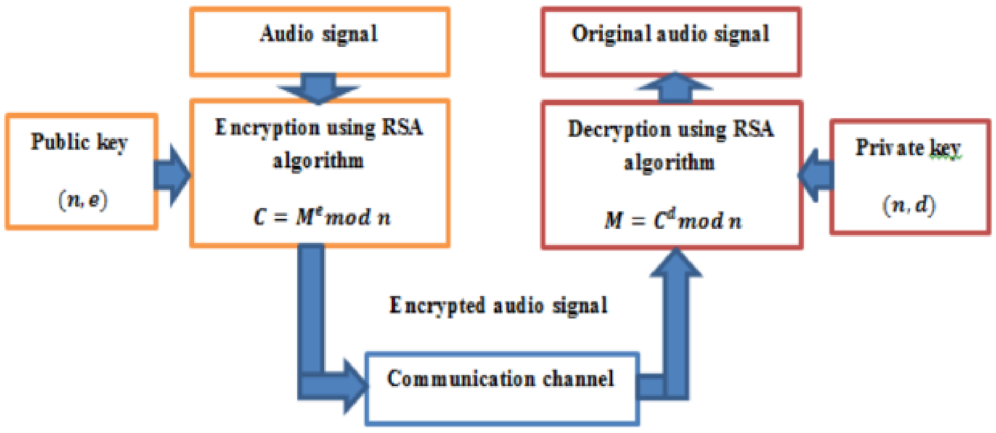
The most important aspect in our daily life is data communication. The main issue in data communication is data security to preserve its availability, integrity, proper access control as well as confidentiality. Therefore, Data protection is essential from misuse. The necessity to protect communication today in the e-age from intruders has become greater than ever before. Data protection has been traditionally ensured with cryptography which plays a major role throughout many applications such as e-commerce, e-mail, mobile phone communication, Pay-Tv, sending financial information and so forth. Cryptography can be defined as the science of implementing and developing techniques to encrypt a message in a way that could be impossible for intruders and illegal persons to detect or modify its contents whether it’s being in storing or sending case. Only the intendent user can recover the content of the encrypted message by decryption operation using secret key which is shared between the transmitter and receiver.

Fundamentally, cryptosystems can be categorized to two main types based on the way in which encryption and decryption processes are carried out in the cryptosystem: symmetric key (secret key) cryptosystems and asymmetric key (public key) cryptosystems. In symmetric key cryptosystems, the same key is shared between the transmitter for encryption and the receiver for decryption the data. The strength of symmetric algorithms depends on the size of the secret key. Blowfish, Advanced Encryption Standard (AES) and Data Encryption Standard (DES) are examples of symmetric key encryption techniques. In asymmetric key cryptosystems, each user in the system has two different keys:

private which is used for encryption at the transmitter and public which is used for decryption at the receiver. Although these two keys are different but they are mathematically related. Thus, reconstructing of the original message by decrypting it is feasible. Asymmetric cipher has several advantages over conventional symmetric ciphers. It means that if the opponent can have both the algorithm of encryption and the public key, he will still need to the private key to decrypt the original message which is available only with the intendent recipient. The disadvantage of this type of encryptions is that they are need more computations than symmetric ciphers. Hence, encryption and decryption processes may take longer time. For a short message, this is not suitable, but for bulk data encryption, it certainly does. RSA is an example of asymmetric key encryption methods.

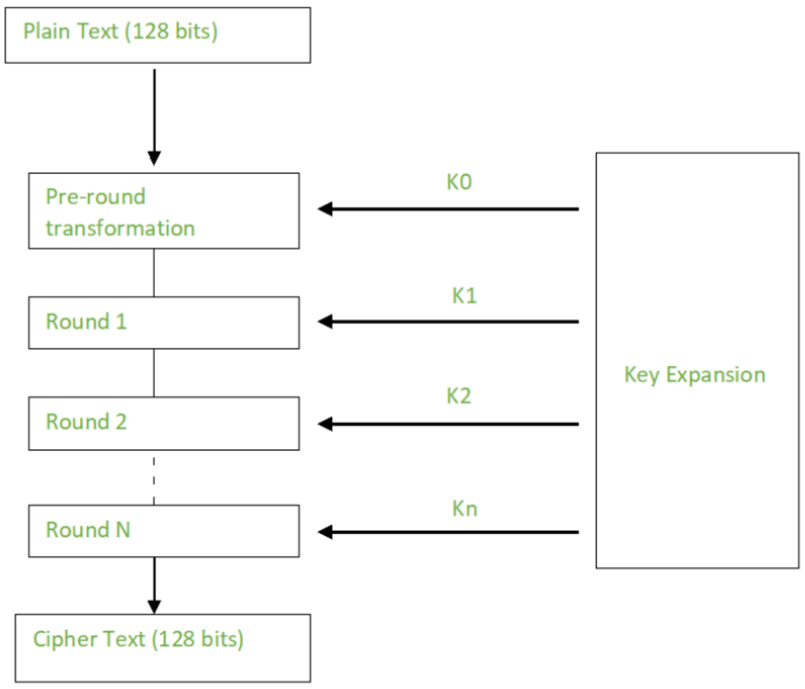
3. RSA Algorithm

RSA is basically an authentication system and Internet encryption method. Initially, two large prime numbers are chosen and multiplied in this algorithm to create the public and the private keys pair which are further used in the encryption and decryption operations. There is no need to send the private key throughout the Internet if RSA algorithm is used. The private key is utilized to decrypt the secret message at the receiver which has been ciphered or encrypted by using the public key at the transmitter. Everyone can know the public key which is used to encrypt the messages, but the encrypted messages by the public key can be decrypted only with the private key. Three steps are involved in RSA which are key generation, encryption and decryption processes.

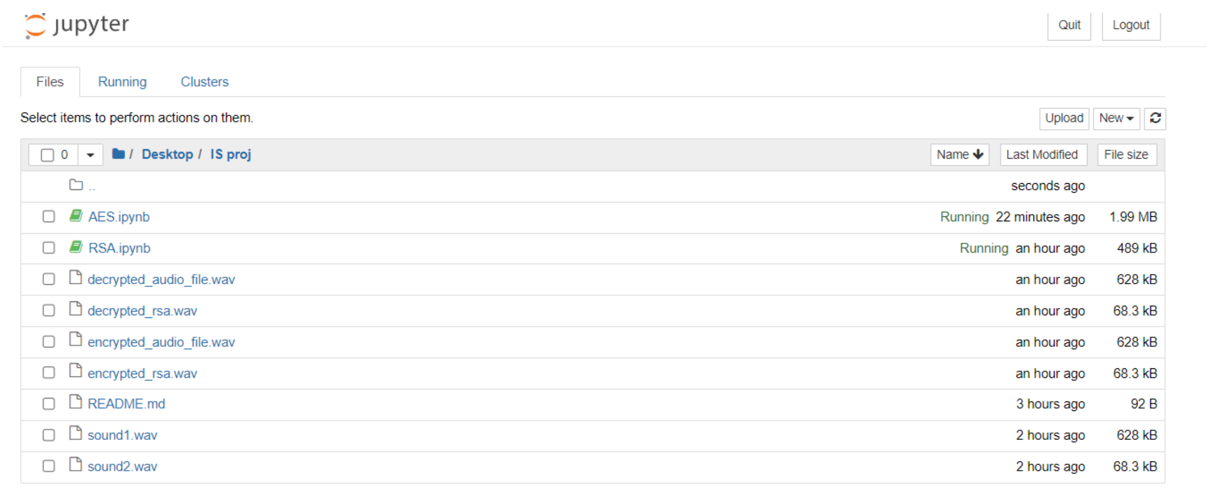


4. AES Algorithm

Advanced Encryption Standard (AES) is a specification for the encryption of electronic data established by the U.S National Institute of Standards and Technology (NIST) in 2001. AES is widely used today as it is a much stronger than DES and triple DES despite being harder to implement. AES takes 128 bits as input and outputs 128 bits of encrypted cipher text as output. AES relies on substitution-permutation network principle which means it is performed using a series of linked operations which involves replacing and shuffling of the input data. AES performs operations on bytes of data rather than in bits. Since the block size is 128 bits, the cipher processes 128 bits (or 16 bytes) of the input data at a time.



5. Code and Results



Select items to perform actions on them.

0 / Desktop / IS proj

Name		Last Modified	File size
..		seconds ago	
AES.ipynb	Running	22 minutes ago	1.99 MB
RSA.ipynb	Running	an hour ago	489 kB
decrypted_audio_file.wav		an hour ago	628 kB
decrypted_rsa.wav		an hour ago	68.3 kB
encrypted_audio_file.wav		an hour ago	628 kB
encrypted_rsa.wav		an hour ago	68.3 kB
README.md		3 hours ago	92 B
sound1.wav		2 hours ago	628 kB
sound2.wav		2 hours ago	68.3 kB

Using RSA

Importing dependencies

In [1]:
```
!pip install sounddevice
!pip install librosa
!pip install IPython
!pip install itertools
```

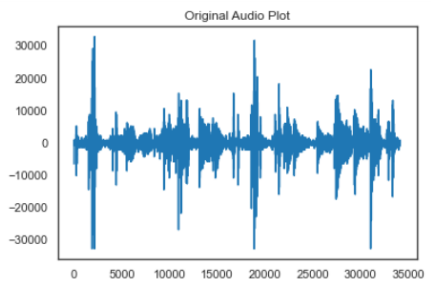
In [8]:
```
from scipy.io import wavfile
import numpy as np
import matplotlib.pyplot as plt
import sounddevice as sd
import seaborn as sns
import pandas as pd


import librosa
import librosa.display
import IPython.display as ipd

from itertools import cycle
```

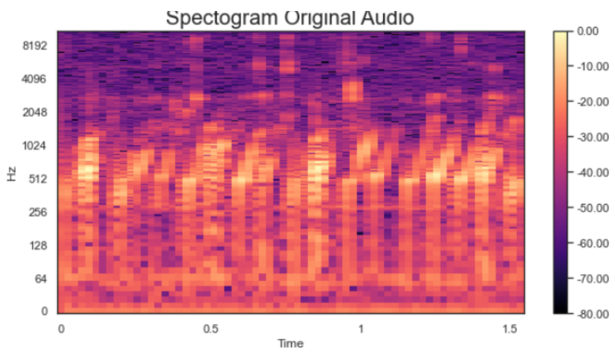
In [45]:
```
sns.set_theme(style="white", palette=None)
color_pal = plt.rcParams["axes.prop_cycle"].by_key()["color"]
color_cycle = cycle(plt.rcParams["axes.prop_cycle"].by_key()["color"])
```

Taking input

In [72]:
```
fs, data = wavfile.read('sound2.wav')
plt.plot(data)          # fs = sampling frequency = 44.1kHz
plt.title("Original Audio Plot")
data_1 = np.asarray(data, dtype = np.int32)
```

```
[-6571 -4382 -4390 ...   727   727     0]
```





Playing that sound

```
In [47]: #sd.play(data, fs)
         ipd.Audio('sound2.wav')

Out[47]: 0:00 / 0:01
```

```
In [49]: y, sr = librosa.load('sound2.wav')
         D = librosa.stft(y)
         S_db = librosa.amplitude_to_db(np.abs(D), ref=np.max)
         fig, ax = plt.subplots(figsize=(10, 5))
         img = librosa.display.specshow(S_db,
                                        x_axis='time',
                                        y_axis='log',
                                        ax=ax)
         ax.set_title('Spectogram Original Audio', fontsize=20)
         fig.colorbar(img, ax=ax, format=f'%0.2f')
         plt.show()
```

Generating public and private keys for RSA algorithm

Select two prime no's. Suppose P = 53 and Q = 59.

Now First part of the Public key : n = P*Q = 3127.

We also need a small exponent say e : But e Must be

```
1) An integer.

2) Not be a factor of n.

3) 1 < e < Φ(n) [Φ(n) is discussed below],
Let us now consider it to be equal to 3.
```

Our Public Key is made of n and e

Select two prime no's. Suppose P = 53 and Q = 59.

Now First part of the Public key : n = P*Q = 3127.

We also need a small exponent say e : But e Must be

```
1) An integer.

2) Not be a factor of n.

3) 1 < e < Φ(n) [Φ(n) is discussed below],
Let us now consider it to be equal to 3.
```

Our Public Key is made of n and e

1) We need to calculate Φ(n):

```
Such that Φ(n) = (P-1)(Q-1)
  so,  Φ(n) = 3016
```

2) Now calculate Private Key, d :

```
d = (k*Φ(n) + 1) / e for some integer k
For k = 2, value of d is 2011.
```

```
In [60]: p1 = int(input("Enter a prime number: "))
         p2 = int(input("Enter another prime number: "))

         n = p1*p2
         print("n = p1*p2 = ",n)

         import math
         print("A small, odd number, co-prime with n: ")
         for i in range(2,101):
             if math.gcd(n,i)==1:
                 e=i
                 break
         print(e)
```

```
        e=i
        break
print(e)
k = int(input("Enter value of k:"))
```

```
Enter a prime number: 53
Enter another prime number: 59
n = p1*p2 =  3127
A small, odd number, co-prime with n:
2
Enter value of k:2
```

In [61]:
```
phi = (p1-1)*(p2-1)
print("phi = ",phi)
d = int((k*phi+1)/e)
print("d= ",d)
```

```
phi =  3016
d=  3016
```

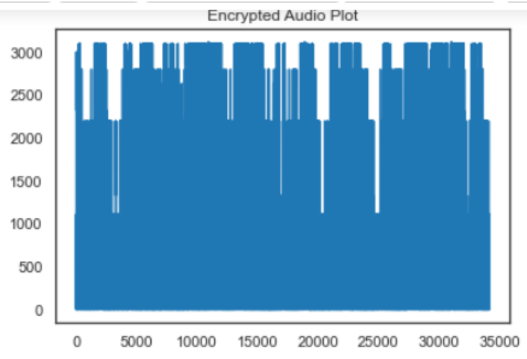
In [62]:
```
public_key = n,e
private_key = n,d

print("Public Key = ", public_key)
print("Private Key = ",private_key)
```

```
Public Key =  (3127, 2)
Private Key =  (3127, 3016)
```

Encrpytion of audio file

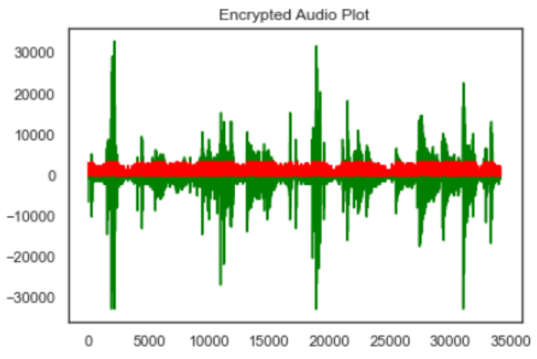
In [63]:
```
encrypted = (data**e)%n
plt.plot(encrypted)
plt.title("Encrypted Audio Plot")
```

Out[63]:
```
Text(0.5, 1.0, 'Encrypted Audio Plot')
```



In [73]:
```
fs, data = wavfile.read('sound2.wav')
plt.plot(data,color='green')          # fs = sampling frequency = 44.1kHz
plt.title("Original Audio Plot")
encrypted = (data**e)%n
plt.plot(encrypted,color='red')
plt.title("Encrypted Audio Plot")
```

Out[73]:
```
Text(0.5, 1.0, 'Encrypted Audio Plot')
```



Saving the saved file

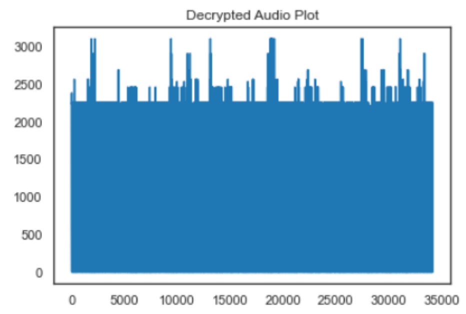
```
In [64]: encrypted = np.asarray(encrypted, dtype=np.int16)
         wavfile.write('encrypted_rsa.wav', fs, encrypted)
         print("A file titled 'encrypted_rsa.wav' is generated which is the encrypted audio to be communicated")

         A file titled 'encrypted_rsa.wav' is generated which is the encrypted audio to be communicated
```

Loading and decrypting

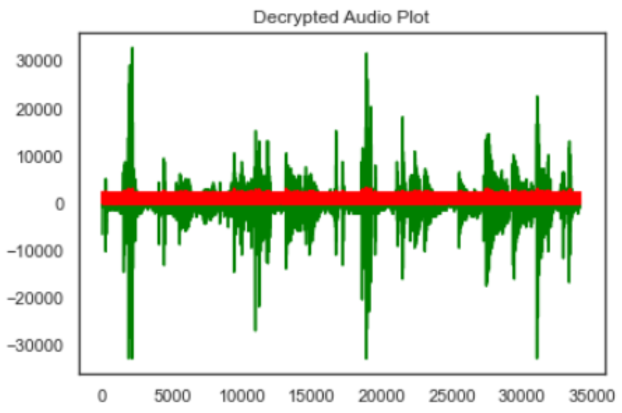
```
In [65]: fs, Data = wavfile.read('encrypted_rsa.wav')
```

Decryption of data

```
In [66]: decrypted = (Data**d)%n
         plt.plot(decrypted)
         plt.title('Decrypted Audio Plot')

Out[66]: Text(0.5, 1.0, 'Decrypted Audio Plot')
```

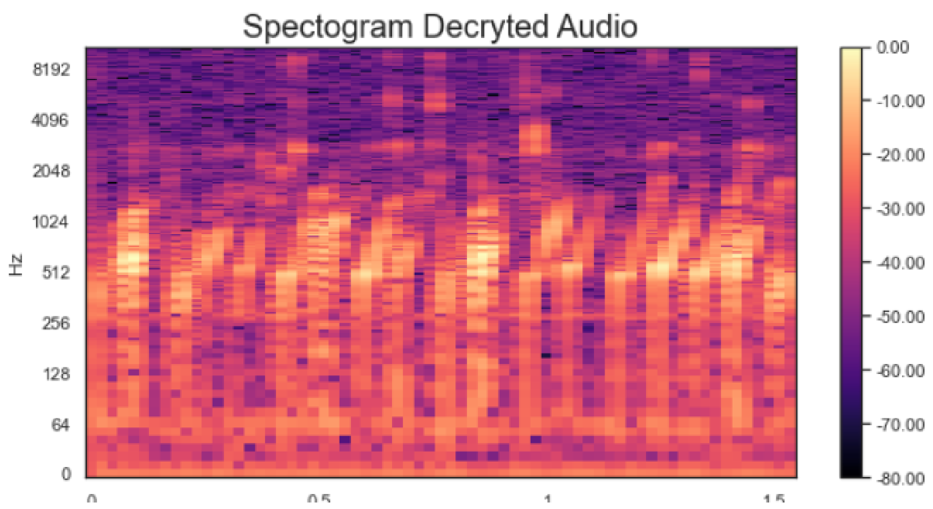


```
In [74]: fs, data = wavfile.read('sound2.wav')
         plt.plot(data,color='green')            # fs = sampling frequency = 44.1kHz
         plt.title("Original Audio Plot")
         plt.plot(decrypted,color='red')
         plt.title("Decrypted Audio Plot")

Out[74]: Text(0.5, 1.0, 'Decrypted Audio Plot')
```



```
In [67]: with open('decrypted_rsa.wav', 'wb') as fd:
             fd.write(decrypted)

In [68]: ipd.Audio('sound2.wav')
         #sd.play(encrypted, fs)

Out[68]:
```

In [71]:
```
y1, sr1 = librosa.load('sound2.wav')
D1 = librosa.stft(y1)
S_db1 = librosa.amplitude_to_db(np.abs(D1), ref=np.max)
fig1, ax1 = plt.subplots(figsize=(10, 5))
img1 = librosa.display.specshow(S_db1,
                               x_axis='time',
                               y_axis='log',
                               ax=ax1)
ax1.set_title('Spectogram Decryted Audio', fontsize=20)
fig.colorbar(img1, ax=ax1, format=f'%0.2f')
plt.show()
```



Using AES

In [1]:
```
pip install Crypto.Cipher
```

```
Requirement already satisfied: Crypto.Cipher in c:\users\jia\anaconda3\lib\site-packages (1)
Requirement already satisfied: requests in c:\users\jia\anaconda3\lib\site-packages (from Crypto.Cipher) (2.27.1)
Requirement already satisfied: idna<4,>=2.5 in c:\users\jia\anaconda3\lib\site-packages (from requests->Crypto.Cipher) (3.3)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\users\jia\anaconda3\lib\site-packages (from requests->Crypto.Cipher)
(1.26.9)
Requirement already satisfied: charset-normalizer~=2.0.0 in c:\users\jia\anaconda3\lib\site-packages (from requests->Crypto.Cip
her) (2.0.4)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\jia\anaconda3\lib\site-packages (from requests->Crypto.Cipher) (2
021.10.8)
Note: you may need to restart the kernel to use updated packages.
```

Importing dependencies

In [16]:
```
from scipy.io import wavfile
import numpy as np
import matplotlib.pyplot as plt
import sounddevice as sd
import seaborn as sns
import pandas as pd


import librosa
import librosa.display
import IPython.display as ipd

from itertools import cycle

import random
import string
from Crypto.Cipher import AES
```

In [49]:
```
sns.set_theme(style="white", palette=None)
color_pal = plt.rcParams["axes.prop_cycle"].by_key()["color"]
color_cycle = cycle(plt.rcParams["axes.prop_cycle"].by_key()["color"])
```

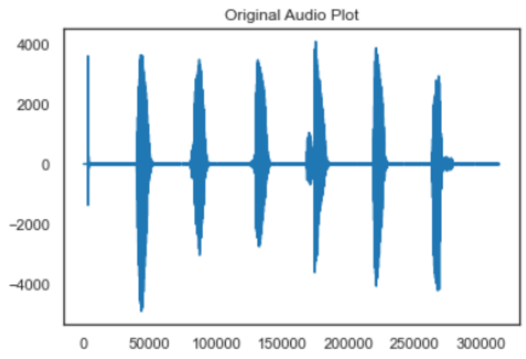
Taking input

```
In [64]: with open('sound1.wav', 'rb') as fd:
             contents = fd.read()
```

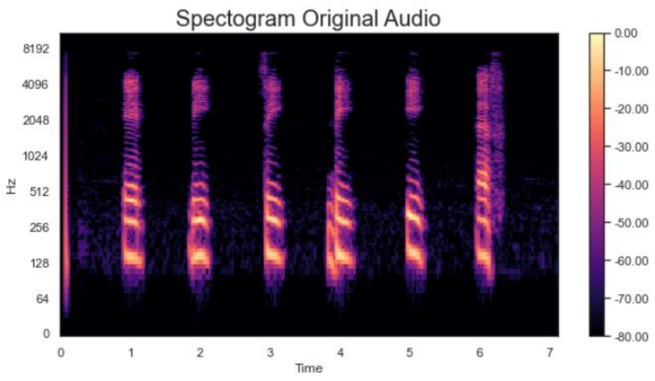
Playing that sound

```
In [65]: #sd.play(data, fs)
         ipd.Audio('sound1.wav')

Out[65]:
```

```
In [66]: fs, data = wavfile.read('sound1.wav')
         plt.plot(data)            # fs = sampling frequency = 44.1kHz
         plt.title("Original Audio Plot")
         data_1 = np.asarray(data, dtype = np.int32)
```



```
In [67]: y, sr = librosa.load('sound1.wav')
         D = librosa.stft(y)
         S_db = librosa.amplitude_to_db(np.abs(D), ref=np.max)
         fig, ax = plt.subplots(figsize=(10, 5))
         img = librosa.display.specshow(S_db,
                                        x_axis='time',
                                        y_axis='log',
                                        ax=ax)
         ax.set_title('Spectogram Original Audio', fontsize=20)
         fig.colorbar(img, ax=ax, format=f'%0.2f')
         plt.show()
```



Getting ready with AES

```
In [54]: AES_KEY = ''.join(random.choice(string.ascii_uppercase + string.ascii_lowercase + string.digits) for x in range(32))

         AES_IV = ''.join(random.choice(string.ascii_uppercase + string.ascii_lowercase + string.digits) for x in range(16))
```

```
In [55]: print("AES Key is ", AES_KEY)
         print("AES Initialization vector is ", AES_IV)

         AES Key is  RxQ0pEqaIeet3D0wjA7QoT9LZyprbmsE
         AES Initialization vector is  nfg5c8FWZBAc7nK0
```

Encrpytion of audio file

```
In [56]: encryptor = AES.new(AES_KEY.encode("utf-8"), AES.MODE_CFB, AES_IV.encode("utf-8"))
         encrypted_audio = encryptor.encrypt(contents)
```

Saving the encrypted file

```
In [57]: with open('encrypted_audio_file.wav', 'wb') as fd:
             fd.write(encrypted_audio)
         print("A file titled 'encrypted_audio_file.wav' is generated which is the encrypted audio to be communicated")

         A file titled 'encrypted_audio_file.wav' is generated which is the encrypted audio to be communicated
```

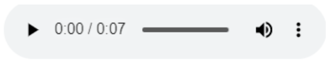
Loading

```
In [58]: with open('encrypted_audio_file.wav', 'rb') as fd:
             contents = fd.read()
```

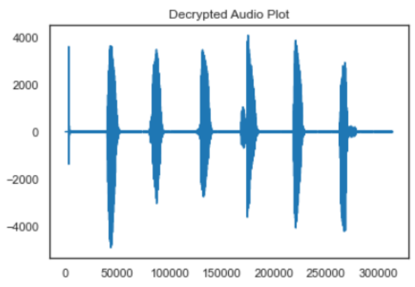
Decryption of data

```
In [59]: decryptor = AES.new(AES_KEY.encode("utf-8"), AES.MODE_CFB, AES_IV.encode("utf-8"))
         decrypted_audio = decryptor.decrypt(contents)
```

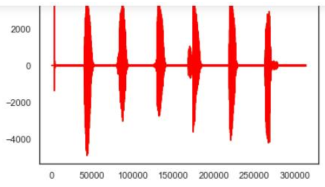
Out[61]:



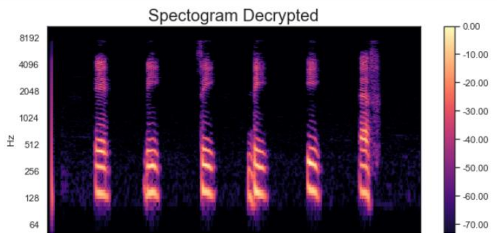
```
In [62]: fs, data = wavfile.read('decrypted_audio_file.wav')
         plt.plot(data)              # fs = sampling frequency = 44.1kHz
         plt.title("Decrypted Audio Plot")
         data_1 = np.asarray(data, dtype = np.int32)
```



```
In [71]: fs, data = wavfile.read('sound1.wav')
         plt.plot(data,color='green')          # fs = sampling frequency = 44.1kHz
         plt.title("Original Audio Plot")
         plt.plot(data,color='red')
         plt.title("Decrypted Audio Plot")

Out[71]: Text(0.5, 1.0, 'Decrypted Audio Plot')
```



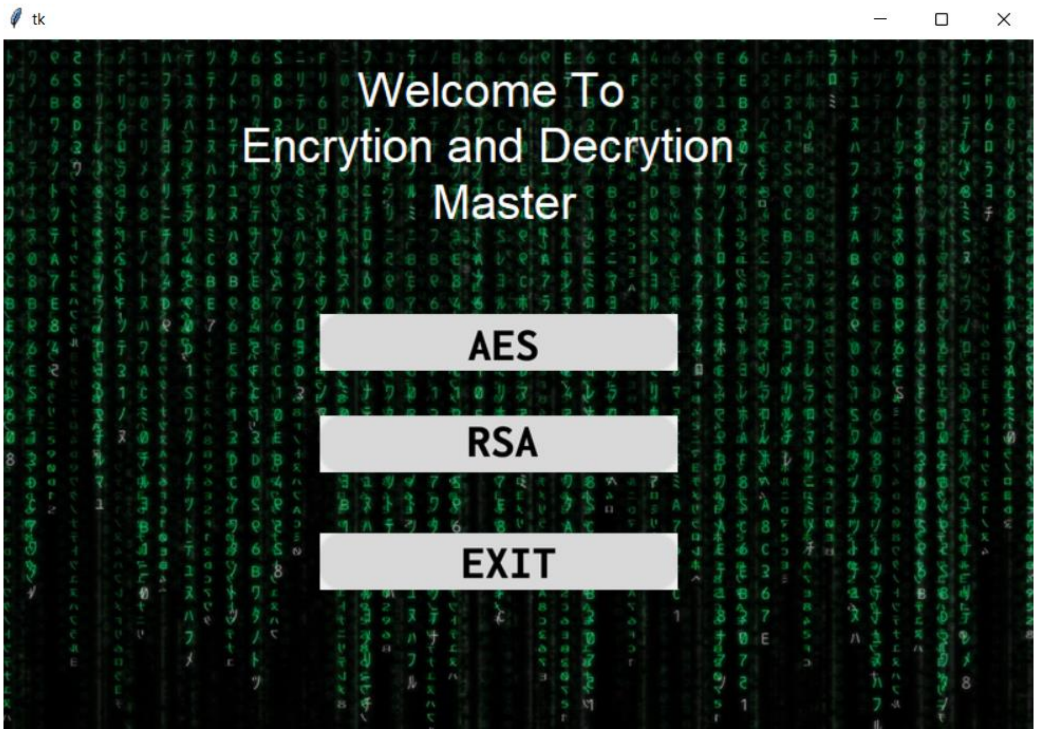
```
In [63]: y1, sr1 = librosa.load('decrypted_audio_file.wav')
         D1 = librosa.stft(y1)
         S_db1 = librosa.amplitude_to_db(np.abs(D1), ref=np.max)
         fig1, ax1 = plt.subplots(figsize=(10, 5))
         img1 = librosa.display.specshow(S_db1,
                                         x_axis='time',
                                         y_axis='log',
                                         ax=ax1)
         ax1.set_title('Spectogram Decrypted', fontsize=20)
         fig.colorbar(img1, ax=ax1, format=f'%0.2f')
         plt.show()
```

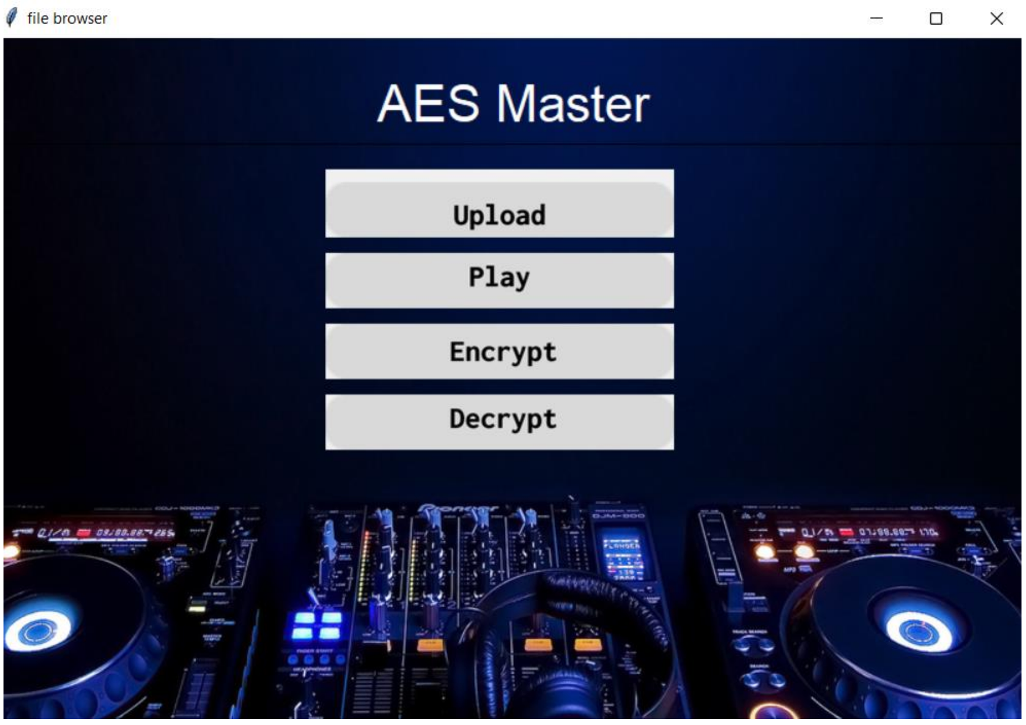


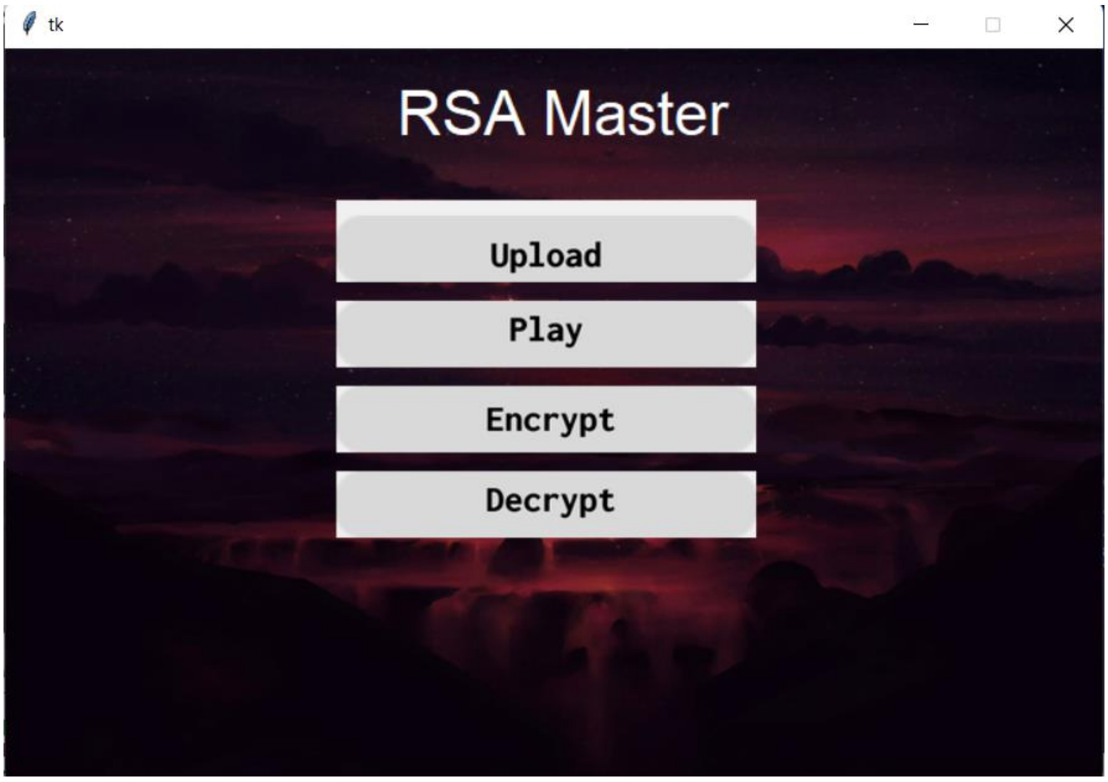
6. GUI

To enable a user friendly and attractive interface, we have designed widgets using the tkinter library of python.

With a click of a button, one can upload the audio file, listen to it, encrypt it and then decrypt and hear it again.







7. Future Work and Conclusion

In our project we have successfully performed encryption and decryption processes to an audio signal using two different techniques. RSA algorithm is the first one while a new suggested technique that depends on the concept of symmetric cryptography (AES) is the second one.

During our work, we found that AES gives better results than RSA method since it produces an audio signal of high quality as the original signal. However, A major issue with AES is that, as a symmetric algorithm, it requires that both the encryptor and the decryptor use the same key. This gives rise to a crucial key management issue – how can that all-important secret key be distributed to perhaps hundreds of recipients around the world without running a huge risk of it being carelessly or deliberately compromised somewhere along the way?

We aim to answer this question by combining the strengths of AES and RSA encryption.

In many modern communication environments, including the internet, the bulk of the data exchanged is encrypted by the speedy AES algorithm. To get the secret key required to decrypt that data, authorized recipients publish a public key while retaining an associated private key that only they know. The sender then uses that public key and RSA to encrypt and transmit to each recipient their own secret AES key, which can be used to decrypt the data.

8. References

- https://www.researchgate.net/publication/340725899_ENCRYPTION_AND_DECRYPTION_OF_AUDIO_SIGNAL_BASED_ON_RSA_ALGORITHN
- https://www.geeksforgeeks.org/advanced-encryption-standard-aes/
- https://www.cloudflare.com/en-gb/learning/ssl/how-does-public-key-encryption-work/
- https://crypto.stackexchange.com/questions/47991/aes-vs-rsa-which-is-stronger-given-two-scenarios