

IoT Network Communication Security Using Steganography And Cryptography System Configuration Manual

MSc Research Project
Cybersecurity

Anugraha Sureshkumar
Student ID: x21154325

School of Computing
National College of Ireland

Supervisor: Muhammad Salahuddin Jawad

National College of Ireland
MSc Project Submission Sheet



School of Computing

Anugraha Sureshkumar

Student Name:

Student ID:x21154325.....

Programme:Cybersecurity..... **Year:** ...2022.....
 Research Project

Module:

Lecturer: Muhammad Salahuddin Jawad

Submission Due Date: ...15/12/2022.....

Project Title: IoT Network Communication Security Using Steganography And Cryptography System

Word Count:807..... **Page Count:**7.....

I hereby certify that the information in this (my submission) is about the research I conducted for this project. All information other than contributions will be fully referenced and listed in the relevant bibliography section at the project's redirect. ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To another author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

SignatureAnugraha.....

Date:14/12/2022.....

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	<input type="checkbox"/>
Attach a Moodle submission receipt of the online submission to each project (including multiple copies).	<input type="checkbox"/>
You must ensure that you retain a HARD COPY of the project for your reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on the computer.	<input type="checkbox"/>

Assignments submitted to the Programme Coordinator's Office must be placed in the assignment box outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Anugraha Sureshkumar
Student ID: x21154325

1 Hardware

To set up a machine, you need a Pentium processor with multiple cores and hyperthreading turned on and at least 16 GB of RAM.

2 IDE

Google colab: Colab lets anyone write and run any Python code through the browser. It works best for machine learning, data analysis, and education.

Open-Source Module

NumPy- NumPy is a Python package that adds support for multidimensional arrays and matrices, coupled with high-level mathematical operations.

Pillow- Python Imaging Library adds functionality for accessing, manipulating, and saving numerous image file formats to the Python programming language. Windows, Mac OS X, and Linux are supported.

PyTorch- PyTorch is a Torch-based machine learning framework used for computer vision and NLP.

Imageio- Python's Imageio package makes it simple to read and write many different types of image data, such as those used in animations, 3D models, and scientific formats.

3 Implementation

We can divide the solution into several module types to initiate the implementation process.

The collection of datasets is based on downloading the dataset from relevant links. The following image can be seen as follows.

```
[ ] DIV2K_valid_HR.zip 100%[=====>] 428.19M 20.0MB/s in 23s
2021-11-15 10:30:14 (18.9 MB/s) - 'DIV2K_valid_HR.zip' saved [448993893/448993893]

Archive: DIV2K_valid_HR.zip
  inflating: val/_/0897.png
  inflating: val/_/0887.png
  inflating: val/_/0806.png
  inflating: val/_/0834.png
  inflating: val/_/0896.png
  inflating: val/_/0881.png
  inflating: val/_/0828.png
  inflating: val/_/0833.png
  inflating: val/_/0877.png
  inflating: val/_/0826.png
  inflating: val/_/0879.png
  inflating: val/_/0812.png
  inflating: val/_/0809.png
  inflating: val/_/0865.png
  inflating: val/_/0882.png
  inflating: val/_/0830.png
  inflating: val/_/0892.png
  inflating: val/_/0859.png
```

Fig 6 Dataset Download

After downloading the dataset, the fundamental deep learning steganography algorithm development is used for training. The training method includes 100 encoding, decoding, and picture verification epochs. The calculation of several metrics, such as accuracy, SSIM, loss, and PSNR, is depicted in the graph below.

After training with the fundamental algorithm, a file was prepared for encoding and decoding the output.



Fig 7 Basic Algorithm Output

This method development appears to attain a maximum of 0.004 bits per pixel, which is insufficient for application.

A better algorithm model is created and trained, with the training outcomes for this model displayed in the figure below.

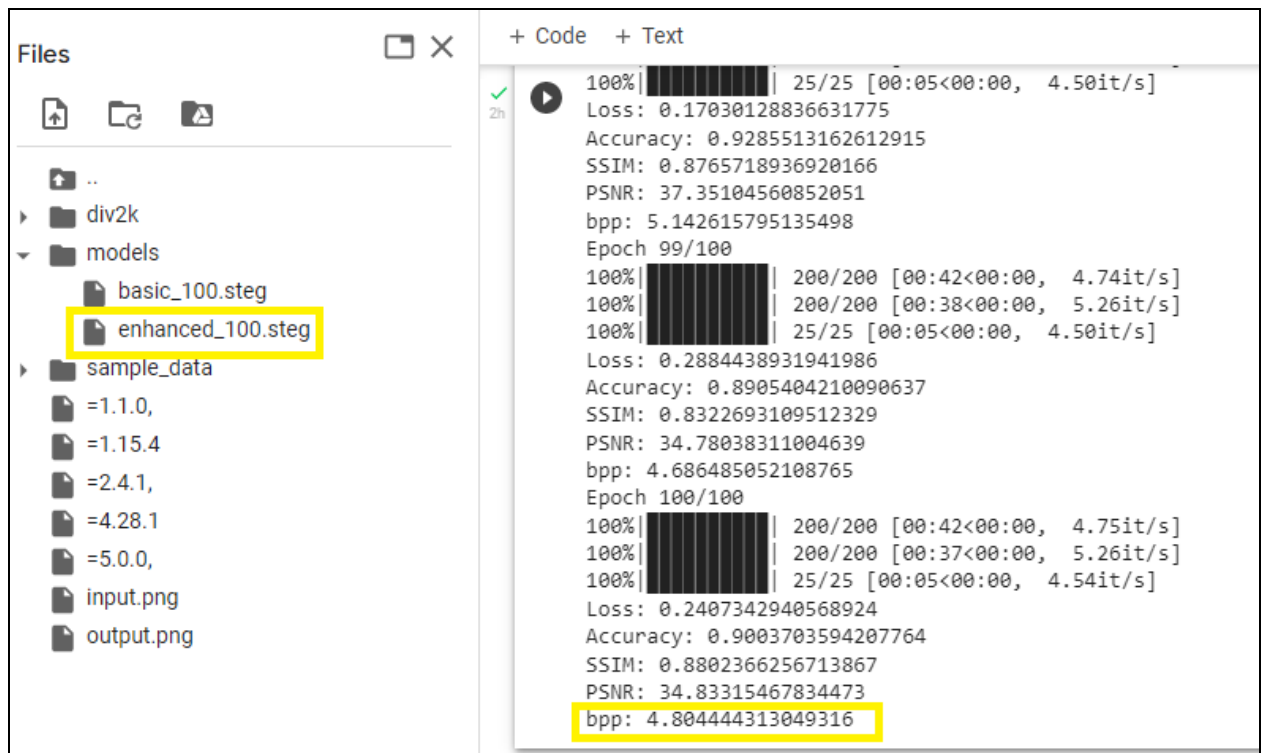


Fig 8 Enhanced Algorithm Output

The diagram above shows that the model file is generated following the training, and parameters are also obtained. The project's objective of getting over 3 bits per pixel has been fully achieved, and the findings show that we received 4.884 bits per pixel, which is feasible in real time.

Now that training is complete, and the files have been developed, the model files are evaluated for their ability to encode and decode picture data.

First, the model file is loaded, and an image file is given to conceal the text input, as shown below.

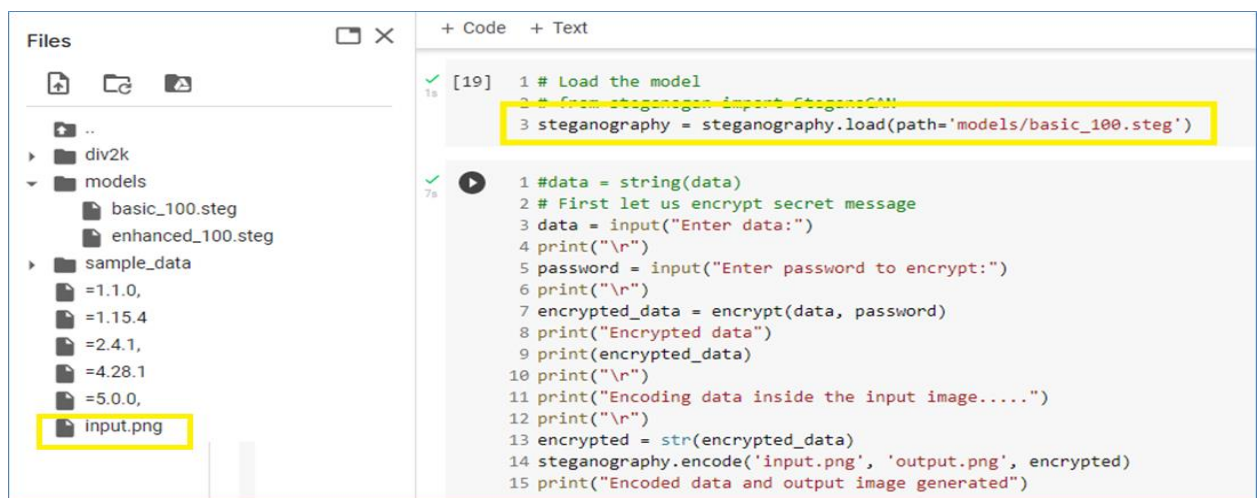


Fig 9 Loading of Model

Following the conversion of the data into the ciphertext, it is concealed inside the source image so that the output can be seen below.

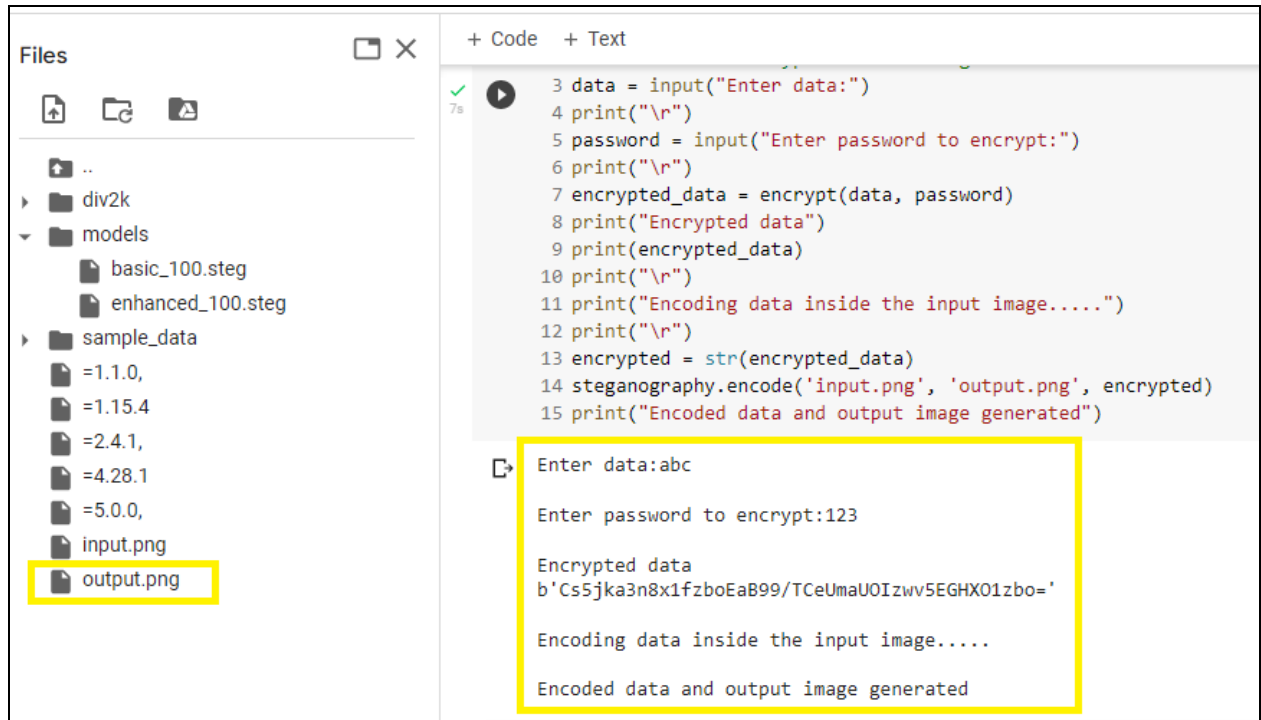


Fig 10 Encrypting And Encoding Data

The figure below shows the decoded output image file.

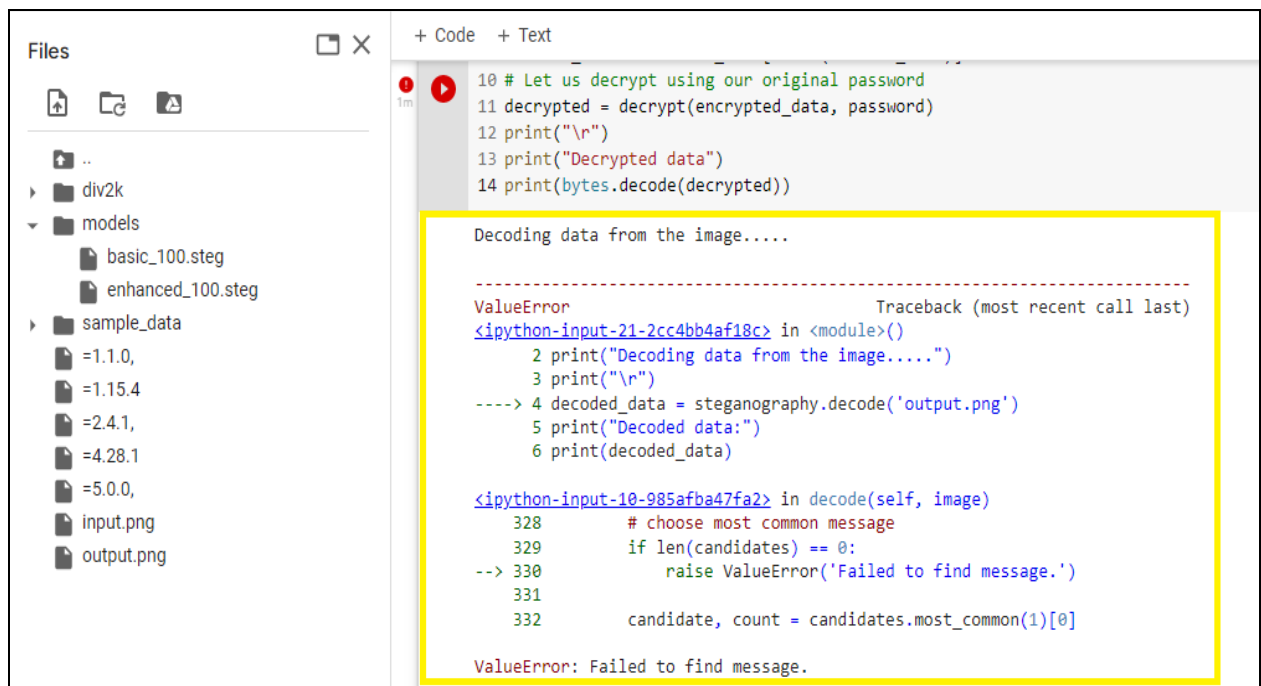
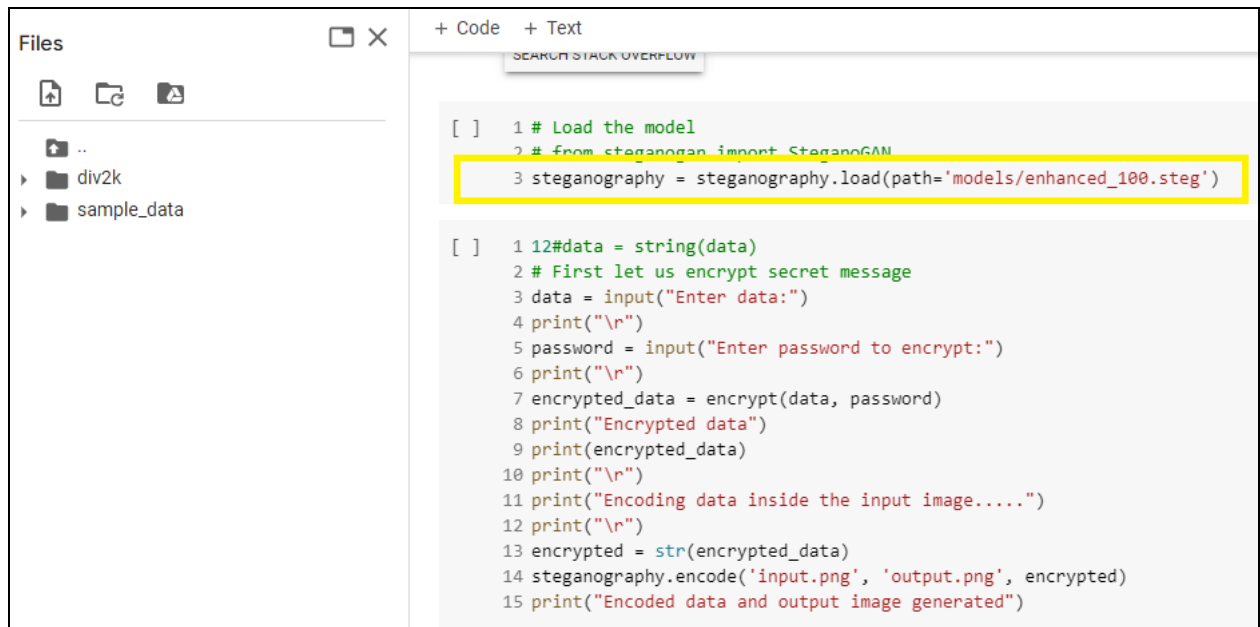


Fig 11 Failed Decoding of Data

No data were identified when an image file was inspected to hide data. The low number of bits per pixel prevented proper data encoding.

The generated model file is loaded for encoding the data, which can be seen below.



The screenshot shows a Jupyter Notebook interface. On the left, a file explorer shows a directory structure with folders 'div2k' and 'sample_data'. The main area contains Python code. The first cell is highlighted with a yellow box and contains the following code:

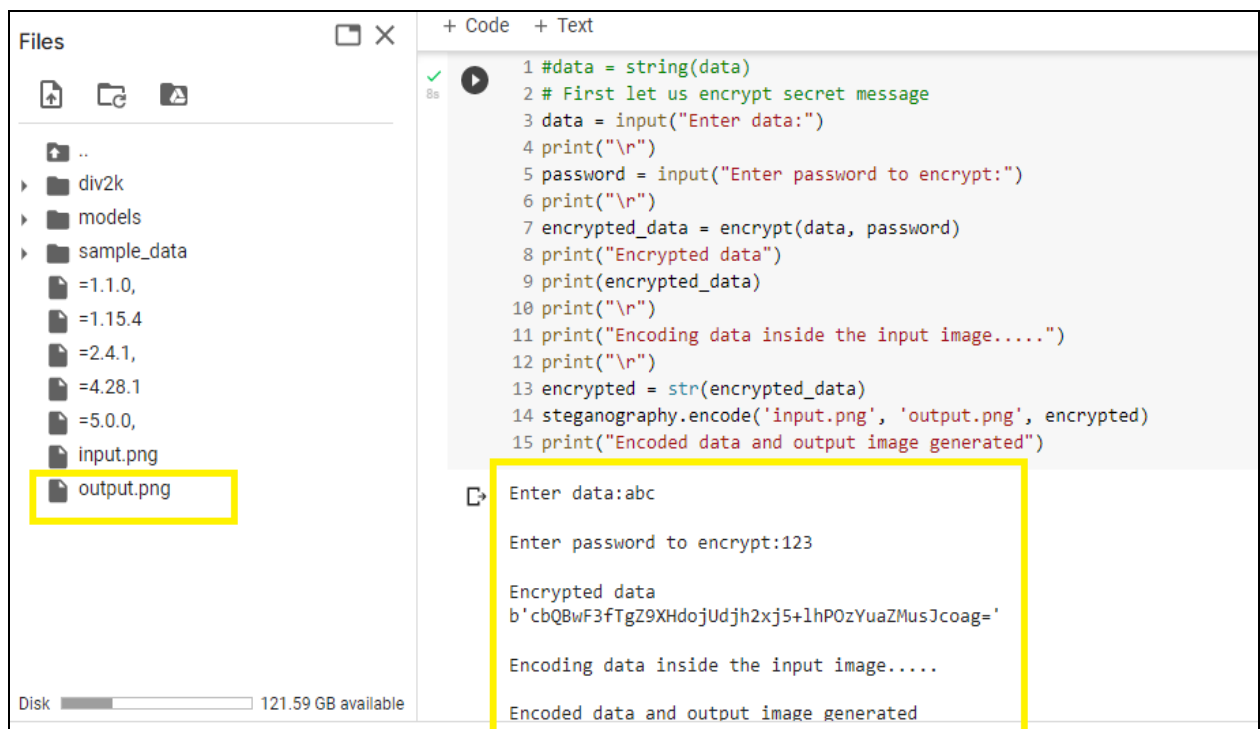
```
[ ] 1 # Load the model
2 # from steganogan import SteganoGAM
3 steganography = steganography.load(path='models/enhanced_100.steg')
```

The second cell contains the following code:

```
[ ] 1 12#data = string(data)
2 # First let us encrypt secret message
3 data = input("Enter data:")
4 print("\r")
5 password = input("Enter password to encrypt:")
6 print("\r")
7 encrypted_data = encrypt(data, password)
8 print("Encrypted data")
9 print(encrypted_data)
10 print("\r")
11 print("Encoding data inside the input image.....")
12 print("\r")
13 encrypted = str(encrypted_data)
14 steganography.encode('input.png', 'output.png', encrypted)
15 print("Encoded data and output image generated")
```

Fig 12 Loading of the Enhanced Model File

After the data encoding procedure, the information is turned to cypher text, as illustrated below.

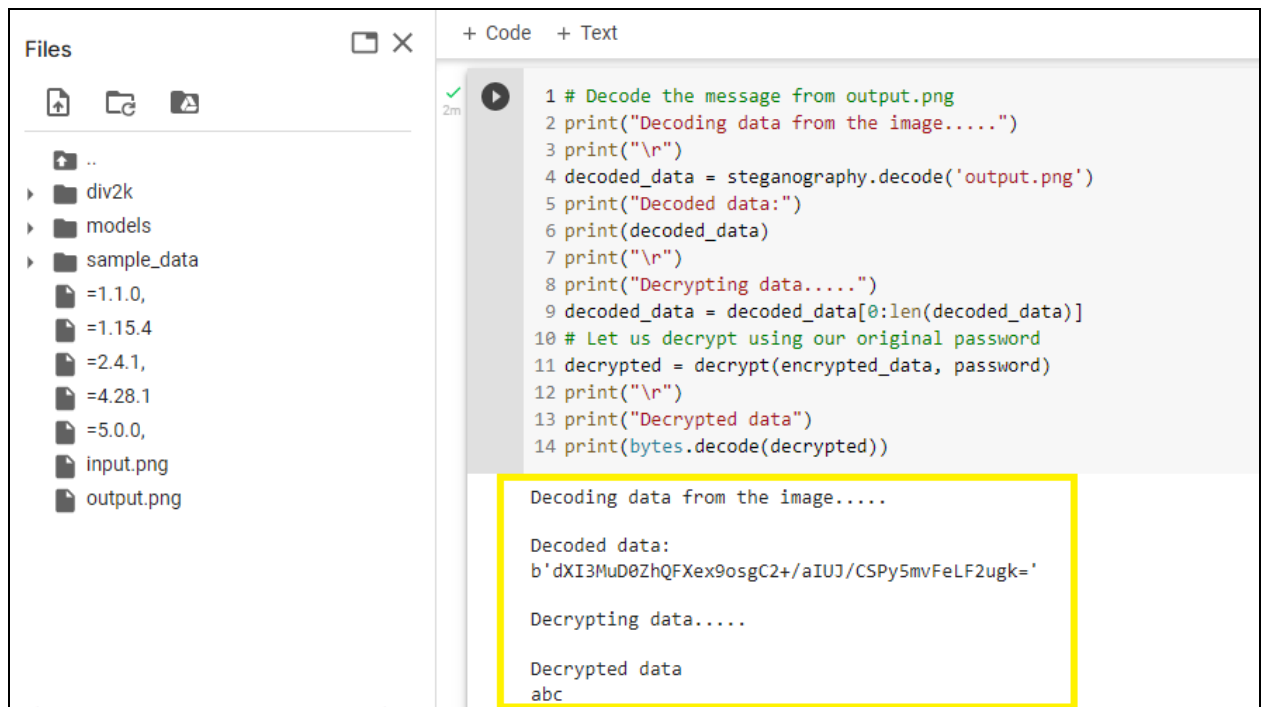


The screenshot shows the same Jupyter Notebook interface as Fig 12. The file explorer on the left now shows a list of files: '=1.1.0', '=1.15.4', '=2.4.1', '=4.28.1', '=5.0.0', 'input.png', and 'output.png'. The 'output.png' file is highlighted with a yellow box. The main area shows the execution of the code from the previous figure. The output of the code is displayed in a yellow box:

```
Enter data:abc
Enter password to encrypt:123
Encrypted data
b'cbQbwF3fTgZ9XHdojUdjh2xj5+lhPOzYuaZMusJcoag='
Encoding data inside the input image.....
Encoded data and output image generated
```

Fig 13 Encrypting And Encoding

Information Decoding recovers data from an image file, as shown below.



```
+ Code + Text
2m
1 # Decode the message from output.png
2 print("Decoding data from the image.....")
3 print("\n")
4 decoded_data = steganography.decode('output.png')
5 print("Decoded data:")
6 print(decoded_data)
7 print("\n")
8 print("Decrypting data.....")
9 decoded_data = decoded_data[0:len(decoded_data)]
10 # Let us decrypt using our original password
11 decrypted = decrypt(encrypted_data, password)
12 print("\n")
13 print("Decrypted data")
14 print(bytes.decode(decrypted))

Decoding data from the image.....

Decoded data:
b'dXI3MuD0ZhQFXex9osgC2+/aIUJ/CSPy5mvFeLF2ugk='

Decrypting data.....

Decrypted data
abc
```

Fig 14 Successful Decrypting And Decoding

References

How to use google colab (2019) *GeeksforGeeks*. Available at:<https://www.geeksforgeeks.org/how-to-use-google-colab/> (Accessed: November 11, 2022).