Configuration Manual

MSc Research Project
MSc in Cyber Security

Pooja Revanna Kumar
Student ID: 20190417

School of Computing
National College of Ireland

Supervisor: Dr Vanessa Ayala Rivera

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Pooja Revanna Kumar		
Student ID:	20190417		
Programme:	MSc in Cyber Security	**Year:**	2021-2022
Module:	Internship		
Lecturer:	Dr Vanessa Ayala Rivera		
Submission Due Date:	16th December 2021		
Project Title:	Cloud Data Security Improvement Using Steganography by Pseudo Random Number Generation (PRNG)		
Word Count:	**509**	**Page Count: 8**	

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Pooja Revanna Kumar

Date: 16th December 2021

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

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

Configuration Manual

Pooja Revanna Kumar
Student ID: 20190417

1 Introduction

In this Manual consists of the process that is carried out to build and execute the application using different tools and software. To perform the image Steganography the PRNG algorithm has been used. Firstly, image will be encoded with the secret data to be hidden in the image. This encoded data is safer to use and pass through the different networks. Then we decode this image to get the secret data hidden in the main image. To perform this code is written in Python and running with the help of Jupyter in the ubuntu system.

2 Configuration of System

2.1 Hardware Configuration

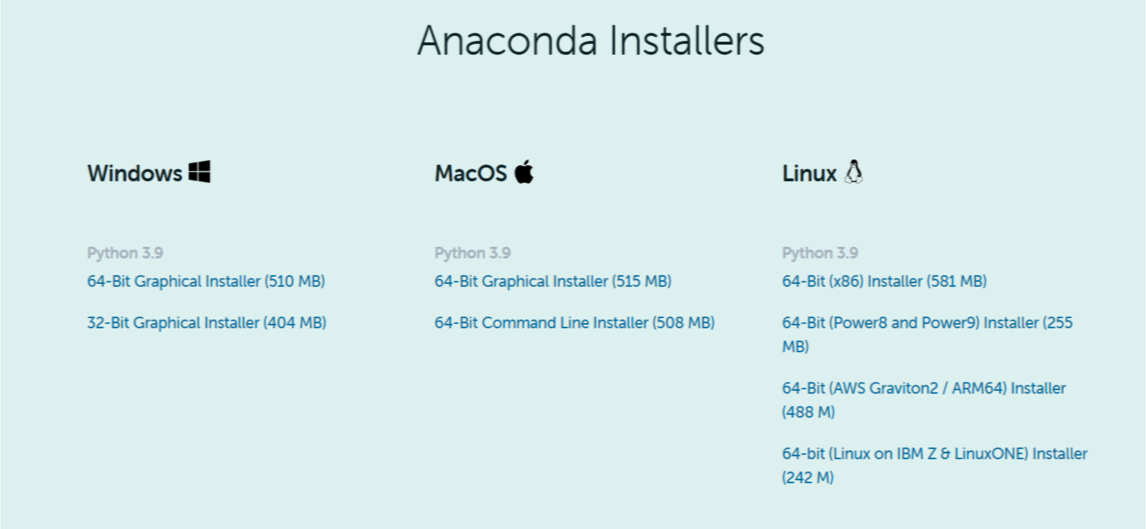
Hardware	Configurations
Processor	Intel i5
OS (Operating System)	Ubuntu 18.04.6
Hard Disk	1TB
System	64-bits
RAM	8GB

2.2 Software Configuration

Software	Name (Version)
Programming Language	Python 3.9
Compiler	Jupyter Notebook 6.2.0
Navigator	Anaconda 1.10.0

In this manual I am added the main methods to install and configure the required applications and tools that is necessary to run the project.

Anaconda needs to be installed from these https://www.anaconda.com/products/individual official link.


Anaconda Installers

Windows

Python 3.9
64-Bit Graphical Installer (510 MB)
32-Bit Graphical Installer (404 MB)

MacOS

Python 3.9
64-Bit Graphical Installer (515 MB)
64-Bit Command Line Installer (508 MB)

Linux

Python 3.9
64-Bit (x86) Installer (581 MB)
64-Bit (Power8 and Power9) Installer (255 MB)
64-Bit (AWS Graviton2 / ARM64) Installer (488 M)
64-bit (Linux on IBM Z & LinuxONE) Installer (242 M)

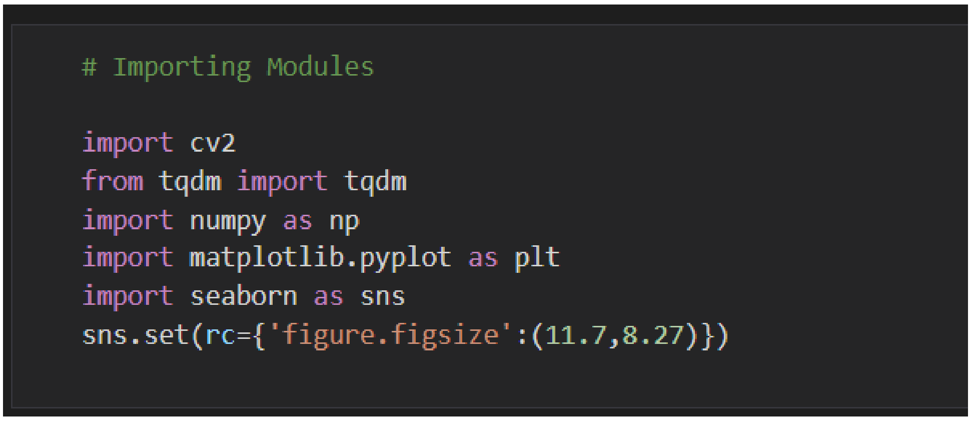
Once the Anaconda is installed it makes easy installation of the Python and Jupiter notebook, just need to run below commands in the console.

```
## Python Installation
sudo apt install python3.9


## Jupyter installation
conda install -c conda-forge jupyterlab
```

3 Functioning

To run this application different libraries are installed in the system. This application code file need build-in libraries from the python. Those are added at the beginning of the code file.



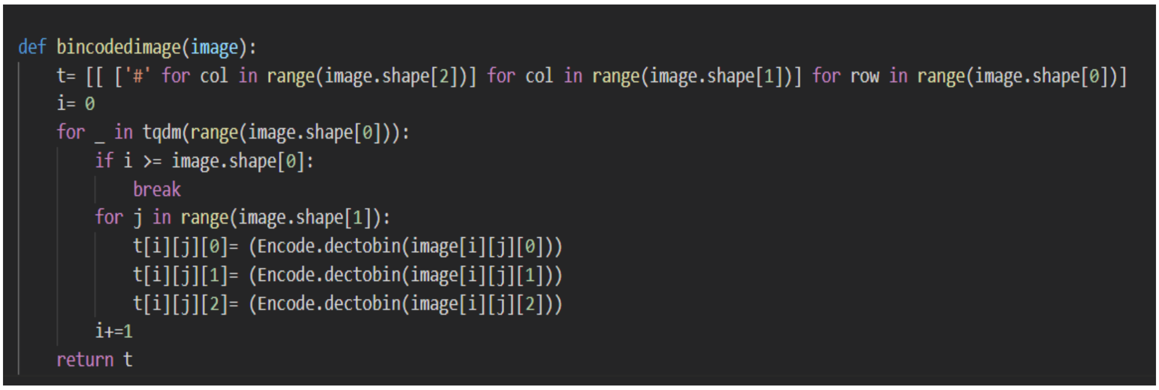
```
# Importing Modules

import cv2
from tqdm import tqdm
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
sns.set(rc={'figure.figsize':(11.7,8.27)})
```

Figure 1: Libraries listed

3.1 Image Encoding

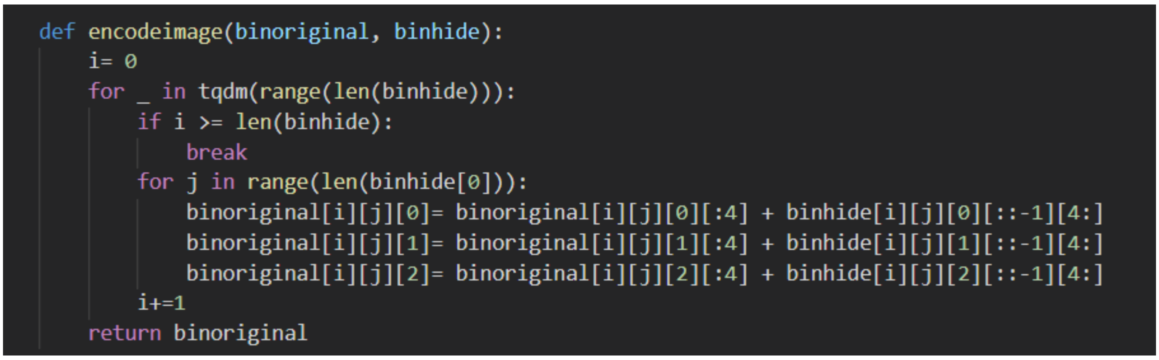
Once we input the images the conversion of image to Binary image will be happen with the help of the bincodedimage(image) function.



```
def bincodedimage(image):
    t= [[ ['#' for col in range(image.shape[2])] for col in range(image.shape[1])] for row in range(image.shape[0])]
    i= 0
    for _ in tqdm(range(image.shape[0])):
        if i >= image.shape[0]:
            break
        for j in range(image.shape[1]):
            t[i][j][0]= (Encode.dectobin(image[i][j][0]))
            t[i][j][1]= (Encode.dectobin(image[i][j][1]))
            t[i][j][2]= (Encode.dectobin(image[i][j][2]))
        i+=1
    return t
```

Figure 2: Binary conversion of the image

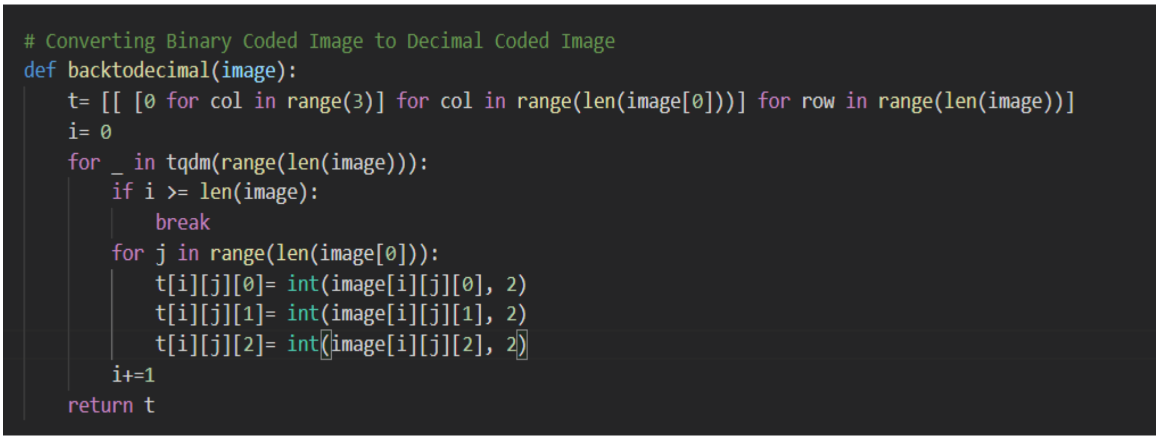
This Binary image will be then encoded using the PRNG to perform image Steganography using the encodeimage(binoriginal, binhide) function.



```
def encodeimage(binoriginal, binhide):
    i= 0
    for _ in tqdm(range(len(binhide))):
        if i >= len(binhide):
            break
        for j in range(len(binhide[0])):
            binoriginal[i][j][0]= binoriginal[i][j][0][:4] + binhide[i][j][0][::-1][4:]
            binoriginal[i][j][1]= binoriginal[i][j][1][:4] + binhide[i][j][1][::-1][4:]
            binoriginal[i][j][2]= binoriginal[i][j][2][:4] + binhide[i][j][2][::-1][4:]
        i+=1
    return binoriginal
```

Figure 3: Image encoding

Then this encoded image will be converted back to decimal code of the image to show for the viewer using backtodecimal(image) function.



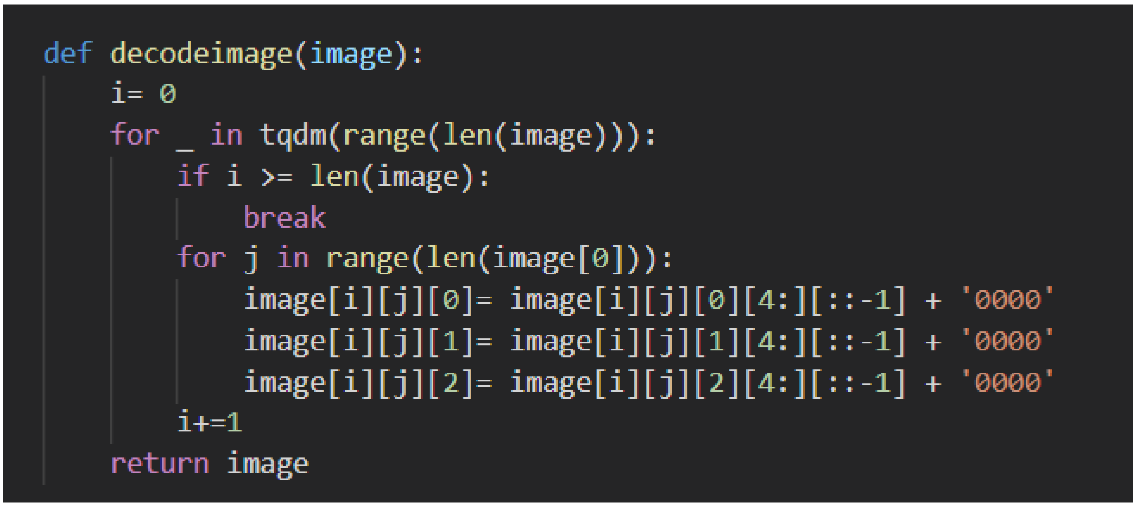
```
# Converting Binary Coded Image to Decimal Coded Image
def backtodecimal(image):
    t= [[ [0 for col in range(3)] for col in range(len(image[0]))] for row in range(len(image))]
    i= 0
    for _ in tqdm(range(len(image))):
        if i >= len(image):
            break
        for j in range(len(image[0])):
            t[i][j][0]= int(image[i][j][0], 2)
            t[i][j][1]= int(image[i][j][1], 2)
            t[i][j][2]= int(image[i][j][2], 2)
        i+=1
    return t
```

Figure 4: Binary-to-Decimal conversion

3.2 Image Decoding

Encoded Image will be passed as the input file here. Once we get the input we need to convert this image to binary using the method bincodedimage(image) function. Using same function showen in figure 1.

This binary image will be send to decodeimage(image) function in this we decode the image and extract the hidden data added in the image.



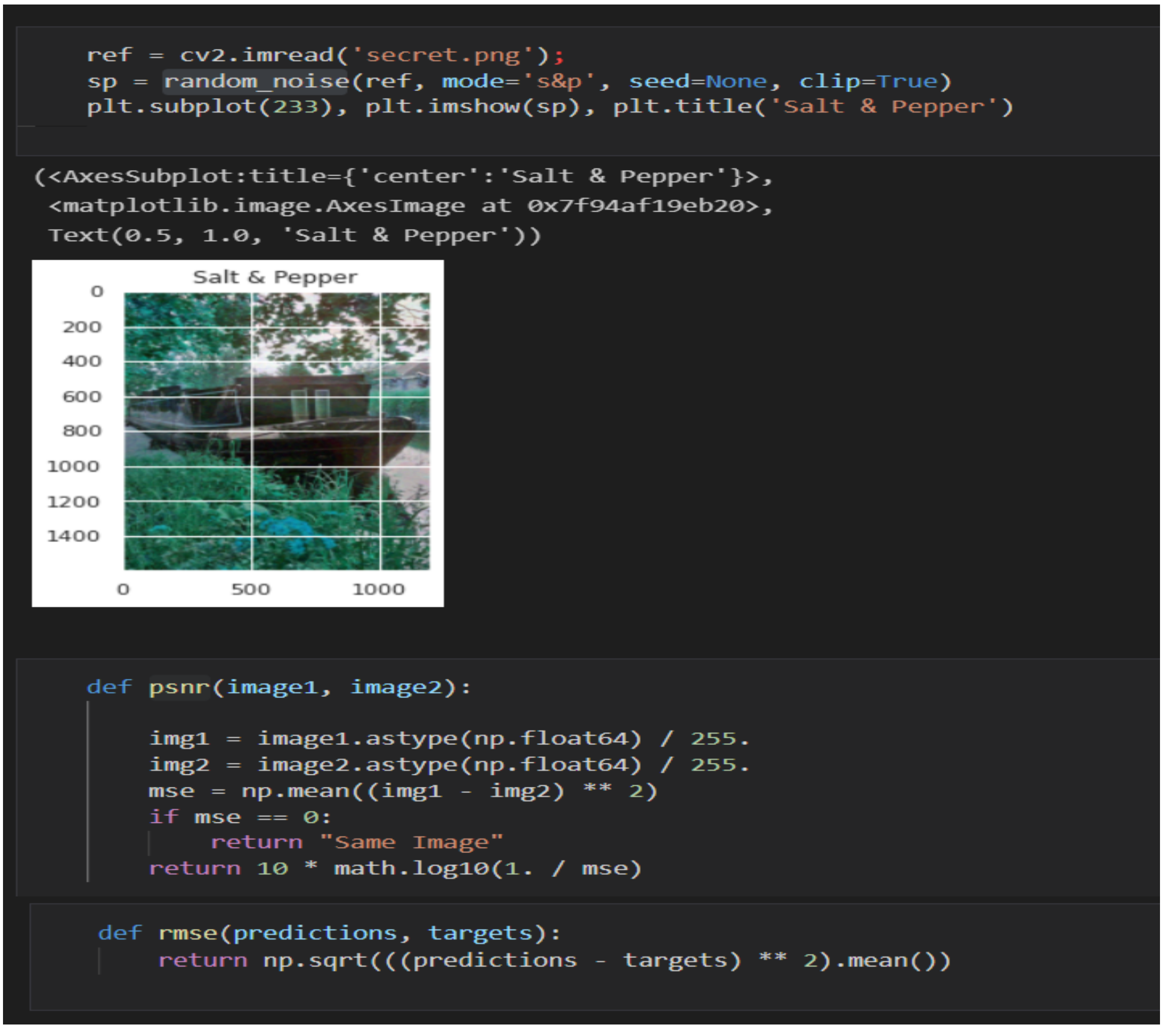
```python
def decodeimage(image):
    i= 0
    for _ in tqdm(range(len(image))):
        if i >= len(image):
            break
        for j in range(len(image[0])):
            image[i][j][0]= image[i][j][0][4:][::-1] + '0000'
            image[i][j][1]= image[i][j][1][4:][::-1] + '0000'
            image[i][j][2]= image[i][j][2][4:][::-1] + '0000'
        i+=1
    return image
```

Figure 5: Decode of Image

Once the decoding is done the output will be in binary image. This needs to converted back to decimal to check the hidden image using backtodecimal(image) function showen in figure 4.

From this we get the data which is hidden in the Cover image and viwer can check data.

To evaluate the cover and secret image I am using PSNR, MSE, SNR methods below gives the respective code of same.



```
ref = cv2.imread('secret.png');
sp = random_noise(ref, mode='s&p', seed=None, clip=True)
plt.subplot(233), plt.imshow(sp), plt.title('Salt & Pepper')
```

```
(<AxesSubplot:title={'center':'Salt & Pepper'}>,
 <matplotlib.image.AxesImage at 0x7f94af19eb20>,
 Text(0.5, 1.0, 'Salt & Pepper'))
```

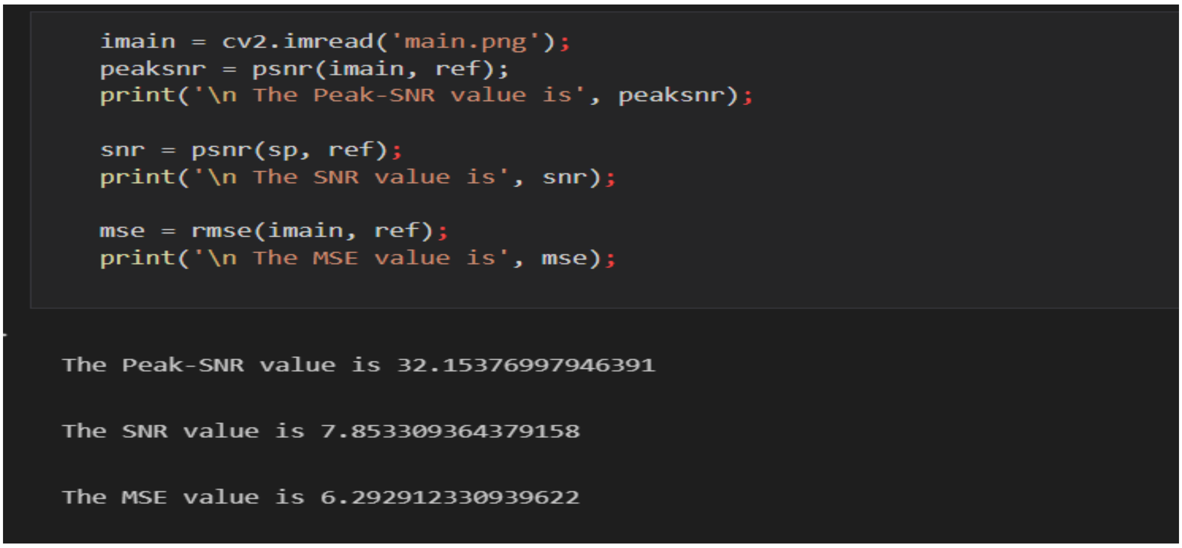
```
def psnr(image1, image2):

    img1 = image1.astype(np.float64) / 255.
    img2 = image2.astype(np.float64) / 255.
    mse = np.mean((img1 - img2) ** 2)
    if mse == 0:
        return "Same Image"
    return 10 * math.log10(1. / mse)
```

```
def rmse(predictions, targets):
    return np.sqrt(((predictions - targets) ** 2).mean())
```

Figure 6: PSNR, SNR and MSE code.

Salt & pepper is added to conduct the SNR analysis and the output of the analysis are shown below.



```
imain = cv2.imread('main.png');
peaksnr = psnr(imain, ref);
print('\n The Peak-SNR value is', peaksnr);

snr = psnr(sp, ref);
print('\n The SNR value is', snr);

mse = rmse(imain, ref);
print('\n The MSE value is', mse);
```

```
 The Peak-SNR value is 32.15376997946391

 The SNR value is 7.853309364379158

 The MSE value is 6.292912330939622
```

Figure 7: Output of PSNR, SNR and MSE

References

Anaconda | Individual Edition [WWW Document], n.d. URL https://www.anaconda.com/products/individual (accessed 12.16.21).

Download Python [WWW Document], n.d. . Python.org. URL https://www.python.org/downloads/ (accessed 12.16.21).