

Title: Detect and Analyze stenographic hidden data within digital files

1. Objective

The objective of this experiment was to detect and analyze steganographic content hidden within a digital file. This lab involved using a **custom-built Python-based steganography detection tool** to load a sample image, scan it for hidden data, and document the findings.

2. Tools and Environment Used

The following tools and environment components were used for this analysis:

- **Analysis Tool:** Custom Steganalysis Tool (Python-based GUI)
- **Key Libraries:** [PySide6, Pillow, matplotlib, numpy, scipy, exifread]
- **Sample File:**

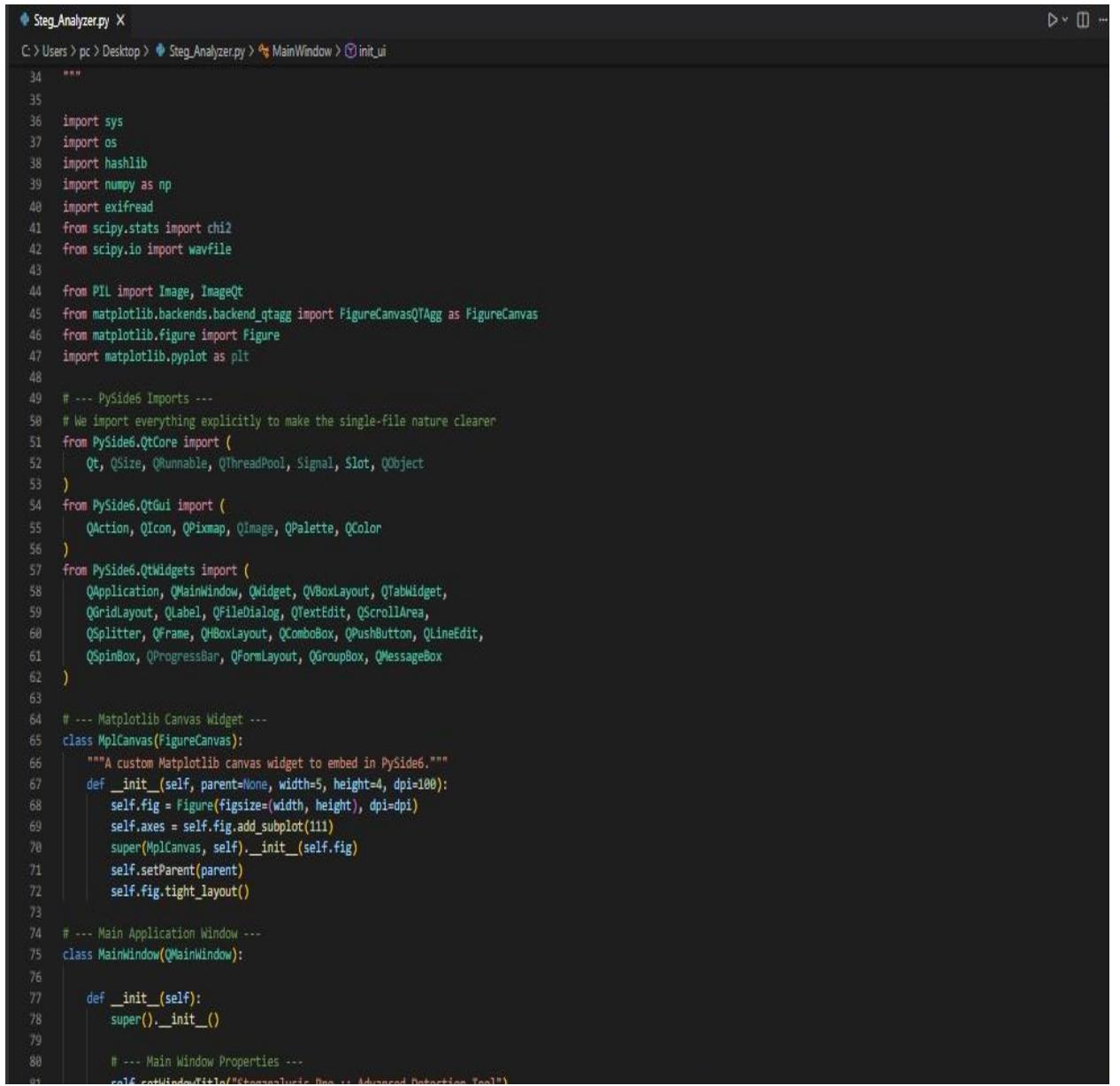


- **Operating System:** Window 10 64-bit
-

3. Step-by-Step Procedure

The following steps were performed using the custom analysis tool:

1. **Tool Launch:** The custom Python-based tool was launched.

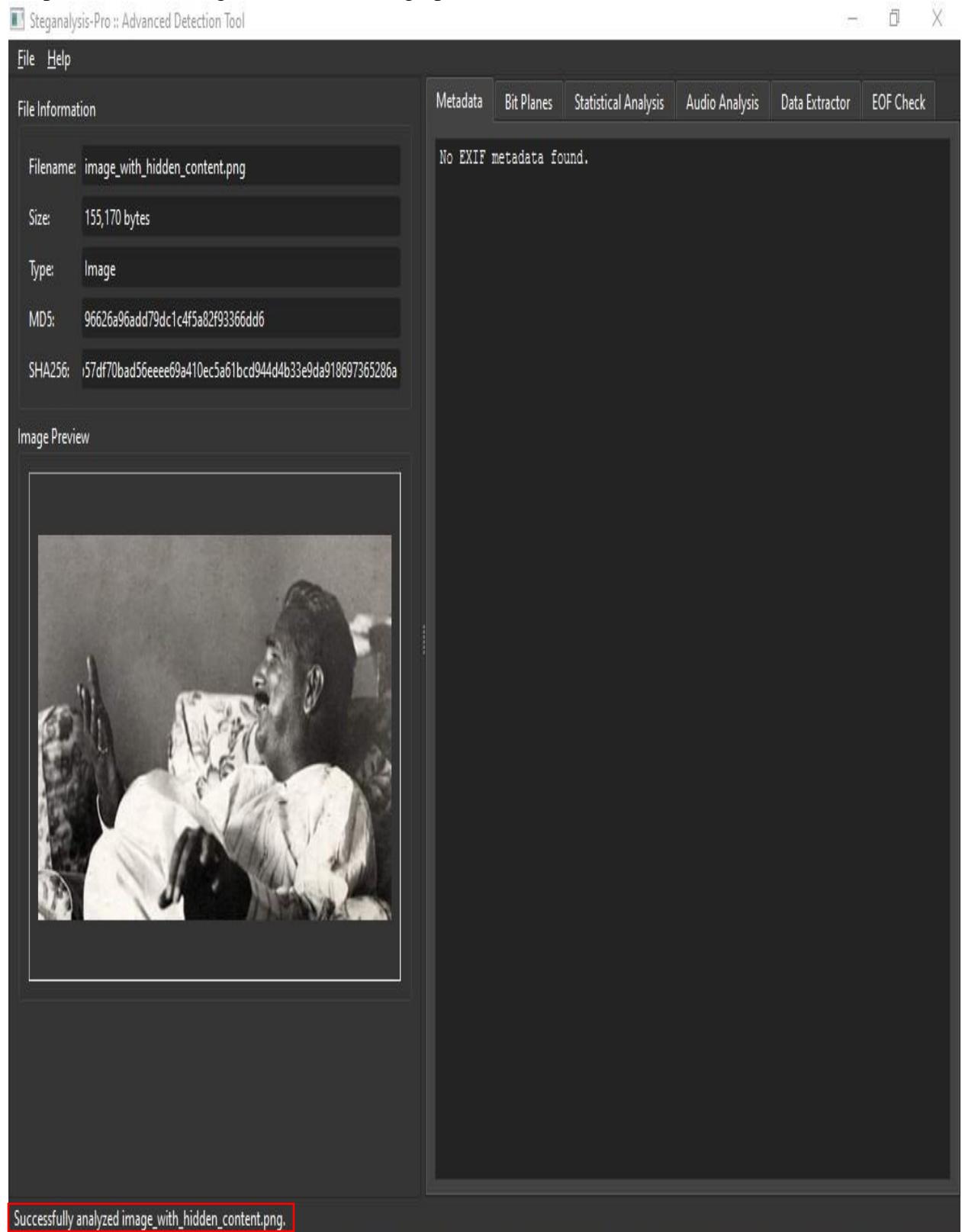


A screenshot of a code editor window titled "Steg_Analyzer.py X". The file path is "C:\Users\pc\Desktop > Steg_Analyzer.py > MainWindow > init_ui". The code is written in Python and defines two classes: `MplCanvas` and `MainWindow`. The `MplCanvas` class inherits from `FigureCanvasQTAgg` and implements a `__init__` method to initialize a Matplotlib figure with a specific size and dpi. The `MainWindow` class inherits from `QMainWindow` and has a `__init__` method that calls the parent's `__init__` method and sets the window title to "Ketanayak's Steg - Advanced Detection Tool". The code uses various standard Python libraries like `sys`, `os`, `hashlib`, `numpy`, `exifread`, `scipy.stats`, `scipy.io`, `PIL`, `matplotlib.backends.backend_qtagg`, `matplotlib.figure`, and `matplotlib.pyplot`, along with PySide6's `QtCore` and `QtWidgets` modules.

```
 34 """
 35 
 36 import sys
 37 import os
 38 import hashlib
 39 import numpy as np
 40 import exifread
 41 from scipy.stats import chi2
 42 from scipy.io import wavfile
 43 
 44 from PIL import Image, ImageQt
 45 from matplotlib.backends.backend_qtagg import FigureCanvasQTAgg as FigureCanvas
 46 from matplotlib.figure import Figure
 47 import matplotlib.pyplot as plt
 48 
 49 # --- PySide6 Imports ---
 50 # We import everything explicitly to make the single-file nature clearer
 51 from PySide6.QtCore import (
 52     Qt, QSize, QRunnable, QThreadPool, Signal, Slot, QObject
 53 )
 54 from PySide6.QtGui import (
 55     QAction, QIcon, QPixmap, QImage, QPalette, QColor
 56 )
 57 from PySide6.QtWidgets import (
 58     QApplication, QMainWindow, QWidget, QVBoxLayout, QTabWidget,
 59     QGridLayout, QLabel, QFileDialog, QTextEdit, QScrollArea,
 60     QSplitter, QFrame, QHBoxLayout, QComboBox, QPushButton, QLineEdit,
 61     QSpinBox, QProgressBar, QFormLayout, QGroupBox, QMessageBox
 62 )
 63 
 64 # --- Matplotlib Canvas Widget ---
 65 class MplCanvas(FigureCanvas):
 66     """A custom Matplotlib canvas widget to embed in PySide6."""
 67     def __init__(self, parent=None, width=5, height=4, dpi=100):
 68         self.fig = Figure(figsize=(width, height), dpi=dpi)
 69         self.axes = self.fig.add_subplot(111)
 70         super(MplCanvas, self).__init__(self.fig)
 71         self.setParent(parent)
 72         self.fig.tight_layout()
 73 
 74 # --- Main Application Window ---
 75 class MainWindow(QMainWindow):
 76 
 77     def __init__(self):
 78         super().__init__()
 79 
 80         # --- Main Window Properties ---
 81         self.setWindowTitle("Ketanayak's Steg - Advanced Detection Tool")
```

2. **File Loading:** The image file was loaded into the tool using its "Load Image" functionality.

3. **Analysis:** The tool's "Analyze" or "Detect" function was executed. This initiated the script to scan the image for hidden stenographic data.



4. **Results Review:** The tool's output (e.g., in a text box, status bar, or pop-up) was reviewed to determine if hidden content was found.

S Steganalysis-Pro :: Advanced Detection Tool

File Help

File Information

Filename: image_with_hidden_content.png

Size: 155,690 bytes

Type: Image

MD5: 46f2e7fcab9bcd18e636c8cd52f7e0d7

SHA256: 3891fcfed0bcfb465ab79a56e2364e1e5b7f9b88330bd349c073bc6d4

Image Preview



Extractor Options

Metadata Bit Planes Statistical Analysis Audio Analysis Data Extractor EOF Check

Bits per sample: 1 (LSB) Max bytes to extract: 1024 Extract Data

--- Extracted Printable Text (first 1024 bytes) ---

137 YOU GOT IT! This is my secret lab message!
Koi Qabil Ho Tou Hum Shane Kayi Dete Hain
Dhundne Waly ko Duniya Bhee Nayi Dete

Hain86_ar87\$nhq#m7_,\$r9+H+rHr7\$nhqRU*U[RGr87_8#9_n9qmrG\$+r6nHm#\$
q#I\$9\$nIr6nFhr9\$97\$Fq@\$
nF6=q#nG6n9nFr7#mn8m#+UI[Rr7m\$8\$H\$mn8nHF76gGF73n7\$rHrH7#I*TUH_Hmn6m\$H66,\$n7\$
6g\$7#m#m#g#8r9G6GnHrG\$#m#n87\$6r9H#n6r7\$g@
r8qmRT\$876q\$9#rFnHnF796#9jn6FI#9\$TUZI[jnI\$rg\$nhnF\$
9,\$m#gGI#In67q#g9qmgnGnHmq,#F#grI#nF\$G9UTURm\$M,\$q\$7r7#I#nHm\$8n7\$ri#p@
m,q8m#8UUI[jVmnnFq#qr8mq#I\$erFhr7mI#qr6n7qIm#7rI#rI\$9mr6r6\$GnF6Hm\$g\$Rn\$
mn6m\$8nInHnI\$8m#I\$8n87\$#qmm\$Fn8

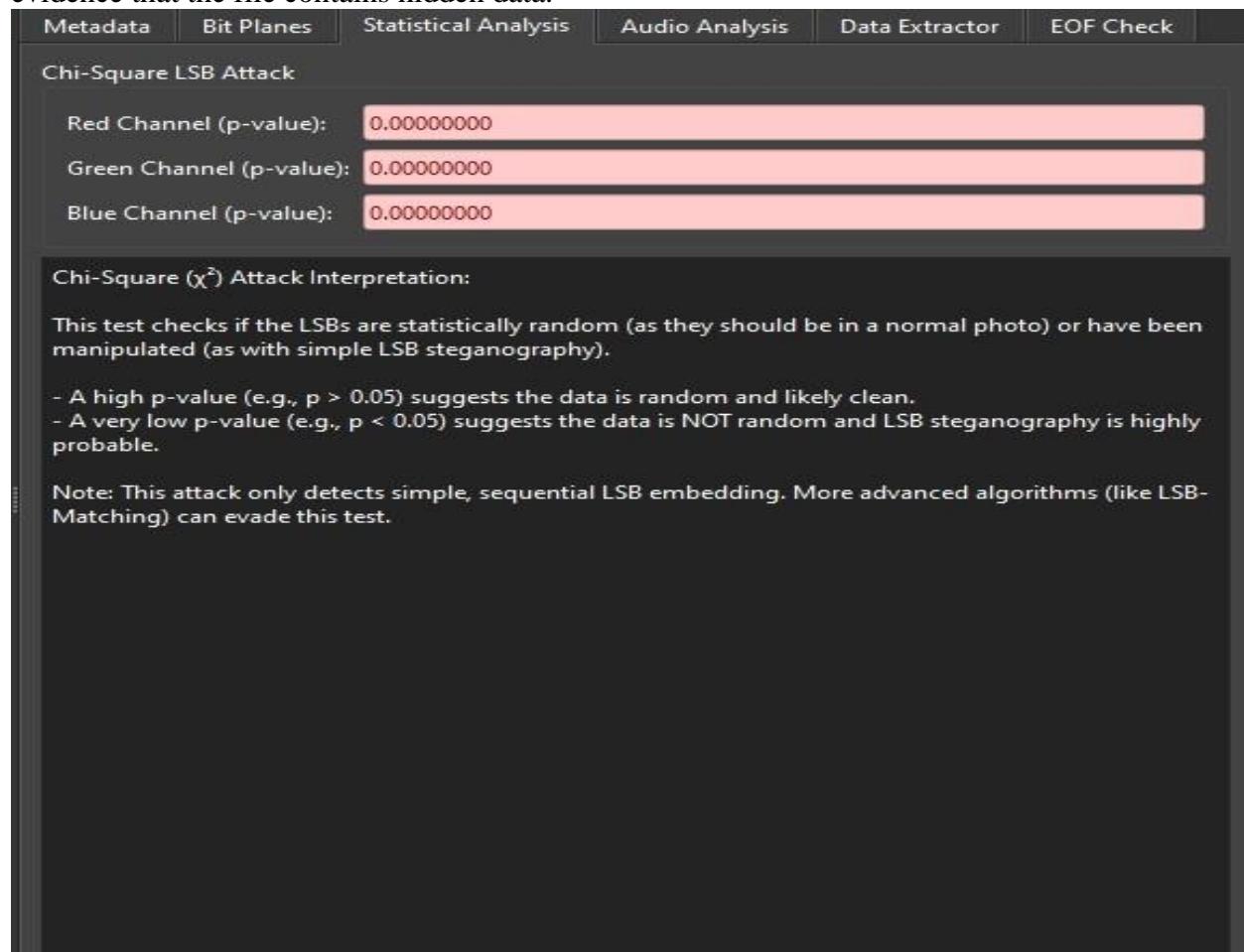
Extraction complete. Rendering data...

4. Results and Findings

The analysis of the sample image file using the custom tool yielded the following results:

- **Detection Status:** Hidden content was found and successfully extracted.
- **Hidden Content Type:** You Got It! This is my secret message.
- **Analysis Details:** The tool performed a Chi-Square (χ^2) attack on the image's Least Significant Bits (LSBs) to detect statistical anomalies. The p-value results strongly indicate data manipulation:
 - Red Channel (p-value): 0.00000000
 - Green Channel (p-value): 0.00000000
 - Blue Channel (p-value): 0.00000000

As per the test's interpretation, a p-value below 0.05 suggests the data's LSBs are not statistically random and that LSB steganography is "highly probable"⁵. These results provide clear forensic evidence that the file contains hidden data.



5. Conclusion

This experiment successfully demonstrated the application of a custom Python tool to detect and analyze steganography. The tool not only extracted a hidden text message ("You Got It! This is my secret message.") But also provided statistical proof of data manipulation.

The Chi-Square analysis confirmed that the image's LSBs were not random (p -value = 0.0), a clear indicator of LSB steganography. This lab confirms the custom tool is effective for both extracting hidden content and forensically identifying the methods used to embed it.