**Project Report**

***Steganography Tool with AES Encryption***

**Introduction**

With the increasing exchange of digital information, protecting sensitive data has become a crucial challenge. While encryption techniques are widely used to secure the **content** of a message, they do not conceal the **existence** of that message. This limitation often draws the attention of unauthorized entities who may attempt to crack the encryption. To overcome this problem, researchers and practitioners have turned towards **steganography**, which focuses on concealing data inside seemingly harmless digital files.

This project aims to design and develop a **Steganography Tool** that hides either text or complete files within image files using the **Least Significant Bit (LSB)** technique. To add a second layer of security, the system incorporates **AES-GCM (Advanced Encryption Standard in Galois Counter Mode)** encryption. By combining these two approaches, the tool not only hides the data but also encrypts it, ensuring confidentiality and integrity.

The tool has been implemented with a **Tkinter-based Graphical User Interface (GUI)** for desktop usage and has also been deployed as an **online tool** for wider accessibility. Users can securely embed and extract information without requiring specialized technical skills.

**Abstract**

The project demonstrates the integration of **cryptography** and **steganography** into a single user-friendly system. The steganographic part is based on **LSB manipulation**, which embeds data bits into the least significant bits of pixel values in an image. Since these changes are imperceptible to the human eye, the carrier image looks visually unchanged.

To protect the embedded information from unauthorized access, the system optionally applies **AES-GCM encryption**. This provides both **confidentiality** (ensuring the data cannot be read without the correct key) and **integrity** (detecting whether data has been tampered with).

The project was implemented in **Python 3**, using **Pillow** for image processing, **PyCryptodome** for encryption, and **Tkinter** for the GUI. The tool supports both **text embedding** and **file embedding**, with drag-and-drop support for convenience. It has been tested with different image sizes and works best with **lossless formats** such as PNG and BMP.

A working version of the project is also hosted online for demonstration:  
👉 [https://advanced-steganograp-f3g1.bolt.host](https://advanced-steganograp-f3g1.bolt.host/)

**Tools Used**

* **Python 3** – Core programming language for implementation.
* **Pillow (PIL fork)** – Handles image reading, pixel manipulation, and saving.
* **PyCryptodome** – Provides AES-GCM encryption and decryption functions.
* **Tkinter** – GUI framework for building desktop applications.
* **tkinterdnd2** – Adds drag-and-drop functionality to the interface.
* **Bolt Host** – Cloud platform used for deploying the online version.
* **GitHub** – Source code hosting, version control, and collaboration.

**Steps Involved in Building the Project**

1. **Requirement Analysis**
   * Defined the core objective: allow users to hide text/files inside images securely.
   * Identified the need for encryption in addition to steganography.
2. **Design Phase**
   * Designed system flow: Input → Encryption (optional) → Embedding → Output image.
   * Created two modes: **Text Mode** (for short messages) and **File Mode** (for complete files).
   * Planned GUI layout with options to select images, input text, choose files, and enter passphrases.
3. **Implementation**
   * Implemented **LSB encoding and decoding** for embedding and extraction.
   * Integrated **AES-GCM encryption/decryption** for secure data handling.
   * Developed the GUI with buttons, dialogs, and drag-and-drop.
4. **Testing**
   * Embedded and extracted data across images of varying sizes.
   * Verified AES encryption by testing with correct and incorrect passphrases.
   * Checked compatibility with PNG and BMP images (lossless).
5. **Deployment**
   * Finalized Python desktop application.
   * Deployed an online demo using **Bolt Host** to allow browser-based access.

**Conclusion**

The **Steganography Tool** developed in this project demonstrates how **data hiding** and **data encryption** can be effectively combined to provide secure communication. By embedding text or files within an image, and further encrypting them with AES-GCM, the tool ensures that hidden information remains undetectable and unreadable without authorization.

The graphical interface makes the system user-friendly, while the online deployment expands accessibility beyond desktop usage.