HIDE AND SEEK: EXPLORING STEGANOGRAPHY

OBJECTIVE:

The main aim of this project would be a succesful implementation of a system capable of steganographical encoding of given data. We`ll be using web-technologies to implement the system, so that if the project is succesful and robust enough; it can be deployed on a open server in near future which can be accesible by the commons.

The system on succesful completion will also sport multiple levels of stegano- & crypto-graphical applications, ranging from basic to full-fledged security. Besides the main aim, our side aim also will be to make the system modular enough, so that it can be customized to fit the users` need.

Apart from all the targets/aims listed above, we will also not forget the basic requirements of a good enough software application; Reliability, Portability, Robustness, Modular, Easy-To-Use, Secure & Satisfying.

PROBLEM DEFINITION:

We do not aim to make the system to solve some particular problems from a particular source, however our problem statement can be generalized to the following:

“ While classical cryptography is about concealing the content of messages, steganography is about concealing their existence. Steganography must not be confused with cryptography, where we transform the message so as to make its meaning obscure to a person who intercepts it. Such protection is often not enough. The detection of enciphered message traffic between a soldier and a hostile government, or between a known drug smuggler and someone not yet under suspicion has obvious implications. In some applications, it is enough to hide the identity of either the sender or the recipient of the message, rather than its very existence; But with Steganography, one has got an upper hand, as no one(at-least the interceptors) would/can look for data that they doubt exists. "

METHODOLOGY:

Classical steganography concerns itself with ways of embedding a secret message (which might be a copyright mark, a covert communication, or a serial number) in a cover message (such as a video film, an audio recording, or computer code).

The embedding is typically parameterized by a key; without knowledge of this key (or a related one) it is difficult for a third party to detect or remove the embedded material. Once the cover object has material embedded in it, it is called a stego object. Thus, for example, we might embed a mark in a covertext giving a stegotext, or embed a text in a cover image giving a stego-image; and so on. (This terminology was agreed at the First International Workshop on Information Hiding.)

In order to accomplish the system`s objectives a number different known methods of steganography will be implemented. The system can accept both message(data to be hidden) and cover data, and will be able to embed the message in the cover data. For extraction of the embedded message the system would require a stego-object and its key.

The system may also on choice encrypt the message before embedding it, thereby adding an extra layer of security. There are different methods how we can embed the message in the cover data, one of them being simple polyglotting i.e, merging two files into one each with their own header information. This example gives us an idea that data can change based on the perception.

The data secretly hidden or sent is called concealed information, while the digital file hosting is known as host or carrier or cover.

IMAGES AS CARRIERS:

There are different approaches to steganography, the most common one being images. We will discuss a number of techniques used for image steganography. A number of computer programs are available that will embed information in an image. Some of them just set the least significant bits of the image pixels to the bits of the embedded information . Information embedded in this way may be invisible to the human eye but is trivial for an alert third party to detect and remove.

Slightly better systems assume that both sender and receiver share a secret key and use a conventional cryptographic keystream generator to expand this into a long pseu- dorandom keystream. The keystream is then used to select pixels or sound samples in which the bits of the ciphertext are embedded.

Not every pixel may be suitable for encoding ciphertext: changes to pixels in large fields of monochrome color, or that lie on sharply defined boundaries, might be visible. Therefore some systems have an algorithm that determines whether a candidate pixel can be used by checking that the variance in luminosity of the surrounding pixels is neither very high (as on a boundary) nor very low (as in a monochrome field). Wherever a pixel passes this test, we can tweak its least significant bit to embed a bit of our message.

AUDIO FILE CARRIERS:

Many methods for the concealment of the audio file have been developed. However, and without taking into account the technique, it is important to consider the following elements: perceptive invisibility (the concealed information must go unnoticed to the audience; in this case, hearing); robustness (resistance present in the utilized technique when facing the carrier`s manipulation, such as compression or filtering); and capacity (amount of information that can be hidden in the host without affecting the rest of its characteristics).

A simple technique for hiding data is through the modification of the Least Significant Bits . It requires the modification of those bits whose contribution to the carrier signal are less valuable. Although it constitutes a simple technique, and the changes in the host file are hardly perceptible to the human ear, it is sensible to alterations, which translates into low robustness. It is important to mention that various techniques have been developed, based on the LSB alteration, so as to improve the quantity of bits that can be modified , which allows for larger quantities of data to be concealed.

Another technique is known as spread spectrum. In this technique the audio is coded throughout the whole frequency range. This allows the audio to be transmitted over different frequencies that vary according to the spectrum extension method employed. Those techniques that utilize this type of methodology are the safest method transmitting a hidden message in an audio file; nevertheless, random noise can lead to data loss.

DATA ORDERING:

Ordering data that does not have an ordering constraint is often an effective method of steganography. Each permutation of a set of objects can be mapped to a positive integer. This mapping can then be used to encode hidden data by altering the order of objects that are not considered ordered by the carrier medium. While this technique generally does not change the information quality, hidden data can easily be lost if the medium is encoded again. For example, if I have a GIF whose color map contains hidden data, I can open the GIF in my favorite graphics-editing package and save it again. Visually, the resulting GIF should be identical to the original, but the ordering of the color map may have been lost.

SOFTWARE/TOOLS FOR IMPLEMENTATION:

As stated earlier we are going to use web technologies for implementation of the system, some of the technologies or tools we are going to use are stated below..

Frontend:

- HTML
- CSS
- JavaScript

Backend:

- Ruby
- Ruby on Rails Framework

- HTTP server(Apache or nginx)

HTML, CSS, JavaScript:

We are going to use these for the frontend as they are the standards in frontend web. The reason for using these are that they can be easily maintained and are also reliable. JavaScript is used to add interactivity to the system

Ruby:

Ruby is A dynamic, open source programming language with a focus on simplicity and productivity. It has an elegant syntax that is natural to read and easy to write.

Ruby on Rails:

Rails is a web application development framework written in the Ruby programming language. It is designed to make programming web applications easier by making assumptions about what every developer needs to get started. It allows you to write less code while accomplishing more than many other languages and frameworks. Experienced Rails developers also report that it makes web application development more fun.

Rails is opinionated software. It makes the assumption that there is a "best" way to do things, and it's designed to encourage that way - and in some cases to discourage alternatives. If you learn "The Rails Way" you'll probably discover a tremendous increase in productivity. If you persist in bringing old habits from other languages to your Rails development, and trying to use patterns you learned elsewhere, you may have a less happy experience.

The Rails philosophy includes two major guiding principles:

- **Don't Repeat Yourself:** DRY is a principle of software development which states that "Every piece of knowledge must have a single, unambiguous, authoritative representation within a system." By not writing the same information over and over again, our code is more maintainable, more extensible, and less buggy.
- **Convention Over Configuration:** Rails has opinions about the best way to do many things in a web application, and defaults to this set of conventions, rather than require that you specify minutiae through endless configuration files.

REFERENCES:

- Ross J. Anderson and Fabien A. P. Petitcolas “ On the Limits of Steganography “ , IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, VOL. 16, NO. 4, MAY 1998 [pg. 474]