CERIAS

The Center for Education and Research in Information Assurance and Security

A Critical Look at Steganographic Investigations

Michael Burgess, Advisor: Dr. Marcus Rogers

Project Overview

Steganography, the practice of hiding hidden information in plain sight, has been a threat for hundreds of years in different medium. In today's world, hiding files and information digitally inside of images, audio, programs, and most any other file-type could pose a very real danger when two individuals are communicating without anyone knowing they are doing so. Researcher Michael Burgess designed a process and made a tool that takes any file and injects (and extracts) it inside of any mono wave file, as long as the wave file is approximately double the size of the target hidden file. The resulting file has the same size and properties of the original wave file, and no difference can be heard by the human ear. Alongside, all current anti-stego tools have a difficult time detecting that anything is hidden. With a tool as simple as this being able to pass by detection, steganographic investigations need to be taken much more seriously, and include more discovery of these tools rather than the files themselves.

Method and Techniques

Files can be deconstructed into a listing of Hex Code.
Using HexEdit 4.0, one can convert any file into this type of list and export it as a .txt file.
The "encode.py" program takes this text file and converts it into a list of hex elements, which can be then converted into a list of decimal values ranging between 0 and 255.

Take for example an abbreviated array:

- [AA, 9F, 2B, 1C, ...]
- = [170, 159, 43, 28, ...]

Properties of a typical mono wave file:

- 44,000 Samples per Second
- Each Sample is defined by a value between [-32,768, +32,768).
  - Therefore there are 65,536 different states per sample.

Therefore, for a mono wave file, one can create an array of samples. For example:

- [220, 1138, -1099, 20230, ...]

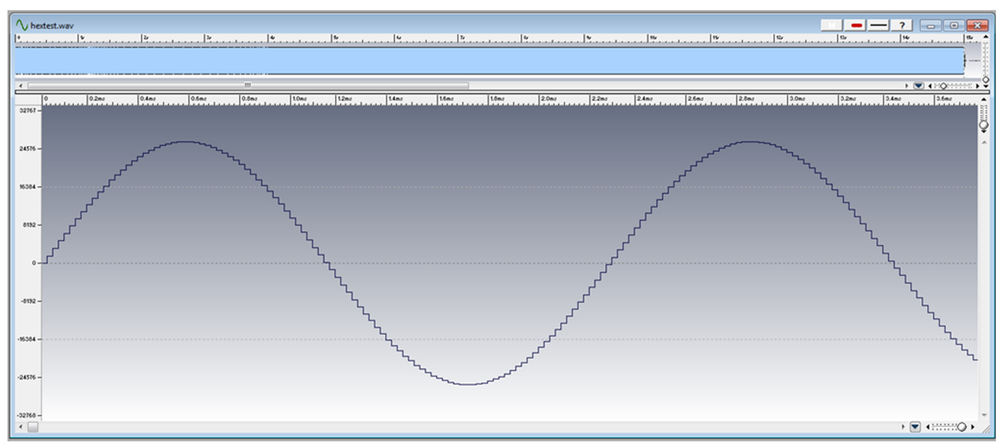
If one were to normalize these values by making each element divisible by 256, they create a blank carrier that, when exported, still sounds the same to the human ear as the unedited array.

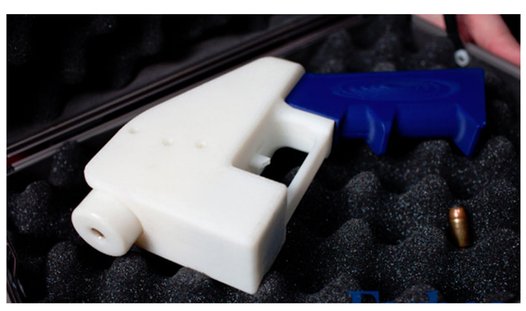
The program does this by using "x – x modulo 256" for each entry x. However, it first counts the hidden file's array length and only executes the formula for that many elements. Thus, making the wave file:

- [220-220, 1138-114, -1099-181, 20230-6, ... + unedited wave samples]
- = [0, 1024, -1280, 20224, ...]

To inject the hidden message, the program adds the hidden file array to the normalized wave array:

- = [0+170, 1024 + 159, -1280 + 43, 20224 + 28, ... + unedited wave samples]





The program then repackages this array as a wave file, including the same properties as the original input wave, and exports the new steganographic wave file. The program also prints the length of the embedded file, which is used during the decoding process. Although the example may look to be very different from the original, be sure to consider the total number of 65,536 possible values.

The program "decode.py" is similar. The program asks for the stego-wave file and the length of the message. The program then strips the wave file of that many samples, takes modulo 256 of each, converts that to hex, and exports it to a text file that is compatible with HexEditor. Hexeditor can then import this text file and repackage it as an arbitrary file. The user then can rename the extension to create an exact copy of the original hidden file.

Conclusion

The goal of this project is not to give a new tool for steganography, but rather to show how easy it is to make a tool that eludes all of the anti-stego programs currently in use, and show just how dangerous this could become in the future. It is doubtful that any tool in particular could ever completely find or even eliminate steganography. Other strategies should be used such as detecting stego programs and artifacts left on a machine from these programs. Overall, the threat of steganography should be taken much more seriously in investigative procedures.

ORIG

STEG