Steganography examples

Now days we have more than 100 steganography programs ranging from free downloads to commercial products are available. In this section we will see some simple steganography example. In this example we used airport map. A GIF map with size about 11, 067 bytes of Vermont and Burlington airport in WAV, GIF and JPEG files are used.

In the first example we used least significant bit substitution which employs Gif-It-Up and hides information in GIF image using Nelsonsoft program . In the above Fig 2 We employed Gif-It-Up to insert airport map in Washington DC GIF image. After applying steganography on the Gif the file changes to 677, 733 bytes in length and 256 unique colors, here we can observe slight difference in

size because size of original carrier is 632, 778 bytes in length with 249 unique colors. By applying color extension the file size is changed because of distortion and is larger than original size and if the color extension is not employed the differences are less noticeable.

Figure 3 clearly shows the carrier file's palettes from Washington mall before and after message insertion. Gif-It-Up modifies the color palette like all least significant bit insertion programs which act on 8 bit color images and finally ends up with many duplicate color pairs.

For Lossy compression and JPEG files , Allan Latham designed JP Hide-&-Seek (JPHS) . JPEG algorithm uses discrete cosine transform coefficients , to overwrite these transform coefficients JP Hide-&-Seek uses least significant bit . In some cases for least significant bit encryption and randomization we use Blowfish crypto algorithm. An example is shown in fig 4 where airport map is embedded in JPEG file. The size of the original carrier file is 207, 244 bytes and it contains 224, 274 unique colors but after embedding, the file size changes to 207, 275 bytes in size and also contains 227, 870 unique colors. Discrete cosine transforms and 24 bit color coding is used in JPEG file so no color palette for it.

One more technique for hiding information is using S-Tools. Andy brown designed S-Tools which can be used to hide information inside BMP, GIF and WAV files. Least significant bit substitute is used by S-Tools for pulse code modulation and lossless compression such as 8 or 24- substitution in the files. S-Tools may employs password for least significant bit randomization. To encrypt data it uses Triple-DES, DES, MDC or IDEA. In Fig 5 it shows two images comparing the signal level between a WAV carrier file before and after the map was hidden. The steganography WAV file is 178, 298 bytes in length whereas the original WAV file is 178, 544 bytes in length. Because of the smaller size of the figure it is hard to see in detail but slight differences are noticeable at the beginning and end of the audio sample. Low intensity portions of signal can be avoided by using steganography tools which have built in intelligence. Because of the larger sizes audio files are well suited for information hiding and makes hidden items difficult to find.

In this article we used Gif-It-Up, JPHS, and S-Tools for example purposes only. They are easy to use and perform their tasks well moreover they are free. There are many other programs used for hiding information in Tag Image File Format, Portable Network Graphics, JPEG, MP3, BMP, GIF, WAV, Paintbrush and many other file types. StegoArchive. Com is one site where we can get some good steganography software and those software’s are compatible with many operating systems that are available in present market .

Up to now we had discussed only about image and audio files it doesn’t mean steganography media is limited to only image and audio files. For information hiding we can use some other file types which also have similar characteristics, but we need to exploit those files for information hiding. One such kind of file type is Hydan. It was developed by Rakan El Khalil. And it can be used to conceal text messages in . exe files, Red Hat Linux, OpenBSD files, FreeBSD files and NetBSD. It hides the information by defining the sets of functionally equivalent instructions and the redundancy in i386 instruction set helps in hiding , simply like a grammar-based mimicry like ADD instructions are a 0 bit and SUB instructions are 1 bit. In every 110-instruction bytes we can hide one message byte by using this program and application file’s size also remains same . We can also employ Blowfish encryption.

Detecting Steganography

Prisoner’s Problem, which is originally used to describe the cryptography scenario and now it, is often used for describing the steganography. Alice and Bob are the two prisoners involved in this problem both are locked in two separate prison cells and they are allowed to send messages to each other. And they wish to exchange some hidden plan, but they have a problem with William, the warden, who can read all of their messages. And they know that warden will terminate their secret communication if he discovers it.

Warden William can act in both passive and active mode.

Passive Mode: Warden William examines each message and depending upon his ability to detect the hidden message he decides to forward the message or not.

Active Mode: Warden William can modify messages in Active mode. Actually the warden might modify all the messages to disrupt any covert channel as an attempt, which makes Alice and Bob to use a very robust steganography method

Depending upon how complex the algorithm is and the knowledge on steganography shows the difficulty of the warden’s task. Warden doesn’t know anything about the steganography method in steganography model, which was employed by Alice and Bob. Also this is poor assumption on prisoner’s part because security through obscurity only works some time and also particularly disastrous if applied to cryptography. Simply it can be described as digital forensics model searching a Website or some hard drive for possible use of steganography.

It is assumed in secret key steganography that William may know the steganography algorithm but he does not know the stego/crypto key which was employed by the prisoners. And it is consistent with the assumption that we made, a user of cryptography should make, according to Kerckhoff’s Principle , Secrecy of algorithm is not only the security factor but key management too. For practise, this may also be too strong of an assumption, because the complete information would access to the carrier file source.

The detection of steganography by an unknown user is called Steganalysis. Some research articles appeared on steganalysis before late 90s. The main intention of steganalysis is to detect or estimate the hidden information by observing the data transfer and no assumptions are made about the steg algorithm. The detection of hidden data may not sufficient, the hidden data must be extracted and also we must disable the hidden message. It makes recipient hard to extract the data. Also they should alter the hidden message to send misinformation to the recipient. For evidence gathering, steganography detection and extraction is sufficient. During investigation of criminal groups, alteration or destruction of those hidden information also come under legitimate law enforcement goals.

Depending upon the information known, Steganalysis techniques are classified into several types. They are:

- Steganography-only attack: For analysis we use steganography medium in this type of attack.
- Known carrier attack: For these types of attacks we use steganograpy media and carrier for analysis.
- Known message attack: The name itself shows that known message is attacked; here the known message is nothing but hidden message.
- Chosen steganography attack: In chosen steganography algorithm the algorithm used and the steganography mediums are known.
- Chosen message attack: We use a known message and algorithm for creating steganography media
- Known steganography attack: The known products are carrier, steganography algorithm and medium.

Depending upon operating mode and for digital media, steganography methods can be broadly classified in to image domain or transform domain. In Image domain tools it uses bit by bit manipulations to hide the message in the carrier which is least significant bit insertion. In transform domain tools we also manipulate the steganography algorithm and hiding the information in actual transformations such as JPEG’s transforms cosine coefficients.

To inspect the steganography media and carrier is one simple approach. Many steganography tools works mainly in the image domain to choose message bits in the carrier. Also it is easy to hide the message in the area of louder sound or brighter color; the program may not seek those areas out. Thus, visual inspection may be sufficient to cast suspicion on a steganography medium.

One more approach is to find structural oddities. And this approach needs manipulation. In some palette based images we have to use more number of colors most of them must be duplicate so that it appear twice and it can be easily identified in least significant bit . So this is the reason why we use palette images for least significant bit insertion. So now we can hide information by modifying the order of duplicate colors in palette which causes structural change.

These techniques generally alter the statistics of the carrier. The longer hidden messages will alter the statistics of carrier more than shorter ones. When the analyst is working in blind and if he wants to detect hidden messages then statistical analysis is commonly employed. Statistical steganalysis contains large body of work.

For audio and video files we can perform Statistical analysis, the analysis shows whether the properties of those files are in expected norm or deviated. To measure the amount of redundant information and distortion in medium we can use first-order statistics like means, chi-square (X2) and variances. A prediction can be done by using these measures , whether the contents have been modified or suspicious.

Because of high degree of randomness (ones and zeros appear with equal likelihood), Encrypting the hidden message became much harder. To avoid detection, some steganography algorithms take pains to preserve the carrier file's first order statistics which makes steganalysis much harder.

Recovery of the hidden message adds much complexity when compared to detecting the hidden message. To recover the message we require the knowledge of encryption key, crypto algorithm or an estimate of the message length.

To make our analysis more straightforward we can choose carrier file type-specific algorithms.   One such type is JPEG. Generally JPEG is used in different types of algorithms which made it to receive a lot of research attention. If the file is modified using JPEG compression method , then the hidden information in that file is much easy to detect and it becomes poor carrier medium if we use simple least significant bit insertion.

JPEG files are used in several algorithms to hide information, and all work differently. The different types of algorithms used are F5 , JSteg , JP Hide & Seek and OutGuess. Hamming code is used in F5, the hidden data in least significant bits are used in JSteg and OutGuess uses first order statistics . Finally a random process is used in Hide&Seek to select least significant bits.

If the statistical tests are advanced then it uses linear analysis, very high order statistics . Also some times it has to depend on statistics by wavelet and markov are. A detailed discussion is done on these tests in steganography tools sections. And moreover this topic is out of scope and it doesn’t need much attention.

Steganalysis is signature based. Now a days many detection systems are using signatures. Many antivirus detection software and intrusion detection are using signature based. Also everyone started using Anomaly based and that practise is slowly beginning to emerge. The new practises are more flexible and they response time is much lesser when compared to old practises.. Although the old systems are accurate there response time is more. Blind steganography is one such practise which is robust and responds much faster. Also it shows difference between steg images using statistics (wavelet decomposition) and scales.

Steganalysis is not only applicable on image and audio files, we ca use these type of statistical analysis for various types of files. The original size of carrier is retained in Hydan program. But if we use instructions that are functionally equivalent the size may be varied. The newer versions of hydan will mislead the analyst with maintaining integrity about the statistics.

Steganography can be employed anywhere and those algorithms are used on any type of files , law enforcement doesn’t know any clue about usage of these algorithms and it’s tough task to identify where it was used. Some detection tools are already available which are much accurate. We will discuss about those tools in coming sections.

Steganalysis not only finds embedded steganography but also helps the upcoming writers by giving knowledge on different algorithms. And those algorithms are all about avoiding detection.

Tools For Steganography Detection

Now we will discuss some examples of available software that can be used to detect the steganography programs and this section has stated focus on practicing computer forensics examiner . Also apart from detection software we will also discuss about suspect carrier files and disrupt hidden messages using steganography tools. This section is all about available capabilities but not survey of available tools. Many steganalysis programs were listed in StegoArchive. com.

The duty of forensic examiners is to detect the suspect computer for steganography software. They use different types of software which directly impact steganalysis. For Example JP Hide & Seek might direct the examiner more closely to JPEG files and S-Tools might direct attention on BMP, GIF and WAV files. However, research proved that many steganography detection programs worked at there best. And also there are some clues for this type of steganography that was employed in very first place. Also finding steganography software on suspect computer having files with hidden messages would give rise to the suspicion about the hidden messages.

To detect the presence of stegenography software, WetStone Technologies' Gargoyle (formerly StegoDetect) software can be used . By using search and compare method he employed some hash sets and some data sets of all of the files in the steg s/w distributions. He used them for comparing with the hashes of the files that are subject to search. In Figure 6 we can see the output in target directory where steganography programs are stored. Some nefarious software like instant messaging, password hacking also key logging and some viruses like Trojan horse can be detected using Gargoyle data sets.

Guidance Software's EnCase and AccessData's Forensic Tool (AFT) kit can use the hashsets of National Software Reference Library, Maresware and HashKeeper for a large variety of software. During computer forensic analysis these data first excludes hashes of some known good files from all the search indexes. And to import these hash sets Gargoyle can also be used.

Sometimes it is hard to detect the steganography software. One reason for this problem is with the very smaller size of the s/w coupled with removable media which has high storage capacity. For example S-Tools, this requires not more than 500 KB of disk space and without additional installation it can be executed directly from a USB memory key or floppy. For such cases we cannot find any remnants of the program on the hard drive.

Finding the possible carrier files is one more important function of Steganography detection software. And this detection software helps in providing some clues, which are used by steganography algorithm to hide information in the suspect file. This helps analyst to attempt recovery of the hidden information.

Niels provos stegdetect is one of the most commonly used detection program. Using steganography schemes like Invisible secrets, F5, JPHide and JSteg, Stegdetect helps in finding hidden information in JPEG images. In Figure 7, two files on the hard drive are examined and we can see the output from xsteg. And also a graphical interface for stegdetect. And in Figure 8 the steganography image and the original carrier for the JPEG image is shown. We have to notice that the steganography file is not only marked which means it contains hidden information, but also the use of steganography scheme by the program.

Stego watch by Wetstone Technologies analyzed some set of files. This analysis helped in finding most likely used algorithm for hiding (which also provided clues to find the most likely software employed) also it provided clues about which steg media is used. Based on carrier file characteristics some varieties of user-selectable statistical tests are used and they are altered using different steganography methods. The examiner selects statistical tests that are more likely used and knowing the steganography software which is available on suspect computer.

In Figure 8 it shows the output from Stego Watch when targeted at the JPEG carrier file. The statistical algorithms that are employed for analysis are shown in Steganography Detection Algorithms section. And also figures show that Stego Watch exactly identifies the s/w (JPEG steganography) which was employed.

The Software was developed in Institute for Security Technology Studies which is capable of detecting the hidden data in image files, but not yet available. Statistical models that are independent of the image format, it simple means the models used are not dependent on the image format are used for hiding data. They tested this program on 4 different steg algorithms and 1, 800 images and finally managed to detect the presence of hidden messages with 75 % accuracy and with no false-positive rate.

Extracting hidden data is some how complicated when compared with finding steganography in a file suspected. In most of the cases steganography software uses passwords for randomization, encryption and secrecy. For JPEG files (Outguess 2003) Stegbreak finds password of hidden data by using dictionary attack against Outguess and JPHide. And this Stegbreak is only applicable for JPEG files. One such program uses dictionary attack on suspect files is Stego Break which is a companion program to WetStone’s Stego. The detection scheme doesn’t help directly for password recovery instead they help in finding appropriate clues where rest of investigation and computer forensic examiner comes into play.

There are many ways to attack Steganography as well as digital watermarks. And that attacks helps in altering and removal. And for digital watermarks, software has been created to attack. The effects of the attack are, it reduces the carrier carrying capacity that is used for steganography and also disables the capability of the steganography medium’s carrier. Apart from these effects it also weakens the carrier strength.

Steganography Disruption:

Steganography Disruption software was designed by Fabien Petitcolas. 2mosiac is a disruption program designed by him and this program employs

" presentation attack" against images on website. It attacks digital watermarking by chopping image into smaller images. So these smaller images are placed next to each other on website, which appears as original large image.

When searching on internet for steganographic tools , I was surprised that no tools available on internet . Also to hide information in MP3 files nothing had done so far , ie sound tracks are compressed by using MPEG Audio Layer III format . Because of the CD quality they offered which is at compression ratio of 11 to 1 (128 kilobits per second) , there is rapidly growing interest in MP3 or WMA files through out the world. So for Information Hiding it is a very good opportunity. As a proof of concept , I provided the implementation of MP3 , although in general WMA got better quality but I don't have access to code.

During the compression process MP3Stego hides the information in MP3 files. As a first step data is compressed and then encrypted , later the encrypted data is hidden in MP3 bit stream. Keeping steganographic applications in mind , MP3stego has been written and for MP3 files it might be used as copyright marking system (it may be weak but much better than MPEG copyright flag which was defined by the standard). The bitstream file can be uncompress and recompress by any opponent. ; which will deletes the hidden information – I think this is the only attack we know up to now – but this happens at severe quality loss.

The inner_loop is the heart of the layer III encoding process where hiding process takes place. The inner loop increases the quantiser size by quantizing the input data , the quantiser size was increased until data coded with the available number of bits. The quantization causes some distortions , so one more loop which check that these distortions must not exceed the threshold which was defined by Psycho acoustic model. The number of main_data bits are contained in the part2_3_length variable which are used for scalefactors and the Huffman code data in MP3 bit stream. And the end loop condition in inner loop is changed to encode the bits as its parity. We choose some part2_3_length values using pseudo random bit generator which is based on SHA-1 and only the chosen values are modified.

As we discussed earlier about the power of parity for information hiding. A practical example for it is MP3Stego. Although there is space for improvement , but some interested people might have a look at it.

compilation

Full C code and binaries

- MP3Stego 1. 1. 18 for Windows (Still there are some issues on certain configurations when we try to decode that data. The source code is available but I don’t have time to look at this moment . We can compile this program easily by using the freely available
- Microsoft Visual Studio Express
- Graphic interface.

Example :

encode -E hidden_text1 . txt -P pass1 svega1. wav svega1_stego. mp3 compresses svega1. wav (mono, 44. 1 kHz, 16bit encoded) and also hides hidden_text1. txt. Using password Pass1 the hidden text is encrypted.

Now the output produced is svega1_stego. mp3. If no information was hidden, you would obtain this.

```
decode -X -P pass1 svega1_stego. mp3
```

Now it uncompresses svega1_stego. mp3 into svega1_stego. mp3. pcm and also it attempts to extract the hidden information. Now we decrypt the hidden message and then the message that is decrypted is uncompressed and saved to svega1_stego. mp3. txt.