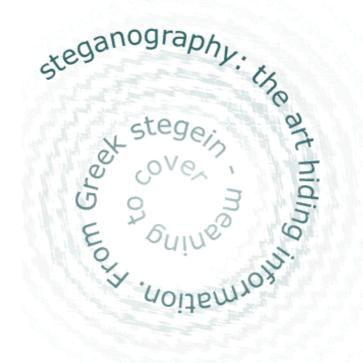
Steganography And Digital Watermarking



Copyright © 2004, Jonathan Cummins, Patrick Diskin, Samuel Lau and Robert Parlett, School of Computer Science, The University of Birmingham.

Permission is granted to copy, distribute and / or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation, except where indicated by * which remain the property of the stated author. A copy of the license is found at http://www.gnu.org/copyleft/fdl.html.

Image* taken from 3D Vizproto '99, Arizona State University.

Introduction

Steganography is derived from the Greek for covered writing and essentially means "to hide in plain sight". As defined by Cachin [1] steganography is the art and science of communicating in such a way that the presence of a message cannot be detected. Simple steganographic techniques have been in use for hundreds of years, but with the increasing use of files in an electronic format new techniques for information hiding have become possible.

This document will examine some early examples of steganography and the general principles behind its usage. We will then look at why it has become such an important issue in recent years. There will then be a discussion of some specific techniques for hiding information in a variety of files and the attacks that may be used to bypass steganography.

Figure 1 shows how information hiding can be broken down into different areas. Steganography can be used to hide a message intended for later retrieval by a specific individual or group. In this case the aim is to prevent the message being detected by any other party.

The other major area of steganography is copyright marking, where the message to be inserted is used to assert copyright over a document. This can be further divided into watermarking and fingerprinting which will be discussed later.

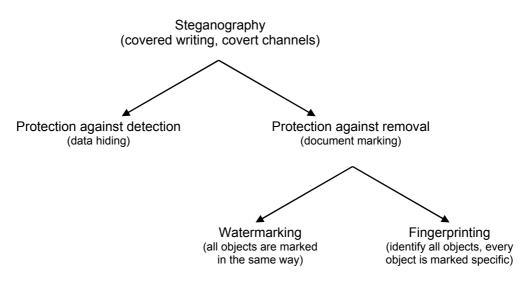


Figure 1*. Types of steganography. Taken from "An Analysis of Steganographic Techniques" by Popa [2].

Steganography and encryption are both used to ensure data confidentiality. However the main difference between them is that with encryption anybody can see that both parties are communicating in secret. Steganography hides the existence of a secret message and in the best case nobody can see that both parties are communicating in secret. This makes steganography suitable for some tasks for which encryption isn't, such as copyright marking. Adding encrypted copyright information to a file could be easy to remove but embedding it within the contents of the file itself can prevent it being easily identified and removed.

Figure 2 shows a comparison of different techniques for communicating in secret. Encryption allows secure communication requiring a key to read the information. An attacker cannot remove the encryption but it is relatively easy to modify the file, making it unreadable for the intended recipient.

Digital signatures allow authorship of a document to be asserted. The signature can be removed easily but any changes made will invalidate the signature, therefore integrity is maintained.

Steganography provides a means of secret communication which cannot be removed without significantly altering the data in which it is embedded. The embedded data will be confidential unless an attacker can find a way to detect it.

| | Confidentiality | Integrity | Unremovability |
|--------------------|-----------------|-----------|----------------|
| Encryption | Yes | No | Yes |
| Digital Signatures | No | Yes | No |
| Steganography | Yes / No | Yes / No | Yes |

Figure 2*. Comparison of secret communication techniques. Taken from "An Analysis of Steganographic Techniques" by Popa [2].

History

One of the earliest uses of steganography was documented in Histories [3]. Herodotus tells how around 440 B.C. Histiaeus shaved the head of his most trusted slave and tattooed it with a message which disappeared after the hair had regrown. The purpose of this message was to instigate a revolt against the Persians. Another slave could be used to send a reply.

During the American Revolution, invisible ink which would glow over a flame was used by both the British and Americans to communicate secretly [4].

Steganography was also used in both World Wars. German spies hid text by using invisible ink to print small dots above or below letters and by changing the heights of letter-strokes in cover texts [5].

In World War I, prisoners of war would hide Morse code messages in letters home by using the dots and dashes on i, j, t and f. Censors intercepting the messages were often alerted by the phrasing and could change them in order to alter the message. A message reading "Father is dead" was modified to read "Father is deceased" and when the reply "Is Father dead or deceased?" came back the censor was alerted to the hidden message.

During World War II, the Germans would hide data as microdots. This involved photographing the message to be hidden and reducing the size so that that it could be used as a period within another document. FBI director J. Edgar Hoover described the use of microdots as "the enemy's masterpiece of espionage".

A message sent by a German spy during World War II read:

"Apparently neutral's protest is thoroughly discounted and ignored. Isman hard hit. Blockade issue affects for pretext embargo on by-products, ejecting suets and vegetable oils."

By taking the second letter of every word the hidden message "Pershing sails for NY June 1" can be retrieved.

More recent cases of steganography include using special inks to write hidden messages on bank notes and also the entertainment industry using digital watermarking and fingerprinting of audio and video for copyright protection.

Digital Rights And Copyright Marking

One of the driving forces behind the increased use of copyright marking is the growth of the Internet which has allowed images, audio, video, etc to become available in digital form. Though this provides an additional way to distribute material to consumers it has also made it far easier for copies of copyrighted material to be made and distributed. In the past, pirating music, for example, used to require some form of physical exchange. Using the Internet a copy stored on a computer can be shared easily with anybody regardless of distance often via a peer-to-peer network which doesn't require the material to be stored on a server and therefore makes it harder for the copyright owner to locate and prosecute offending parties.

It is estimated that Internet file sharing and pirating music in MP3 format costs the global music industry in excess of £2.8 billion a year [6]. There has been a significant drop in CD sales since the Internet took off and the music industry is investing heavily in the research of copyright watermarking which they hope will enable them to bring copyright violators to court.

Copyright marking is seen as a partial solution to these problems. The mark can be embedded in any legal versions and will therefore be present in any copies made. This helps the copyright owner to identify who has an illegal copy.

Requirements Of Hiding Information Digitally

There are many different protocols and embedding techniques that enable us to hide data in a given object. However, all of the protocols and techniques must satisfy a number of requirements so that steganography can be applied correctly. The following is a list of main requirements that steganography techniques must satisfy:

- The integrity of the hidden information after it has been embedded inside the stego object must be correct. The secret message must not change in any way, such as additional information being added, loss of information or changes to the secret information after it has been hidden. If secret information is changed during steganography, it would defeat the whole point of the process.
- The stego object must remain unchanged or almost unchanged to the naked eye. If the stego object changes significantly and can be noticed, a third party may see that information is being hidden and therefore could attempt to extract or to destroy it.
- In watermarking, changes in the stego object must have no effect on the watermark. Imagine if you had an illegal copy of an image that you would like to manipulate in various ways. These manipulations can be simple processes such as resizing, trimming or rotating the image. The watermark inside the image must survive these manipulations, otherwise the attackers can very easily remove the watermark and the point of steganography will be broken.
- Finally, we always assume that the attacker knows that there is hidden information inside the stego object.

Embedding And Detecting A Mark

Figure 3 shows a simple representation of the generic embedding and decoding process in steganography. In this example, a secret image is being embedded inside a cover image to produce the stego image.

The first step in embedding and hiding information is to pass both the secret message and the cover message into the encoder. Inside the encoder, one or several protocols will be implemented to embed the secret information into the cover message. The type of protocol will depend on what information you are trying to embed and what you are embedding it in. For example, you will use an image protocol to embed information inside images.

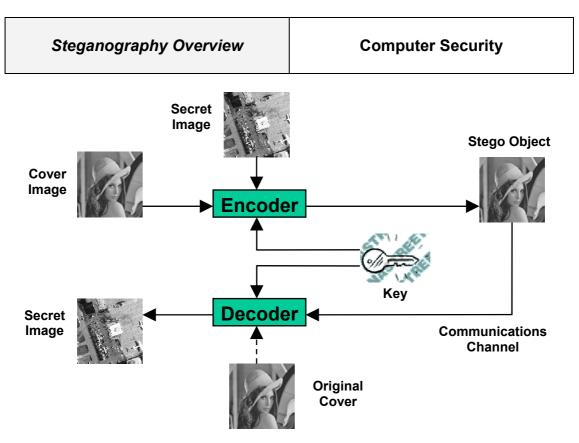


Figure 3. Generic process of encoding and decoding.

A key is often needed in the embedding process. This can be in the form of a public or private key so you can encode the secret message with your private key and the recipient can decode it using your public key. In embedding the information this way, you can reduce the chance of a third party attacker getting hold of the stego object and decoding it to find out the secret information.

In general the embedding process inserts a mark, M, in an object, I. A key, K, usually produced by a random number generator is used in the embedding process and the resulting marked object, \tilde{I} , is generated by the mapping: $I \times K \times M \rightarrow \tilde{I}$.

Having passed through the encoder, a stego object will be produced. A stego object is the original cover object with the secret information embedded inside. This object should look almost identical to the cover object as otherwise a third party attacker can see embedded information.

Having produced the stego object, it will then be sent off via some communications channel, such as email, to the intended recipient for decoding. The recipient must decode the stego object in order for them to view the secret information. The decoding process is simply the reverse of the encoding process. It is the extraction of secret data from a stego object.

In the decoding process, the stego object is fed in to the system. The public or private key that can decode the original key that is used inside the encoding process is also needed so that the secret information can be decoded. Depending on the encoding technique, sometimes the original cover object is also needed in the decoding process. Otherwise, there may be no way of extracting the secret information from the stego object.

After the decoding process is completed, the secret information embedded in the stego object can then be extracted and viewed. The generic decoding process again requires a key, K, this time along with a potentially marked object, Ĩ'. Also required is either the mark, M, which is being checked for or the original object, I, and the result will be either the retrieved mark from the object or indication of the likelihood of M being present in Ĩ'. Different types of robust marking systems use different inputs and outputs.

• Private Marking Systems

Private marking systems can be divided further into different types but all require the original image. Type I systems use I to help locate the mark in \tilde{I} and output the mark.

Type II systems also require M and simply give a yes or no answer to the question "does \tilde{I} " contain the mark M?" This can be seen as a mapping: $\tilde{I} \times I \times K \times M \rightarrow \{0, 1\}$.

Semi-private marking systems work like Type II except they don't require the original image and simply answer the same question through the mapping: $\tilde{l}' \times K \times M \rightarrow \{0, 1\}$.

Private marking systems reveal little information and require the secret key in order to detect the mark. Many current systems fall into this category and they are often used to prove ownership of material in court.

• Public Marking Systems (Blind Marking)

Public marking systems do not require either I or M but extract n bits from \tilde{I} which represents the mark: $\tilde{I} \times K \to M$.

Public marking systems have a wider range of applications and the algorithms can often be used in private systems.

• Asymmetric Marking Systems (Public Key Marking)

Asymmetric marking systems allow any user to read the mark but prevent them from removing it.

Types Of Steganography

Steganography can be split into two types, these are Fragile and Robust. The following section describes the definition of these two different types of steganography.

• Fragile

Fragile steganography involves embedding information into a file which is destroyed if the file is modified. This method is unsuitable for recording the copyright holder of the file since it can be so easily removed, but is useful in situations where it is important to prove that the file has not been tampered with, such as using a file as evidence in a court of law, since any tampering would have removed the watermark. Fragile steganography techniques tend to be easier to implement than robust methods.

Robust

Robust marking aims to embed information into a file which cannot easily be destroyed. Although no mark is truly indestructible, a system can be considered robust if the amount of changes required to remove the mark would render the file useless. Therefore the mark should be hidden in a part of the file where its removal would be easily perceived.

There are two main types of robust marking. Fingerprinting involves hiding a unique identifier for the customer who originally acquired the file and therefore is allowed to use it. Should the file be found in the possession of somebody else, the copyright owner can use the fingerprint to identify which customer violated the license agreement by distributing a copy of the file.

Unlike fingerprints, watermarks identify the copyright owner of the file, not the customer. Whereas fingerprints are used to identify people who violate the license agreement watermarks help with prosecuting those who have an illegal copy. Ideally fingerprinting should be used but for mass production of CDs, DVDs, etc it is not feasible to give each disk a separate fingerprint.

Watermarks are typically hidden to prevent their detection and removal, they are said to be imperceptible watermarks. However this need not always be the case. Visible watermarks can be used and often take the form of a visual pattern overlaid on an image. The use of visible watermarks is similar to the use of watermarks in non-digital formats (such as the watermark on British money).

Overview

By taking advantage of human perception it is possible to embed data within a file. For example, with audio files frequency masking occurs when two tones with similar frequencies are played at the same time. The listener only hears the louder tone while the quieter one is masked. Similarly, temporal masking occurs when a low-level signal occurs immediately before or after a stronger one as it takes us time to adjust to the hearing the new frequency. This provides a clear point in the file in which to embed the mark.

However many of the formats used for digital media take advantage of compression standards such as MPEG to reduce file sizes by removing the parts which are not perceived by the users. Therefore the mark should be embedded in the perceptually most significant parts of the file to ensure it survives the compression process.

Clearly embedding the mark in the significant parts of the file will result in a loss of quality since some of the information will be lost. A simple technique involves embedding the mark in the least significant bits which will minimise the distortion. However it also makes it relatively easy to locate and remove the mark. An improvement is to embed the mark only in the least significant bits of randomly chosen data within the file.

In this section a number of different information hiding techniques will be discussed and examined. The media involved vary from images to plain text. While some techniques may be used to hide a certain type of information, in most cases different information can be hidden depending on space restraints.

Binary File Techniques

If we are trying to hide some secret information inside a binary file, whether the secret information is a copyright watermark or just simple secret text, we are faced with the problem that any changes to that binary file will cause the execution of it to alter. Just adding one single instruction will cause the executing to be different and therefore the program may not function properly and may crash the system.

You may wonder why people would want to embed information inside binary files, since there are so many other types of data format we can embed information in. The main reason for this is people want to protect their copyright inside a binary program. Of course there are other means of protecting copyright in software, such as serial keys, but if you did a search on the Internet, key generators for common programs are widely available and therefore using serial keys alone may not be enough to protect the binary file's copyright.

One method for embedding a watermark in a binary file works as follows. First, let's look at the following lines of code that have been extracted from a binary file:

a = 2; b = 3; c = b + 3; d = b + c;

The above instruction is simply equivalent to:

| b = 3; | b = 3; | b = 3; |
|------------|------------|------------|
| a = 2; | c = b + 3; | c = b + 3; |
| c = b + 3; | a = 2; | d = b + c; |
| d = b + c; | d = b + c; | a = 2; |

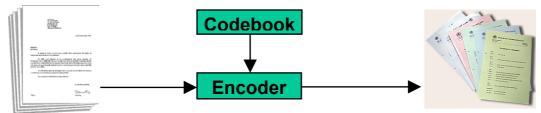
The initialisation of b, c, and d must be done in the same order, but a can be initialised at any time.

To embed a watermark $W = \{w_1, w_2, w_3, w_4, ..., w_n\}$ where $w_i \in \{0, 1\}$. We first divide the source code into n blocks. Each of these blocks is then represented by w_i and this holds the value either 0 or 1. If w_i is 0, then the block of code it represents will be left unchanged. However, if w_i is 1, then you will look for two statements inside the block and switch them over.

Using this method, the watermark can be embedded by making changes to the binary code that does not affect the execution of the file. To decode and extract the watermark, you will need to have the original binary file. By comparing the marked and original files, you can then spot the statement switches and therefore extract the embedded watermark. This method is very simple but is not resistant to attacks. If the attacker has many different versions of the marked files then he may detect the watermark and hence be able to remove it.

Text Techniques

While it is very easy to tell when you have committed a copyright infringement by photocopying a book, since the quality is widely different, it is more difficult when it comes to electronic versions of text. Copies are identical and it is impossible to tell if it is an original or a copied version. To embed information inside a document we can simply alter some of its characteristics. These can be either the text formatting or characteristics of the characters. You may think that if we alter these characteristics it will become visible and obvious to third parties or attackers. The key to this problem is that we alter the document in a way that it is simply not visible to the human eye yet it is possible to decode it by computer.



Original Document

Marked Documents

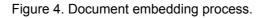


Figure 4 shows the general principle in embedding hidden information inside a document. Again, there is an encoder and to decode it, there will be a decoder. The codebook is a set of rules that tells the encoder which parts of the document it needs to change. It is also worth pointing out that the marked documents can be either identical or different. By different, we mean that the same watermark is marked on the document but different characteristics of each of the documents are changed.

• Line Shift Coding Protocol

In line shift coding, we simply shift various lines inside the document up or down by a small fraction (such as 1/300th of an inch) according to the codebook. The shifted lines are undetectable by humans because it is only a small fraction but is detectable when the computer measures the distances between each of the lines. Differential encoding techniques are normally used in this protocol, meaning if you shift a line the adjacent lines are not moved. These lines will become a control so that the computer can measure the distances between them.

By finding out whether a line has been shifted up or down we can represent a single bit, 0 or 1. And if we put the whole document together, we can embed a number of bits and therefore have the ability to hide large information.

• Word Shift Coding Protocol

The word shift coding protocol is based on the same principle as the line shift coding protocol. The main difference is instead of shifting lines up or down, we shift words left or right. This is also known as the justification of the document. The codebook will simply tell the encoder which of the words is to be shifted and whether it is a left or a right shift. Again, the decoding technique is measuring the spaces between each word and a left shift could represent a 0 bit and a right bit representing a 1 bit.

The quick brown fox jumps over the lazy dog. The quick brown fox jumps over the lazy dog.

In this example the first line uses normal spacing while the second has had each word shifted left or right by 0.5 points in order to encode the sequence 01000001, that is 65, the ASCII character code for A. Without having the original for comparison it is likely that this may not be noticed and the shifting could be even smaller to make it less noticeable.

• Feature Coding Protocol

In feature coding, there is a slight difference with the above protocols, and this is that the document is passed through a parser where it examines the document and it automatically builds a codebook specific to that document. It will pick out all the features that it thinks it can use to hide information and each of these will be marked into the document. This can use a number of different characteristics such as the height of certain characters, the dots above i and j and the horizontal line length of letters such as f and t. Line shifting and word shifting techniques can also be used to increase the amount of data that can be hidden.

• White Space Manipulation

One way of hiding data in text is to use white space. If done correctly, white space can be manipulated so that bits can be stored. This is done by adding a certain amount of white space to the end of lines. The amount of white space corresponds to a certain bit value. Due to the fact that in practically all text editors, extra white space at the end of lines is skipped over, it won't be noticed by the casual viewer. In a large piece of text, this can result in enough room to hide a few lines of text or some secret codes. A program which uses this technique is SNOW [7], which is freely available.

Text Content

Another way of hiding information is to conceal it in what seems to be inconspicuous text. The grammar within the text can be used to store information. It is possible to change sentences to store information and keep the original meaning. TextHide [8] is a program, which incorporates this technique to hide secret messages. A simple example is:

"The auto drives fast on a slippery road over the hill."

Changed to:

"Over the slope the car travels quickly on an ice-covered street."

Another way of using text itself is to use random words as a means of encoding information. Different words can be given different values. Of course this would be easy to spot but there are clever implementations, such as SpamMimic [9] which creates a spam email that contains a secret message. As spam usually has poor grammar, it is far easier for it to escape notice. The following extract from a spam email encodes the phrase "I'm having a great time learning about computer security."

Steganography Techniques

Dear Friend , Especially for you - this red-hot intelligence . We will comply with all removal requests . This mail is being sent in compliance with Senate bill 2116 , Title 9 ; Section 303 ! THIS IS NOT A GET RICH SCHEME . Why work for somebody else when you can become rich inside 57 weeks . Have you ever noticed most everyone has a cellphone & people love convenience . Well, now is your chance to capitalize on this . WE will help YOU SELL MORE and sell more ! You are guaranteed to succeed because we take all the risk ! But don't believe us . Ms Simpson of Washington tried us and says "My only problem now is where to park all my cars" . This offer is 100% legal . You will blame yourself forever if you don't order now ! Sign up a friend and you'll get a discount of 50% . Thank-you for your serious consideration of our offer . Dear Decision maker ; Thank-you for your interest in our briefing . If you are not interested in our publications and wish to be removed from our lists, simply do NOT respond and ignore this mail ! This mail is being sent in compliance with Senate bill 1623 ; Title 6 ; Section 304 ! THIS IS NOT A GET RICH SCHEME ! Why work for somebody else when you can ...

A very basic form of steganography makes use of a cipher. A cipher is basically a key which can be used to decode some data to retrieve a secret hidden message. Sir Francis Bacon created one in the 16th Century [10] using messages with two different type faces, one bolder than the other. By looking at the positions of the bold characters in relation to the rest of the text, a secret message could be decoded. There are many other different ciphers which could be used to the same effect.

XML

XML is becoming a widely used standard for data exchange. The format also provides plenty of opportunities for data hiding. This is important for verifying documents to see if they have been altered and also for copyright reasons. You can embed a code for example, which can be traced back to the source. A method for hiding information in XML comes courtesy of the University of Tokyo [11].

Many different files can exist when XML is used. There is the XML file itself but there can be transformation files (.xsl), validation files (.dtd) and style files (.css). All of these files can be used to hide data but the main XML file is usually the best due to its larger size. This technique concentrates on just the XML file, more elaborate techniques could use a combination of all four files to increase robustness.

One way of hiding data in XML is to use the different tags as allowed by the W3C. For example both of these image tags are valid and could be used to indicate different bit settings

```
Stego key:
<img></img> -> 0
<img/> -> 1
```

In this way a piece of XML like the following could be used to encode a simple bit string.

Stego data:

That XML stores the bit string 01110. Another way of hiding data is by using the space inside a tag. Once again the following XML code is used as the key while the code after is an example of how it could be used to store a string:

```
Stego key:
<tag>, </tag>, or <tag/> -> 0
<tag >, </tag >, or <tag /> -> 1
```

Stego data:

<user ><name>Alice</name ><id >01</id></user>
<user><name >Bob</name><id>>02</id ></user >

The XML data in this case stores the bit strings 101100 and 010011.

Other ways of storing data include using the order in which attributes or elements appear. For example, assigning the combination of element A followed by element B the bit value of 1 while if A is followed by some element C, it would be assigned the value of 0.

Hiding data using the scheme outlined above would be pretty easy. In the case of using white space, a simple text manipulation program could be used to add the spaces and then a reader could be created to parse the XML and retrieve the hidden data. The same is true for the usage of different tags. The structure of elements would be a little more difficult as changing elements could have an adverse impact on the way the XML is displayed but if cleverly designed, this could be overcome. In this example the containment of elements is used:

<favorite><fruit>SOMETHING</fruit></favorite> -> 0 <fruit><favorite>SOMETHING</favorite></fruit> -> 1

In this example the order of the elements is used:

```
<user><name>NAME</name><id>ID</id></user> -> 0
<user><id>ID</id><name>NAME</name></user> -> 1
```

Image Techniques

• Simple Watermarking

A very simple yet widely used technique for watermarking images is to add a pattern on top of an existing image. Usually this pattern is an image itself - a logo or something similar, which distorts the underlying image.







Figure 5. Visible watermarking.

In the example above, the pattern is the red middle image while the portrait picture of Dr. Axford is the image being watermarked. In a standard image editor it is possible to merge both images and get a watermarked image. As long as you know the watermark, it is possible to reverse any adverse effects so that the original doesn't need to be kept. This method is only really applicable to watermarking, as the pattern is visible and even without the original watermark, it is possible to remove the pattern from the watermarked image with some effort and skill.

• LSB – Least Significant Bit Hiding (Image Hiding)

This method is probably the easiest way of hiding information in an image and yet it is surprisingly effective. It works by using the least significant bits of each pixel in one image to hide the most significant bits of another. So in a JPEG image for example, the following steps would need to be taken

- 1. First load up both the host image and the image you need to hide.
- 2. Next chose the number of bits you wish to hide the secret image in. The more bits used in the host image, the more it deteriorates. Increasing the number of bits used though obviously has a beneficial reaction on the secret image increasing its clarity.
- Now you have to create a new image by combining the pixels from both images. If you decide for example, to use 4 bits to hide the secret image, there will be four bits left for the host image. (PGM - one byte per pixel, JPEG - one byte each for red, green, blue and one byte for alpha channel in some image types)

Host Pixel: <u>1011</u>0001 Secret Pixel: <u>0011</u>1111 New Image Pixel: **10110011**

4. To get the original image back you just need to know how many bits were used to store the secret image. You then scan through the host image, pick out the least significant bits according the number used and then use them to create a new image with one change - the bits extracted now become the most significant bits.

Host Pixel: 1011<u>0011</u> Bits used: 4 New Image: **00110000**

Original Images





Bits Used: 4





Figure 6. Least significant bit hiding.

Bits Used: 7



To show how this technique affects images, Figure 6 shows examples using different bit values. Dr. Ryan's image on the left is the host image while Mr. Sexton's on the right is the secret one we wish to hide.

This method works well when both the host and secret images are given equal priority. When one has significantly more room than another, quality is sacrificed. Also while in this example an image has been hidden, the least significant bits could be used to store text or even a small amount of sound. All you need to do is change how the least significant bits are filled in the host image. However this technique makes it very easy to find and remove the hidden data [12].

• Direct Cosine Transformation

Another way of hiding data is by way of a direct cosine transformation (DCT). The DCT algorithm is one of the main components of the JPEG compression technique [13]. This works as follows [14], [15]:

- 1. First the image is split up into 8 x 8 squares.
- 2. Next each of these squares is transformed via a DCT, which outputs a multi dimensional array of 63 coefficients.
- 3. A quantizer rounds each of these coefficients, which essentially is the compression stage as this is where data is lost.
- 4. Small unimportant coefficients are rounded to 0 while larger ones lose some of their precision.
- 5. At this stage you should have an array of streamlined coefficients, which are further compressed via a Huffman encoding scheme or similar.
- 6. Decompression is done via an inverse DCT.

Hiding via a DCT is useful as someone who just looks at the pixel values of the image would be unaware that anything is amiss. Also the hidden data can be distributed more evenly over the whole image in such a way as to make it more robust.

One technique hides data in the quantizer stage [14]. If you wish to encode the bit value 0 in a specific 8 x 8 square of pixels, you can do this by making sure all the coefficients are even, for example by tweaking them. Bit value 1 can be stored by tweaking the coefficients so that they are odd. In this way a large image can store some data that is quite difficult to detect in comparison to the LSB method.

This is a very simple method and while it works well in keeping down distortions, it is vulnerable to noise.



Original Image



Watermarked Image



JPEG compressed

Figure 7. Direct Cosine Transformation.

Other techniques, which use DCT transformations, sometimes use different algorithms for storing the bit. One uses pseudo noise to add a watermark to the DCT coefficients while another uses an algorithm to encode and extract a bit from them. These other techniques are generally more complex and are more robust than the technique described.

• Wavelet Transformation

While DCT transformations help hide watermark information or general data, they don't do a great job at higher compression levels. The blocky look of highly compressed JPEG files is due to the 8 x 8 blocks used in the transformation process. Wavelet transformations on the other hand are far better at high compression levels and thus increase the level of robustness of the information that is hidden, something which is essential in an area like watermarking [16].

This technique works by taking many wavelets to encode a whole image. They allow images to be compressed so highly by storing the high frequency "detail" in the image separately from the low frequency parts. The low frequency areas can then be compressed which is acceptable as they are most viable for compression. Quantization can then take place to compress things further and the whole process can start again if needed.

A simple technique using wavelets to hide information is exactly like one of the techniques discussed in the previous section [17]. Instead of altering the DCT coefficients with pseudo noise, instead the coefficients of the wavelets are altered with the noise within tolerable levels.

Embedding information into wavelets is an ongoing research topic, which still holds a lot of promise.

Sound Techniques

• Spread Spectrum

Spread spectrum systems encode data as a binary sequence which sounds like noise but which can be recognised by a receiver with the correct key. The technique has been used by the military since the 1940s because the signals are hard to jam or intercept as they are lost in the background noise. Spread spectrum techniques can be used for watermarking by matching the narrow bandwidth of the embedded data to the large bandwidth of the medium.

• MIDI

MIDI files are good places to hide information due to the revival this format has had with the surge of mobile phones, which play MIDI ring tones. There are also techniques which can embed data into MIDI files easily [18].

MIDI files are made up of a number of different messages. Some of these messages control the notes you hear while others are silent and make up the file header or change the notes being played. The message we are interested in is one called Program Change (PC). A PC basically changes the type of instrument being played on a certain channel. If there are multiple PC messages in succession the instrument played will be the one selected at the very end of the message chain and due to the fact these messages occur so frequently, there are no noticeable side effects to the sound.

Each PC message can contain a number from 0 to 127, which corresponds to the number of different instruments that can be played [19]. So all you need to do is string together the necessary number of PC messages to contain the hidden data.

Obviously this method doesn't allow for huge amounts of data to be stored nor is it a very good way of hiding data as it can be easily seen.

• MP3

The MP3 format is probably the most widespread compression format currently used for music files. Due to this, it also happens to be very good for hiding information in. The more inconspicuous the format, the more easily the hidden data may be overlooked.

There are very few working examples of hiding information in MP3 files but one freely available program is MP3Stego [20]. The technique used here is similar to the frequency transformations discussed earlier. Basically the data to be hidden is stored as the MP3 file is created, that is during the compression stage [21].

As the sound file is being compressed during the Layer 3 encoding process, data is selectively lost depending on the bit rate the user has specified. The hidden data is encoded in the parity bit of this information. As MP3 files are split up into a number of frames [22] each with their own parity bit, a reasonable amount of information can be stored. To retrieve the data all you need to do is uncompress the MP3 file and read the parity bits as this process is done. This is an effective technique which leaves little trace of any distortions in the music file.

Other Techniques

• Video

For video, a combination of sound and image techniques can be used. This is due to the fact that video generally has separate inner files for the video (consisting of many images) and the sound. So techniques can be applied in both areas to hide data. Due to the size of video files, the scope for adding lots of data is much greater and therefore the chances of hidden data being detected is quite low.

• DNA

A relatively new area for information hiding is within DNA. In one technique explained by Peterson [23] a message "JUNE6_INVASION:NORMANDY" was hidden inside some DNA. This was done in a scheme quite similar to some of the text techniques discussed earlier.

A single strand of DNA consists of a chain of simple molecules called bases, which protrude from a sugar-phosphate backbone. The four varieties of bases are known as adenine (A), thymine (T), guanine (G), and cytosine (C). A table was drawn up with different three base combinations equalling different words in the alphabet along with a few other things.

To create the secret message, DNA was synthesised following this table with the bases in the right order. Then it was sandwiched between another two strands of DNA which acted as markers to point the sender and recipient of the message to the message. The final step taken was to add in some random DNA strands in order to further prevent the detection of the secret message.

As DNA is incredibly small, it can be hidden in a dot in a book or magazine much like the old microdot technique used in World War II. It is also robust enough to be posted through the mail and still be decoded. This could prove to be a very effective technique in the future.

Limitations

There are limitations on the use of steganography. As with encryption, if Alice wants to communicate secretly with Bob they must first agree on the method being used. Demeratus, a Greek at the Persian court, sent a warning to Sparta about an imminent invasion by Xerxes by removing the wax from a writing tablet, writing the message on the wood and then covering it in wax again [3]. The tablet appeared to be blank and fooled the customs men but almost fooled the recipient too since he was unaware that the message was being hidden.

With encryption, Bob can be reasonably sure that he has received a secret message when a seemingly meaningless file arrives. It has either been corrupted or is encrypted. It is not so clear with hidden data, Bob simply receives an image, for example, and needs to know that there is a hidden message and how to locate it [24].

Another limitation is due to the size of the medium being used to hide the data. In order for steganography to be useful the message should be hidden without any major changes to the object it is being embedded in. This leaves limited room to embed a message without noticeably changing the original object.

This is most obvious in compressed files where many of the obvious candidates for embedding data are lost. What is left is likely to be the most perceptually significant portions of the file and although hiding data is still possible it may be difficult to avoid changing the file.

Detection

Although many of the uses of steganography are perfectly legal, it can be abused by certain groups. The potential exists for terrorist groups to communicate using these techniques to hide their messages and rumours persist that Al-Qaeda have used it to communicate. Also of concern is that these techniques may be used by paedophiles to hide pornographic images within seemingly innocuous material.

As a result the need for detection of steganographic data has become an important issue for law enforcement agencies. Attempting to detect the use of steganography is called steganalysis and can be either passive, where the presence of the hidden data is detected, or active, where an attempt is made to retrieve the hidden data.

This detection is similar to that described earlier for checking for the presence of a watermark. However, whereas before detection will be used when a mark is expected and may involve using the original file, in this case the original file is unavailable and there is no expected mark. Instead the file must be checked for the presence of data hidden in a variety of formats. Due to the vast number of hiding techniques, detecting them all is infeasible and indeed detecting the presence of any could be time consuming.

Detecting hidden data remains an active area of research and is outlined in various papers including [25], [26].

Attacks

Information hiding techniques still suffer from several limitations leaving them open to attack and robustness criteria vary between different techniques. Attacks can be broadly categorized although some attacks will fit into multiple categories [27].

Basic Attacks

Basic attacks take advantage of limitations in the design of the embedding techniques. Simple spread spectrum techniques, for example, are able to survive amplitude distortion and noise addition but are vulnerable to timing errors. Synchronisation of the chip signal is required in

order for the technique to work so adjusting the synchronisation can cause the embedded data to be lost.

It is possible to alter the length of a piece of audio without changing the pitch and this can also be an effective attack on audio files.

Robustness Attacks

Robustness attacks attempt to diminish or remove the presence of a watermark [28]. Although most techniques can survive a variety of transformations, compression, noise addition, etc they do not cope so easily with combinations of them or with random geometric distortions. If a series of minor distortions are applied the watermark can be lost while the image remains largely unchanged. What changes have been made will likely be acceptable to pirates who do not usually require high quality copies. Since robustness attacks involve the use of common manipulations, they need not always be malicious but could just be the result of normal usage by licensed users.

Protecting against these attacks can be done by anticipating which transformations pirates are likely to use. Embedding multiple copies of the mark using inverse transformations can increase the resistance to these attacks.

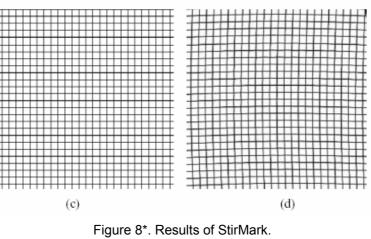
(a)

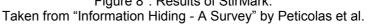
However, trying to guess potential attacks is not ideal. The use of benchmarking for evaluating techniques could help to determine how robust the technique is. StirMark is a tool which applies minor geometric distortions, followed by a random low frequency deviation based around the centre of the image and finally a transfer function to introduce error into all sample values similar to the effects of a scanner. StirMark can serve as a benchmark for image watermarking.

Figure 8 shows the results of StirMark applied to image (a) in image (c). The distortions here are almost unnoticeable and are easier to see when the same distortions are applied to grid (c) to give (d).



(b)





The echo hiding technique encodes zeros and ones by adding echo signals distinguished by different values for their delay and amplitude to an audio signal. Decoding can be done by detecting the initial delay using the auto-correlation of the cepstrum of the encoded signal but this technique can also be used as an attack.

If the echo can be detected then it can be removed by inverting the formula used to add it. The difficult part is detecting the echo without any knowledge of the original or the echo parameters. This problem is known as 'blind echo cancellation'. Finding the echo can be done using a technique called cepstrum analysis.

Other attacks will attempt to identify the watermark and then remove it. This technique is particularly applicable if the marking process leaves clues that help the attacker gain information about the mark. For example an image with a low number of colours, such as a cartoon image, will have sharp peaks in the colour histogram. Some marking algorithms split these and the twin peaks attack takes advantage of this to identify the marks which can then be removed [29].

• Presentation Attacks

Presentation attacks modify the content of the file in order to prevent the detection of the watermark. The mosaic attack takes advantage of size requirements for embedding a watermark. In order for the marked file to be the same size as the original the file must have some minimum size to accommodate the mark. By splitting the marked file into small sections the mark detection can be confused. Many web browsers will draw images together with no visible split enabling the full image to be effectively restored while hiding the mark. If the minimum size for embedding the mark is small enough the mosaic attack is not practical. This attack can defeat web crawlers which download pictures from the Internet and check them for the presence of a client's watermark.





Figure 9. The mosaic attack.

In this example an image had a simple watermark embedded in it using Digimarc included in Jasc Paint Shop Pro. The image was then separated into 16 tiles, each of which was then checked for the presence of the watermark. Tiles are shown separated here for clarity and those surrounded by the red border no longer contain the watermark. However this does show how small the tiles need to be in order to lose all watermark information as 6 tiles still contain the watermark at this size. If the tiles are made small enough, the watermark could be lost.

• Interpretation Attacks

Interpretation attacks involve finding a situation in which the assertion of ownership is prevented [30]. Robustness is usually used to refer to the ability of the mark to survive transformations and not resistance to an algorithmic attack. Therefore the definition of robustness may not be sufficient.

One interpretation attack takes advantage of mark detection being unable to tell which mark came first if multiple marks are found. If the owner publishes a document, d + w (where d is

the original and w is the watermark) a pirate can add a second watermark w' and claim that the document is his and that the original was d + w - w'. Though it is clear that at least one party has a counterfeit copy, it is not clear which one. This would seem to suggest the need to use other techniques to identify the original owner of a file.

• Implementation Attacks

As with other areas in computer security the implementation of a marking system can provide more opportunities for attack than the marking technique itself. If the mark detection software is vulnerable it may be possible for attackers to deceive it.

Digimarc, one of the most widely used picture marking schemes was attacked using a weakness in the implementation. Users register an ID and password with the marking service. A debugger was used to break into the software which checks these passwords and disable the checking. The attacker can change the ID and this will change the mark of already marked images. The debugger also allowed bypassing of checks to see if a mark already existed and therefore allowed marks to be overwritten.

There is a general attack on mark readers which explores an image on the boundary between no mark having been found and one being detected. An acceptable copy of the image can be iteratively generated which does not include the mark.

Clearly the software used to implement steganographic techniques needs to be secure and ideas from other areas of computer security can be used to ensure this.

Conclusion

As steganography becomes more widely used in computing there are issues that need to be resolved. There are a wide variety of different techniques with their own advantages and disadvantages.

Many currently used techniques are not robust enough to prevent detection and removal of embedded data. The use of benchmarking to evaluate techniques should become more common and a more standard definition of robustness is required to help overcome this.

Peticolas et al. propose a definition of robust similar to that being used by the music industry [5]. For a system to be considered robust it should have the following properties:

- The quality of the media should not noticeably degrade upon addition of a mark.
- Marks should be undetectable without secret knowledge, typically the key.
- If multiple marks are present they should not interfere with each other.
- The marks should survive attacks that don't degrade the perceived quality of the work.

As attacks are found that work against existing techniques, it is likely that new techniques will be developed that overcome these deficiencies. The continuing use of digital media will drive development of new techniques and standards for watermarking are likely to be developed.

Meanwhile techniques used by law enforcement authorities to detect embedded material will improve as they continue to try and prevent the misuse of steganography.

- C. Cachin, "An Information-Theoretic Model for Steganography", Proceedings of 2nd Workshop on Information Hiding, MIT Laboratory for Computer Science, May 1998
- [2] R. Popa, An Analysis of Steganographic Techniques, The "Politehnica" University of Timisoara, Faculty of Automatics and Computers, Department of Computer Science and Software Engineering, http://ad.informatik.uni-
- freiburg.de/mitarbeiter/will/dlib_bookmarks/digital-watermarking/popa/popa.pdf, 1998
 [3] Herodotus, *The Hisories*, chap. 5 The fifth book entitled Terpsichore, 7 The seventh book entitled Polymnia, J. M. Dent & Sons, Ltd, 1992
- [4] Second Lieutenant J. Caldwell, *Steganography*, United States Air Force, http://www.stsc.hill.af.mil/crosstalk/2003/06/caldwell.pdf, June 2003
- [5] F. A. P. Petitcolas, R. J. Anderson and M. G. Kuhn, "Information Hiding A Survey", *Proceedings of the IEEE*, vol. 87, no. 7, pp. 1062-1078, July 1999
- [6] BBC News, *Piracy blamed for CD sales slump*, BBC, http://news.bbc.co.uk/1/hi/entertainment/new_media/1841768.stm, February 2002
- [7] M. Kwan, *The Snow Home Page*, http://www.darkside.com.au/snow/index.html, March 2001
- [8] Compris Intelligence, *TextHide*, Compris Intelligence , http://www.compris.com/TextHide/en/
- [9] P. Wayner, SpamMimic, http://www.spammimic.com, 2003
- [10] R. Hipschman, *The Secret Language*, Exploratorium, http://www.exploratorium.edu/ronh/secret/secret.html, 1995
- [11] S. Inoue, K. Makino, I. Murase, O. Takizawa, T. Matsumoto and H. Nakagawa, A Proposal on Information Hiding Methods using XML, http://takizawa.gr.jp/lab/nlp_xml.pdf
- [12] M. D. Swanson, B. Zhu and A. H. Tewfik, "Robust Data Hiding for Images", IEEE Digital Signal Processing Workshop, pp. 37-40, Department of Electrical Engineering, University of Minnesota, http://www.assuredigit.com/tech_doc/more/Swanson_dsp96_robust_datahiding.pdf, September 1996
- [13] L. Leurs, *JPEG Compression*, http://www.prepressure.com/techno/compressionjpeg.htm, 2001
- [14] A. K. Chao and C. Chao, *Robust Digital Watermarking & Data Hiding*, Image Systems Engineering Program, Stanford University,
- http://ise.stanford.edu/class/ee368a_proj00/project7/index.html, May 2000 [15] J. Gailly, *comp.compression Frequently Asked Questions (part 2/3)*, Internet FAQ
- Archives, http://www.faqs.org/faqs/compression-faq/part2/, September 1999
- [16] National Academy of Sciences, How do Wavelets work?, National Academy of Sciences, http://www.beyonddiscovery.org/content/view.page.asp?I=1956, 2003
- [17] C. Shoemaker, *Hidden Bits: A Survey of Techniques for Digital Watermarking*, http://www.vu.union.edu/~shoemakc/watermarking/watermarking.html#watermarkobject, Virtual Union, 2002
- [18] J. Corinna, *Steganography*, Binary Universe, http://www.binaryuniverse.de/articles/5/english/steganodotnet5.html, 2003
- [19] J. Glatt, MIDI is the language of gods, http://www.borg.com/~jglatt/
- [20] F. A. P. Petitcolas, mp3stego,
 - http://www.petitcolas.net/fabien/steganography/mp3stego/, September 2003
- [21] Fraunhofer-Gesellschaft, Audio & Multimedia MPEG Audio Layer-3, Fraunhofer-Gesellschaft, http://www.iis.fraunhofer.de/amm/techinf/layer3/index.html
- [22] S. Hacker, *MP3: The Definitive Guide*, chapt. 2 How MP3 Works: Inside the Codec, http://www.oreilly.com/catalog/mp3/chapter/ch02.html, O'Reilly, March 2000
- [23] I. Peterson, *Hiding in DNA*, Science News Online, http://63.240.200.111/articles/20000408/mathtrek.asp, April 2000
- [24] D. Artz, "Digital Steganography: Hiding Data within Data", Los Alamos National Laboratory, http://www.cc.gatech.edu/classes/AY2003/cs6262_fall/digital_steganography.pdf, May
 - 2001
- [25] J. Callinan and D. Kemick, "Detecting Steganographic Content in Images Found on the Internet", Department of Business Management, University of Pittsburgh at Bradford, http://www.chromesplash.com/jcallinan.com/publications/steg.pdf

- [26] N. Provos and P. Honeyman, "Detecting Steganographic Content on the Internet", CITI Technical Report, http://www.citi.umich.edu/techreports/reports/citi-tr-01-11.pdf, August 2001
- [27] F. A. P. Petitcolas and R. J. Andeson, "Weaknesses of copyright marking systems", *Multimedia and Security Workshop at ACM Multimedia '98*, September 1998
- [28] G. Voyatzis, N. Nikolaidis and I. Pitas, "Digital Watermarking: An Overview", Department of Informatics, University of Thessaloniki, http://citeseer.ist.psu.edu/cache/papers/cs/854/http:zSzzSzposeidon.csd.auth.grzSzpap erszSzPUBLISHEDzSzCONFERENCEzSzVoyatzis98azSzVoyatzis98a.pdf/voyatzis98di gital.pdf/
- [29] S. Cacciaguerra and S. Ferretti, "Data Hiding: Steganography And Copyright Marking", Department of Computer Science, University of Bologna,
- http://www.cs.unibo.it/people/phd-students/scacciag/home_files/teach/datahiding.pdf [30] H. Berghel, and L. O'Gorman, "Digital Watermarking",
- http://www.acm.org/~hlb/publications/dig_wtr/dig_watr.html, January 1997