Data Transmission Security:

A Steganographic Approach

Eric Conrad

CSU Stanislaus

CS 4960

Dr. Melanie Martin

1. Introduction

In the computer science field security has become a hot button issue. With the growth of cyber terrorism, companies must look to alternative ways to transmit their data. There is a two-step process being used around the internet today know as encryption and steganography. Encryption is a process that takes a message and makes it unintelligible for someone if they are able to get the data. Steganography is a process that hides the message within another message. Each technique offers strengths and weaknesses that influence their use for security. This paper will discuss some of the steganographic techniques that have been developed.

Steganography is defined as hiding a message within a larger message in such a way that the presence of the secret message is unknown or is unintelligible to all but the intended recipient (Saha, 2013). The main goal of steganography is to keep your data hidden from potential hackers (Yugala, 2013). The data can be hidden in two ways: it can be hidden and not visible to the human eye, or it can be visible and still not visible to the human eye (Yugala, 2013). When the data is visible the focus must be shifted away to keep it hidden.

There are some guidelines to follow when embedding data into a cover mediums data. The cover data should not be significantly degraded as this will alert outsiders to the fact that something has been altered. The data should be directly embedded within the data instead of a header or wrapper to maintain data consistency across formats (Yugala, 2013). When the data is transmitted distortion and degradation should be expected and use of error correcting codes should be used. The embedded data should also be self-clocking or arbitrarily re-entrant to ensure that even if only a portion of the cover data is available the embedded data may still be retrieved (Yugala, 2013).

1.1 History

Steganography is one of the oldest data protection methodologies with many different techniques that have been continuously developed (Reddy, 2012). It has been used throughout history by many of the dominant countries and their enemies during times of war. One of the earliest examples of this is the ATBASH code that was used in the 2000-1500BC time range (Siva, 2011). The ATBASH code was a substitution cipher for the Hebrew alphabet.

One of the more interesting examples was during WWII in which a German spy sent a message. "*Apparently neutral's protest is thoroughly discounted and ignored. Isman hard hit. Blockade issue affects pretext for embargo on by-products, ejecting suets and vegetable oils.*" This message was decoded by taking the second letter in each word which revealed the message: *Pershing sails from NY June 1* (Yugala, 2013). During this time it would have taken months to figure out this was how messages were being sent and by that time a new method would have been developed.

The most recent sign of digital steganography was used by Osama Bin Laden (Yugala, 2013). There were reports that bin Laden and his associates were using chat rooms and pornographic bulletin boards to send secret messages (Yugala, 2013). It was reported that Mohammed Atta, one of the alleged 9/11 hijackers was repeatedly seen going to a Florida library and downloading pictures of children and Middle Eastern scenes (Yugala, 2013). Authorities have suspected that this was their form of communication.

This field has been developing for centuries and it is only fitting that it would eventually move into the digital age. Since entering into the digital scene it has grown to be used in multiple

facets of data. It can be used with text, images, audio, and video files (Adamy, 2012). This allows for multiple different ways to hide your messages for transmission.

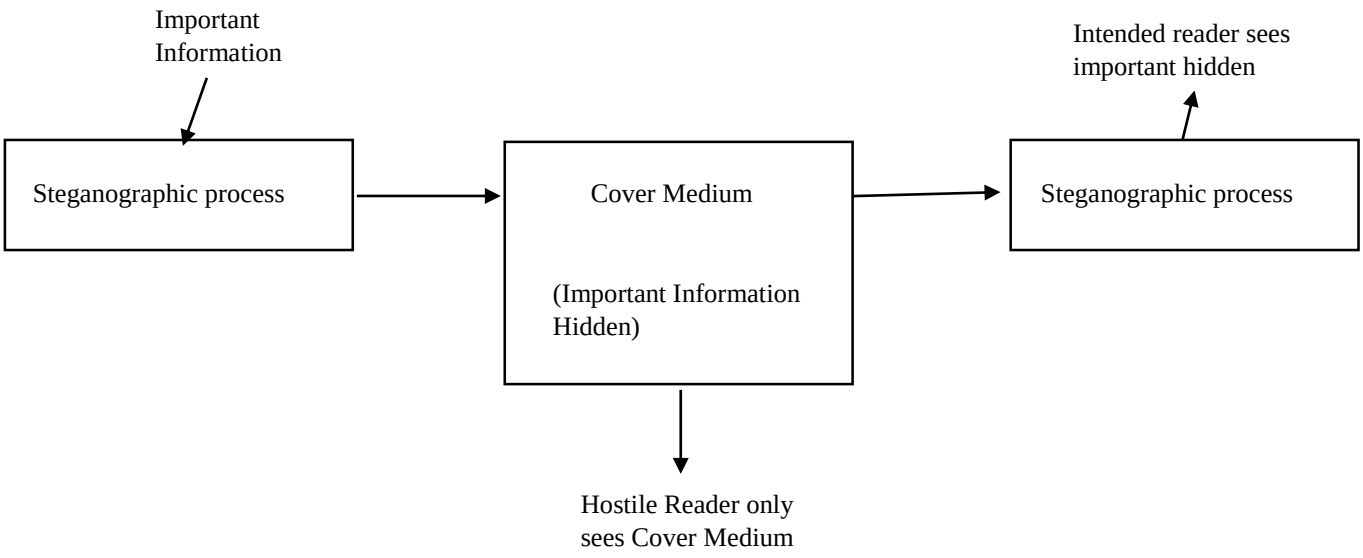

Figure 1: (Adamy, 2012) Basic Process

In figure 1 there is a very basic process for steganography shown. The important information you are trying to send is given to a steganographic process where it is hidden within a cover medium. It is then transmitted looking like the cover medium to prevent outsiders to find the message. It will then be run through another steganographic process on the intended recipients end to get access to the important information.

2. Least Significant Bit

"The most widely used technique to hide data is the usage of the LSB (Verma, 2011)." This process is done by changing the least significant bit, usually the right most bit, to the value that is being hidden.

Using a 24-bit color image with red, green, and blue components, a total of 3 bits can be stored in each pixel. The following example shows a given set of pixels and their binary values and shows the changes when adding the A character to them.

Given:

1. (10010101 00001101 11001001)

2. (10010110 00001111 11001010)

3. (10011111 00010000 11001011)

Now adding (101101101):

1. (10010101 0000110**0** 11001001)

2. (1001011**1** 0000111**0** 1100101**1**)

3. (10011111 00010000 11001011)

The bolded and underlined least significant bits were changed to hide the binary string that was added. Now when reading left to right, top to bottom the binary string that was added is present in the LSB of each binary string. In this case only four bits needed to be changed to insert the character. On average, about half of the bits in an image will need to be modified to hide messages (Verma, 2011).

2.1 Audio LSB

Least significant bit (LSB) encoding is the simplest way to embed information in a digital audio file (Roy, 2012). The ideal transmission rate for LSB encoding is 1 kbps per kHz (Roy, 2012). There is potential in some implementations that two of the least significant bits can be

replaced with two new message bits being added. This increases the amount of data that can be encoded but will increase the amount of noise created. Depending on the content of the audio file the noise may be covered. For example in a subway tunnel the noise would be easily covered, on the other hand noise during a piano solo would definitely be heard by any listener (Roy, 2012).

There are a few ways to handle the encoding process in an audio file. It can be done by starting with the first sample and making changes until the entire message is encoded (Roy, 2012). This is not the best way to do it because it creates a security issue. The issue is that the first part of the file will have different statistical properties than the other part of the file (Roy, 2012). A way to help with this security problem is to pad your message with extra bits to hide that you have made changes, however this actually increases the probability that an attacker would notice a change was made (Roy, 2012).

A second way to encode would be to use a pseudorandom number generator to spread the message throughout the audio file (Roy, 2012). In order to do this the sender and receiver must have the secret key and knowledge of the pseudorandom generation (Roy, 2012). To avoid collisions and overwriting a change that has already been made the receiver must keep track of the samples that have been changed. Another way to avoid collisions is to use a pseudorandom permutation of the entire set using a secret hash function to insure that the same index is never generated more than once (Roy, 2012).

Sample audio stream (16bit)	HEY in binary	Audio stream with message encoded
1001010001001100	0	1001010001001100
1110101011111111	1	1110101011111111
1000000000011011	0	1000000000011010
0111111100101010	0	0111111100101010
0000001110101101	1	0000001110101101
0111010101010101	0	0111010101010100
0111100110101010	0	0111100110101010
0000010101110101	0	0000010101110100
1111010110101011	0	1111010110101010
0111001100101010	1	0111001100101011
1010101011000111	0	1010101011000110
0111110101010101	0	0111110101010100
0111101010101000	0	0111101010101000
0101000101010100	1	0101000101010101
0000000001010100	0	0000000001010100
1111111111111010	1	1111111111111011
0100101010101010	0	0100101010101010
0101010100100010	1	0101010100100011
1111111111111101	0	1111111111111100
0111111111100001	1	0111111111100001
0101010100010101	1	0101010100010101
0101011111111001	0	0101011111111000
0111101010101010	0	0111101010101010
0010010101001010	1	0010010101001011

Figure 2: (Roy, 2012) LSB process encoding the word HEY in a 16bit audio stream

The right most bit in figure 2 is the least significant bit. Figure 2 gives an example of how the audio file looks before and after encoding HEY into the file. Thirteen of the twenty-four samples required the LSB to be changed in order to successfully hide HEY within the sample audio file.

3. Image Based Algorithms

For this type of algorithm a small amount of background is needed. Every image is comprised of a matrix of pixels. Every pixel is represented by four bytes: alpha, red, green, blue. The alpha gives the degree of transparency, red/blue/green gives the intensity of that color in the pixel. This allows for storage of one byte within the alpha byte because it won't affect the color of the image (Yugala, 2013). This allows you to store the message in the alpha bytes of the pixels and transmit a message. At the receiving end, the characters from the pixel are reconstructed to provide a message (Yugala, 2013).

3.1 Text on Image

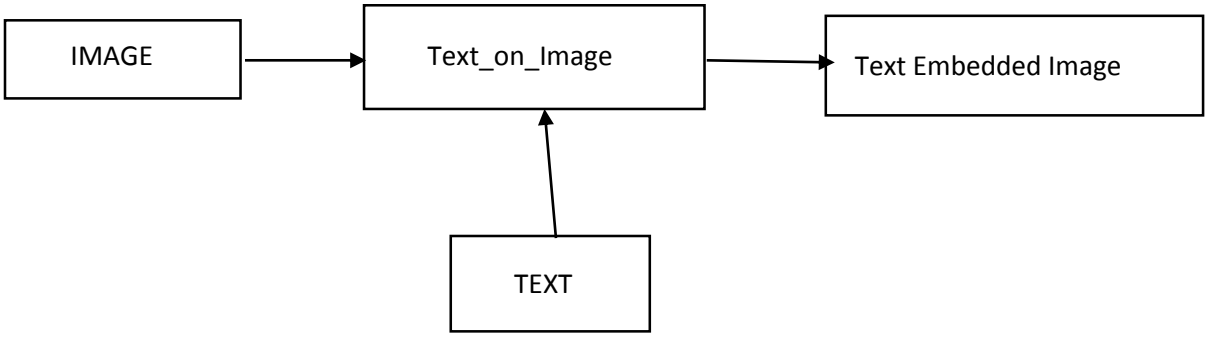

Figure 3 Simple Input-Output (Yugala, 2013)

This process will require an image file and text file and output a text embedded image.

Steps: (Yugala, 2013)

1. Extract all of the pixels from the image and store them into an array named Pixel-Array.
2. Extract all of the characters in the text file and store them into a separate array named Character-Array.
3. This step will be done for every pixel in the array.
    a. If the index of the Pixel-Array is less than the size of the text file:
        i. Store the current character value of the Character-Array in the Alpha field of the current pixel in the Pixel-Array.
    b. Else
        i. Store the value 0 in the alpha field of the current pixel in the Pixel-Array.
    c. Increment the index of both the arrays(Character-Array and Pixel-Array)

This algorithm operates by taking an image file and extracting all the pixels into an array. It also requires a text file and extracts each character into an array. It then uses a recursive function with an if-else statement to store a character inside the alpha field of a pixel. If there are less characters than pixels the remaining pixels will be filled with a zero until the pixel array has been gone through completely.

The above algorithm will only work if the text file is smaller in size than the image file. This algorithm operates by only changing the alpha value of each pixel. By doing this it keeps the changes invisible to the human eye when looking at the picture. This also allows for the text file and image file to be sent within the same time frame. Once the file has been received all that is required is to reverse the algorithm to retrieve the message.

3.2 Image on Image

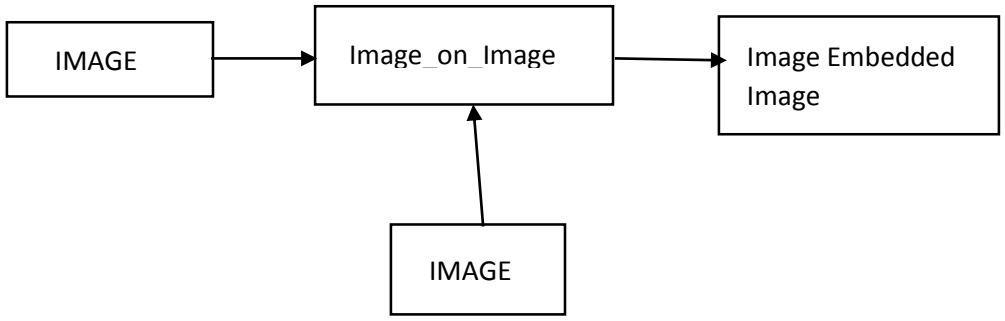

Figure 4 Simple image on image Input-Output (Yugala,2013).

This algorithm requires two image input files and returns one image output file.

Steps: (Yugala, 2013)

1. Extract all of the pixels from the first image and store it in an array called Pixel-Array1.
2. Extract all of the pixels from the second image and store it in an array called Pixel-Array2.
3. Repeat the following for all pixels in Pixel-Array1
    a. If the index of Pixel-Array2 is less than the size of Pixel-Array1
        i. Store the Red Value of the current pixel in the Pixel-Array2 to the Alpha field of the current pixel in the Pixel-Array1.
        ii. Increment the index value of the Pixel-Array1
        iii. Store the Green value of the current pixel in Pixel-Array2 to the Alpha field of the current pixel in Pixel-Array1
        iv. Increment the index of Pixel-Array1

v. Store the Blue value of the current pixel in the Pixel_Array2 to the Alpha field of the current pixel in Pixel-Array1

vi. Increment the index of the Pixel-Array1

b. Else

i. Store the value 0 to the Alpha field of the current pixel in the Pixel-Array1

ii. Increment the index of Pixel-Array1

iii. Store the value 0 to the Alpha field of the current pixel in the Pixel-Array1

iv. Increment the index of Pixel-Array1

v. Store the value 0 to the Alpha field of the current pixel in the Pixel-Array1

c. Increment the index value of the Pixel-Array2

This algorithm is pretty simple in how it operates. The images are taken and all pixels are put into two different arrays. It then gets the first pixel of each array to begin the process of steganography. It uses a recursive function with an if-else statement to determine when the secret message has been fully embedded. Once the secret message has been fully embedded, it sets the rest of the alpha values in the cover image to zero and the process is finished.

This algorithm has a restriction that it will only work if the second image, or secret image, file is one third the size of the first image. This happens because it is necessary to use three pixels to store one pixel of the secret image. Again with this algorithm the receiver must reverse this algorithm to retrieve the secret image (Yugala, 2013).

4. K-means algorithm

The main goal of the k-means clustering method is to divide the data points of a data set into k clusters such that the distance of the data points to the centroids of the clusters is minimized (Xu, 2014). This is a two stage algorithm: the initial stage and the iterative stage (Xu, 2014). The first stage consists of setting the k initial centroids of the clusters (Xu, 2014). The second stage consists of assigning the data points to the nearest centroid and calculating the k new centroids according to the new assignments (Xu, 2014). Since K-means is a heuristic algorithm, there is no guarantee that it will converge to the global optimum, and the results may depend on the initial clusters (Xu, 2014).

The main goal for this experiment is to use an **M x N** gray scale cover image and send a secret message of **L** bits (Xu, 2014). This will be done using the LSB technique to hide the secret image within the image. To get the best results the secret image must be store non-uniformly to each pixel of the cover image. This means that some pixels may receive more bits during this process. This algorithm looks to address two key issues:

1. All the bits L are assigned to the pixels of the cover image
2. The assignment of the bits to the pixels is optimal such that the new stego-image is optimally similar to the original cover image (Xu, 2014).

4.1 Optimal K-means Based Steganograpy Algorithm

The first step is to uniformly distribute the L number of bits into the M x N pixels. To do this we use the $\frac{L}{MxN}$ equation to set the least significant bits of the pixels to the secret message. The next step is to define the distance between each bit to each cluster so the secret message bits can be reassigned to different pixels.

For the *i*th cluster, $0 \leq i < M \times N$, we assume the cluster assigned to it has bits $B_i = \{B_i^1, B_i^2, B_i^3, \ldots.\}$ (Xu, 2014). We define $D_i^k{}_j$ between the bit B_i^k and cluster B_j as follows:

$$D_{ij}^k = \begin{cases} f(i,j,k), & k = 1 \text{ and } j = i-1 \\ g(i), & k = 1 \text{ and } j = i \\ f(i,j,k), & k = |B_i| \text{ and } j = i+1 \\ g(i), & k = |B_i| \text{ and } j = i \\ 0, & i = j \\ \infty, & |j-i| > 1 \end{cases}$$

Figure 5 (Xu, 2014) Equations for moving the pixels to the closest centroid.

where $|B_i|$ is the number of bits of B_i. Function g(*i*) is defined as the Gaussian curvature of the current stego-image at the pixel *i*, $0 \leq i < M \times N$ (Xu. 2014). The g(*i*) function is also defined as the centroid of the *i*th cluster because it provides the overall contribution of each bit of cluster B_i to the Gaussian curvature at pixel *i* (Xu. 2014). F(i,j,k) is defined as the Gaussian curvature of the current stego-image at the pixel *i* after the bit B_i^k is moved from the *i*th cluter to the *j*th cluster (Xu, 2014).

Given an initial assignment of clusters and a new definition of distance and centroid, the K-means algorithm may now be used to refine the clusters (Xu, 2014). We must first assign the L bits to the M x N pixels uniformly to get the initial set of M x N clusters. The next step, named reassignment, is called to assign each bit to the cluster whose distance to that cluster is the shortest (Xu, 2014). By definition of $D_i^k{}_j$ bits may only move among neighboring clusters (Xu, 2014). The centroid is now calculated after reassigning all the bits. The updated centroid for cluster B_i is the new curvature at the pixel *i* (Xu, 2014). The reassignment step is called until

there are no more bits that require movement to a different cluster. Upon completion the resulting stego-image now has the optimal LSB assignment for the secret message and cover image (Xu, 2014).

4.2 Testing K-Means

To test the quality of the images made from the K-means algorithm there will be three metrics to score the outcome. These metrics are Peak Signal to Noise Ratio(PSNR), Mean Square Error(MSE), and Correlation Coefficient(CORR). These test will test the levels of distortion and differences between cover and stego-image.

The PSNR is used to measure the distortion of the image (Xu, 2014). It does this by a comparison of the image quality between the cover image C and the stego-image S (Xu, 2014).

$$PSNR(C,S) = 10\log_{10}\frac{(2^d - 1)^2}{MSE},$$

Figure 6 (Xu, 2014) Equation for distortion testing.

The d is the bit depth of the cover image and is equal to 8 for gray scale images (Xu, 2014). When evaluating the PSNR the higher the number the better quality of the stego-image produced (Xu, 2014). For example a PSNR below 30 dB implies low image quality and the embedding distortion can be obvious (Xu, 2014).

The MSE is the cumulative mean square error between the cover and stego-image (Xu, 2014).

$$MSE(C,S) = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}(S_{ij} - C_{ij}),$$

Figure 7 (Xu, 2014) Equation for the MSE.

In this equation S_{ij} and C_{ij} denote the pixel values of the cover image and the stego-image. The M and N are representing the dimensions of the cover image.

The correlation coefficient will be used to measure the differences between the cover image and the stego-image.

$$CORR(C,S) = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N}(C_{ij}-\overline{C})(S_{ij}-\overline{S})}{\sqrt{(\sum_{i=1}^{M}\sum_{j=1}^{N}(C_{ij}-\overline{C})^2)(\sum_{i=1}^{M}\sum_{j=1}^{N}(S_{ij}-\overline{S})^2)}}$$

Figure 8 (Xu, 2014) Equation for testing the cover and stego-image for similarity/differences

The Cbar is the average of all pixel values in the cover image, and the Sbar is the average of all pixel values in the stego-image (Xu, 2014). The CORR value will be between [-1, 1]. If the value is 1 it means the cover image and stego-image are so similar that there is no big differences (Xu, 2014). If the value is -1 it means the two are far from similar and there are huge differences (Xu, 2014).

Cover image	Stego-image	MSE	PSNR (dB)	CORR
Figure 1 (a)	Figure 1 (b)	354.20	52.13	0.97
Figure 2 (a)	Figure 2 (b)	221.12	56.84	0.99
Figure 3 (a)	Figure 3 (b)	366.84	51.78	0.99
Figure 4 (a)	Figure 4 (b)	273.63	54.71	0.98

Figure 9 (Xu, 2014) Values after running pictures from Figures 10

4.3 Results of K-Means

Experiments were conducted on the images in figure 10. Figure 10 shows the cover image, secret image, stego-image, and the bits assignment. The bits assignment image is exactly

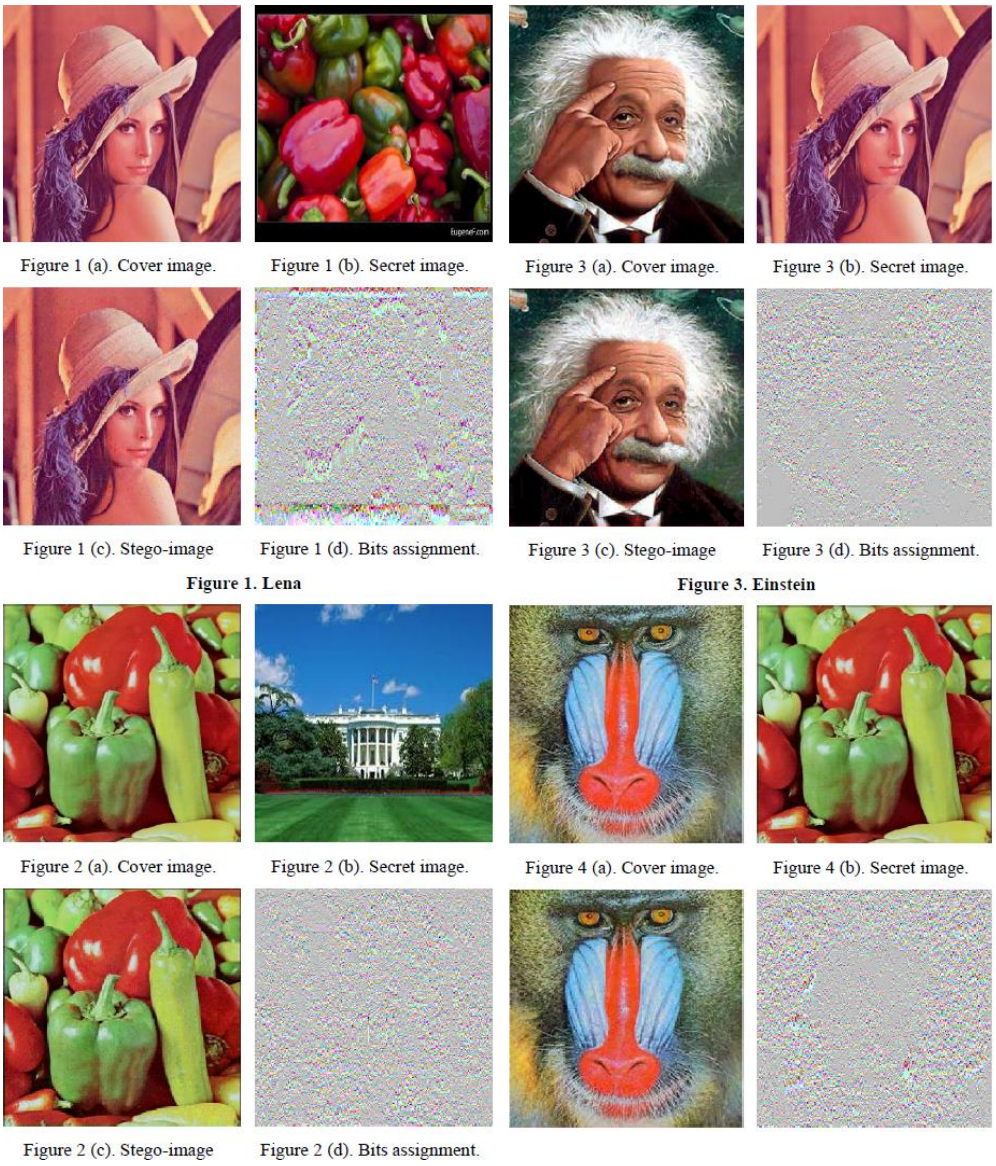

Figure 10 (Xu, 2014) Group of images tested to see if the algorithm worked.

60% of the stego-image. This pattern is created when the secret images bits are assigned to the cover images LSB. After looking at Figure 9 all four images have a PSNR (dB) rating of 50 or

above showing that this algorithm can produce high quality stego-images. All of the CORR values in the table are near 1 meaning the images are very similar.

5. Advantages

The main advantage to using steganography is to conceal information without showing that changes have occurred. The transmission of secret messages are unknown unless the person is actively looking and knows what to look for. Messages can be sent in a variety of formats and can combine multiple formats into one format. When used in combination with encryption agencies can communicate in secret and still protect their communication if discovered. When using encryption the encryption key is necessary to unlock the secret message in order to reverse the steganographic process. This adds a layer of security on top of the secret message just in case the message is discovered.

6. Disadvantages

The biggest disadvantage to using these techniques is the fact that the message is highly vulnerable when using a known technique. If the technique being used is a well-known one anyone looking could easily obtain the message. Another potential disadvantage is that the new stenographic image size is larger than the original. There can be noticeable color changes when using well known pictures that can lead to the data being found. Through the use of new technologies some internet firewalls have the ability to detect steganographic messages and either delete them or quarantine them (Yugala, 2013). When changing the type of the format or replacing readable text can alter the text in the secret message you wish to transmit.

7. Conclusion

In the field of data security companies and users must always be looking for ways to protect the data they are sending. With the need for protection two fields developed: Cryptography and Steganography. Cryptography was used for message security, however this lets everyone know you are sending a message and increases the likelihood of an attack. Steganography was used for transmission security. This technique lets the user hide their message and the fact that it has been sent in the first place.

Steganography can be an effective way to transmit secret messages between one another. Using steganography requires adherence to strict rules to prevent the message from being noticed. When the guidelines are met the messages can flow with minimal problems. When the guidelines are not followed the risk for detection increases and make this technique less useful but encryption can be added on top to bring it back.

"The most widely used technique to hide data is the usage of the LSB (Verma, 2011)." All techniques discussed in this paper used the LSB but applied it in a different way or used it to piece together different data types. It is also the most widely used because it is such an easy concept and only requires the flipping of a bit per sample. It also holds that on average only half the LSB bits will require flipping when hiding your message.

References

Adamy, Dave. "Steganography." *Journal of Electronic Defense* 35.10 (2012): 47. Web. 3 May 2015.

Lokeswara Reddy, V., A. Subramanyam, and P. Chenna Reddy. "Implementation of Least Significant Bit Steganography and Statistical Steganalysis." *CCSEIT* (2012): 671-75. *ACM Digital Library*. Web. 3 May 2015.

Roy, Sangita, Jyotirmayee Parida, Avinash K. Singh, and Ashok S. Sairam. "Audio Steganography Using LSB Encoding Technique with Increased Capacity and Bit Error Rate Optimization." *CCSEIT* (2012): 372-76. Web. 3 May 2015.

Saha, Arijit, Nilotpal Manna, and Surajit Mandal. "9.12 Steganography."*Information Theory, Coding and Cryptography*. India: Pearson, 2013. N. pag. Print.

Siva Sankar, A., T. Jayachandra Prasad, and M. N. Giriprasad. "LSB Based Image Steganography Using Polynomials and Covert Communications in Open Systems Environment for DRM." *ICWET* (2011): 593-97. *ACM Digital Library*. Web. 3 May 2015.

Verma, N. "Review of Steganography Techniques." *ICWET* 2011: 990-993. Print.

Xu, Shuting, and Shuhua Lai. "An Optimal Least Significant Bit Based Image Steganography ALgorithm."*ICIMCS* (2014): 10-12. Web.

Yugala, K. "Steganography." *IJETT* 4.5 (2013): n. pag. Web.