Advances in Science, Technology and Engineering Systems Journal Vol. 3, No. 2, 271-282 (2018)

www.astesj.com

Special Issue on Advancement in Engineering Technology

ASTESJ

ISSN: 2415-6698

# Steganography System with Application to Crypto-Currency Cold Storage and Secure Transfer

Michael J. Pelosi[*,1], Nimesh Poudel, Pratap Lamichhane[1], Danyal Badar Soomro[2]

[1]*Computer Science and Technology Faculty, East Central University, 74820, USA*

[2]*School of Software Engineering, Chongqing University, Chongqing 400044, China*

A R T I C L E I N F O

*Article history:*
*Received: 15 November, 2017*
*Accepted: 05 February, 2018*
*Online: 25 April, 2018*

*Keywords:*
*Steganography*
*One-time pad*
*Steganalysis*
*Information hiding*
*Digital forensics*

A B S T R A C T

*In this paper, we introduce and describe a novel approach to adaptive image steganography which is combined with One-Time Pad encryption and demonstrate the software which implements this methodology. Testing using the state-of-the-art steganalysis software tool StegExpose concludes the image hiding is reliably secure and undetectable using reasonably-sized message payloads (≤25% message bits per image pixel; bpp). Payload image file format outputs from the software include PNG, BMP, JP2, JXR, J2K, TIFF, and WEBP. A variety of file output formats is empirically important as most steganalysis programs will only accept PNG, BMP, and possibly JPG, as the file inputs. In this extended reprint, we introduce additional application and discussion regarding cold storage of crypto-currency account and password information, as well as applications for secure transfer in hostile or insecure network circumstances.*

## 1. Introduction

In this paper, we introduce a comprehensive steganography software system and platform framework based on One-Time Pad (OTP) encryption and adaptive steganography technology. We provide usage recommendations and advice guidelines. The system is tested and shown to be resistant to many common steganalysis attacks. In the context of this paper we are assumed advocate of the steganography; someone who may be a political dissident in an oppressive regime, a religiously persecuted individual, a friendly agent engaging in covert communication, or a lawful individual desiring complete communication privacy, among other compelling examples.

## 2. What is Steganography?

Steganography, the art of invisible communication, is achieved by hiding secret data inside a carrier file such as an image. After hiding the secret data, the carrier file should appear unsuspicious so that the very existence of the embedded data is concealed. A major drawback to encryption is that the existence of the message data is not hidden. Data that has been encrypted, although unreadable, still exists as a suspicion arousing file transfer. If given enough time, once alerted, someone could potentially decrypt the data or derive other intelligence regarding either sender or receiver. A solution to this problem is steganography. This is the ancient art of hiding messages so that they are not detectable.

In steganography, the possible cover carriers are unsuspicious appearing files (images, audio, video, text, or some other digitally representative code) which will hold the hidden information. A message is the information hidden and may be plaintext, cipher text, images, or anything that can be embedded into a bit stream. Together the cover carrier and the embedded message create a stego-carrier. Hiding information may require a stego key which is additional secret information, such as a password or OTP key, required for embedding the information. For example, when a secret message is hidden within a cover image, the resulting product is a stego-image. A possible formula of the process may be represented as: cover medium + embedded message + stego key = stego-medium.

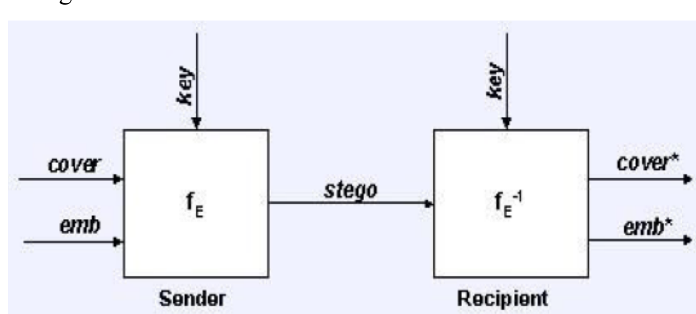



Fig:1 Graphical Version of Steganographic system

*Corresponding Author: Michael J. Pelosi, East Central University, Email: mpelosi@ecok.edu

$f_E$: steganographic function "embedding".
$f_E^{-1}$: steganographic function "extracting".
cover: cover data in which emb will be hidden.
emb: message file to be hidden.
stego: cover data w

The advantage of steganography is that it can be used to secretly transmit messages without the fact of the transmission being discovered. Often, using encryption might identify the sender or receiver as someone with something to hide. It is believed that steganography was first practiced during the Golden Age in ith the hidden message. Greece. An ancient Greek record describes the practice of melting wax off wax tablets used for writing messages and then inscribing a message in the underlying wood. The wax was then reapplied to the wood, giving the appearance of a new, unused tablet. The resulting tablets could be conveniently transported without anyone suspecting the presence of a message beneath the wax.

### *2.1. LSB Steganography*

The simplest and popular image steganographic method is the least significant bit (LSB) substitution. It embeds messages into cover image by replacing the least significant bits directly. The hiding capacity can be increased by using up to 4 least significant bits (one each for Red, Green, Blue, and Alpha color channels, respectively) in each pixel. It has a common weak point i.e. the sample value changes asymmetrically. When the LSB of cover medium sample value is equal to the message bit, no change is made. Otherwise the value 2n is changed to 2n+1 or 2n+1 is changed to 2n. There are many improvements and modifications that have been proposed to strengthen this technique, such as adaptive techniques that alter payload distribution based on image characteristics. If the message is first encrypted and then embedded, the security is enhanced.

### *2.2. One-Time Pad*

The "one-time pad" encryption algorithm was invented in the early 1900's, and has since been proven as unbreakable. The one-time pad algorithm is derived from a previous cipher called Vernam Cipher, named after Gilbert Vernam. The Vernam Cipher was a cipher that combined a message with a key read from a paper tape or pad. The Vernam Cipher was not unbreakable until Joseph Mauborgne recognized that if the key was completely random the cryptanalytic difficulty would be equal to attempting every possible key (Kahn 1996). Even when trying every possible key, one would still have to review each attempt at decipherment to see if the proper key was used. The unbreakable aspect of the one-time pad comes from two assumptions: the key used is completely random and the key cannot be used more than once. The security of the one-time pad relies on keeping the key secret and using each key only once.

The one-time pad is typically implemented by using exclusive-or (XOR) addition to combine plaintext elements with key elements. An example of this is shown in Figure 2. The key used for encryption is also used for decryption. Applying the same key to the cipher text results in the output of the original plaintext.

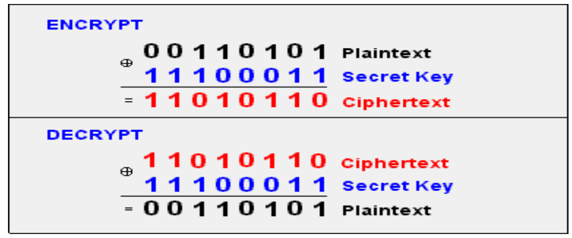



Fig.2 Examples of a One-time Pad implementation using XOR addition.

OTP is immune even to unlimited resources brute-force attacks. Trying all keys simply yields all possible plaintexts, all equally likely to be the actual plaintext. Even with known plaintext, such as part of the message being known, brute force attacks cannot be used, since an attacker is unable to gain any information about the parts of the key needed to decrypt the rest of the message.

## 3. Methodology

The following describes the general method implemented in the software for key generation, encryption, embedding, message transfer, and decryption.

- Random image keys are generated using a key generator program. The key generator program generates one-time pad keys that consist of random colored pixels. Each random colored pixel consists of random values for red, green, and blue colors throughout the image. One image key is generated for every message that is intended to be sent.
- To encrypt a message, a cover image and random key image is selected. Each pixel in the cover image is XOR'ed with the key image X, Y coordinate pixel. Each pixel consists of a 32-bit long integer color value. One byte each corresponds to red, green, and blue components, respectively. The XOR'ed pixel values are then adjusted to hide the message. The bytes in the message are divided up into bits — one bit per pixel. The least significant bit (LSB) in the XOR'ed pixel colors are then adjusted to hide the message. Bit values that do not correspond are adjusted (in general 50% of the values will already be set correctly). LSB's for red, green, or blue are selected based on a local pixel variation score, contingent if the sum of the RGB LSB's are even or odd (even corresponds to a 0 bit, odd to a 1 bit).
- At this point, the newly derived color values are XOR'ed once again with the random image key to generate color values very close to the original image. These pixel color values will be used to construct the steganographic image that will be sent to the receiver.
- Ideally at this point, both the original cover image and the senders copy of the random image key can be destroyed (forensically wiped from the hard drive using a file erasure procedure). This is to prevent later detection and statistical comparisons.
- Upon receipt of the steganographic image, the receiver loads the intended image key and XOR's each pixel of the steganographic image with its respective corresponding X, Y pixel in the image key. This will derive a series of bit values

that correspond to the plaintext message. The bits can be reassembled into bytes (and later 2-byte Unicode characters) that correspond to the plaintext message.

- The start and end of the message are delimited by randomly chosen 10-character delimiting strings that are embedded as EXIF comments into the random image key by the key generator program. Thus, random message padding is incorporated at the start and end of messages.
- The random image key also contains a random number seed, this is used for the random number generator algorithm in use and starts the generator at the proper sequence start value.

### *3.1. Random Number Generation*

A cryptographically secure pseudo-random number generator (CSPRNG) or cryptographic pseudorandom number generator (CPRNG) is a pseudorandom number generator (PRNG) with properties that make it suitable for use in cryptography. Ideally, the generation of random numbers in CSPRNGs uses entropy obtained from a high-quality source, which might be a hardware random number generator or perhaps unpredictable system processes — though unexpected correlations have been found in several such ostensibly independent processes. Several robust CPRNGs are incorporated into the steganography software.

#### *3.1.1 Mersenne Twister*

The Mersenne Twister is a pseudorandom number generator (PRNG). It is by far the most widely used general-purpose PRNG. Its name derives from the fact that its period length is chosen to be a Mersenne prime. The Mersenne Twister was developed in 1997 by Makoto Matsumoto and Takuji Nishimura. It was designed specifically to rectify most of the flaws found in older PRNGs. It was the first PRNG to provide fast generation of high-quality pseudorandom integers. The most commonly used version of the Mersenne Twister algorithm is based on the Mersenne prime 219937−1. The standard library implementation of this, MT19937, uses a 32-bit word length. There is another implementation that uses a 64-bit word length, MT19937-64, that generates a different sequence. The software implements a cryptographically secure version of the Mersenne Twister provided by the algorithm authors Matsumoto and Nishimura.

#### *3.1.2 Other Random Number Generators*

Optional random number generator selections included in the OTP-Steg key generator program include the following (each of these can be optionally selected by the user):

- ISAAC— ISAAC (indirection, shift, accumulate, add, and count) is a cryptographically secure pseudorandom number generator and a stream cipher designed by Robert J. Jenkins, Jr. in 1996.
- CryptGenRandom-CryptGenRandom is a cryptographically secure pseudorandom number generator function that is included in Microsoft's Cryptographic Application Programming Interface. In Win32 programs, Microsoft recommends its use anywhere random number generation is needed.
- RtlGenRandom — On a default Windows XP and later install, CryptGenRandom calls into a function named ADVAPI32!RtlGenRandom, which does not require one to load all the CryptAPI classes for usage.
- Rnd () — Standard API random number generator (for research/testing purposes only – it is not cryptographically secure).

### *3.2 Key Delimiters*

Upon key generation, a pair of key delimiters is also randomly chosen of 10 Unicode characters each for the start delimiter and end delimiter, respectively. These are used to indicate to the decryption program exactly where the message starts, and where it ends. Random padding is added to both ends of the message — the start and the end of the message embedded in the payload file. The key delimiters identify where to start the message text, and where to cut it short at the end of the message. These key delimiters are contained in the EXIF image comment data in the key file. No EXIF comment data whatsoever is contained in the payload file. Also, the key delimiter values are utilized for random number generation seed data used for encryption and decryption.

### *3.3 Expert System to Evaluate and Score Candidate Cover Images*

It is well known from the literature that some cover images present much better candidates for steganographic security than others based on image characteristics. Typically, cover images with a high degree of pixel color variation, very few saturated white or black pixels, and few pixels next to each other of the same color, are excellent payload candidates. We implement an expert system to give the software user immediate knowledge of how good a candidate a potential color image is for detection security. We have incorporated a tentative scoring system that evaluates images based on several factors. The output score ranges from 0 to 100%, with greater than 90% score being a good candidate for a cover image. Scores of 80-90% are marginal, and less than 80% are considered not adequate. In the current preliminary version, a peak signal-to-noise ratio (PSNR) versus a solid color image is calculated. This rating is given a weighting of 25% in the overall score. Also, the number of same color pixels next to each other is given a weighting of 25% for up to 5% of the image pixels (in other words, a 5% of the image pixels are same color next to each other, this rating would be zero). Thirdly, a weighted rating of 25% is given to the number of white pixels, up to 5%. The same weighting is also calculated for black pixels. Each of the four factors is combined for the rating from 0% up to 100%. Ideally, a cover image will have zero white pixels, zero black pixels, very few colors next to each other that are the same, and a very high variation in color over comparison to a solid color image. Table 1 below lists the above and additional cover image scoring factors that could be evaluated in an expert system rating scheme.

### *3.4 Future Security = Small Payloads*

To ensure robustness against potential future attacks we have limited payload relative sizes. The high limit for the bits-per-pixel pixel is approximately 25%. And since only half of pixels are typically altered based on the message, this corresponds to a practical limit of about 12.5%-pixel alteration. By limiting the pixel bit payload, it quite robustly limits detectability now and in the future.

Table 1. Potential Candidate Image Scoring Factors.

| Factor | Description | Value |
|---|---|---|
| PSNR over solid color | Peak signal-to-noise ratio of image to solid color image. | Higher values are better. |
| Percentage of saturated colors | Portion of the image that is either all-white or all black. | Lower values are better. |
| Percentage of nearby same colors | Portion of image that has neighboring pixels of the same color. | Lower values are better. |
| Randomness of LSB's | Measures of randomness of the distribution of the significant bits. | Higher randomness is better. |
| Random RGB LSB distribution | Randomness of each color channel. | Higher values are better. |
| RS test on Cover Image | Clean RS test on cover image. | Lower values are better — indicates less probability of a threshold being reached after encoding. |
| Chi-squared test on Cover Image | Clean Chi-square test on cover image. | Lower values are better. |
| Pure Photograph | Photo has not previously been compression encoded using algorithm like JPEG. | Straight from a high-quality digital camera is best. |
| Original Photograph | No other copies of the photo exist in clean or altered state that can be used for comparison. | Known source and originality is best here. |
| Dimensions | It is well known that extremely large images have less pixel color variation and steganography here is more easily detected. | Approximate pixel dimensions of images frequently found on the Internet are best — about 1600?200 or less pixels. |

Extremely advanced statistical detection techniques are being promulgated that are improving the odds of successfully detecting steganography efforts. There is no guarantee that these steganalysis efforts will not double or triple in effectiveness in the next few years. As a safety measure and margin of security, payload size is strictly limited by the software to an amount that should be reasonably safe for the foreseeable future. This equals future security for messages that may be encrypted today and subsequently intercepted and archived for several years for later decipherment.

### *3.5 Steganalysis*

Steganalysis is "the process of detecting steganography by looking at variances between bit patterns and statistical norms". It is the art of discovering and revealing covert messages. The goal of steganalysis is to identify suspected information streams, determine whether or not they have hidden messages encoded into them, and, if possible, recover the hidden information. Unlike cryptanalysis, where it is evident that intercepted encrypted data contains a message, steganalysis generally starts with several suspect information streams but uncertainty whether any of these contain hidden message. The steganalyst starts by reducing the set of suspect information streams to a subset of most likely altered information streams. This is usually done with statistical analysis using advanced statistics techniques.

Analyzing repetitive patterns may reveal the identification of a steganography tool or hidden information. To inspect these patterns an approach is to compare the original cover image with the stego image and note visible differences. This is called a known-carrier attack. By comparing numerous images, it is possible that patterns emerge as signatures to a steganography tool. Another visual clue to the presence of hidden information is padding or cropping of an image. With some steganographic tools if an image does not fit into a fixed size it is cropped or padded with black spaces. There may also be a difference in the file size between the stego-image and the cover image. Another indicator is a large increase or decrease in the number of unique colors, or colors in a palette which increase incrementally rather than randomly. These are just examples among the many published and effective approaches.

StegExpose is a steganalysis tool specialized in detecting LSB (least significant bit) steganography in lossless images such as PNG and BMP. It has a command line interface and is designed to analyze images in bulk while providing reporting capabilities and customization which is comprehensible for non-forensic experts. The StegExpose rating algorithm is derived from an intelligent and thoroughly tested combination of pre-existing pixel based steganalysis methods including Sample Pairs by Dumitrescu (2003), RS Analysis by Fridrich (2001), Chi-Square Attack by Westfeld (2000) and Primary Sets by Dumitrescu (2002). In addition to detecting the presence of steganography, StegExpose also features the quantitative steganalysis (determining the length of the hidden message). We utilize StegExpose for steganalysis to test the software reliability in hiding messages effectively from steganalysis.

### 3.6 Performance Speed and Robust Steganography

The straightforwardness of the embedding algorithm has also resulted in the good embedding speed. Most of the files worked with using the software take less than 60 to 90 seconds for embedding. Typically, about 30 seconds is required for decryption. Since the bit per pixel payload is less than 25%, the random number generator does not have to repeatedly struggle to find empty pixels that have not been previously encoded.

## 4. Software Implementation

The software implementation consists of three executable files: a key generator program, an encryption program, and a decryption program. The encryption program has image analysis functions and windows built-in to aid in cover image categorization.

### 4.1. Key Generation

A screenshot of the key generation program shown below. The key generation program constructs image keys of random colored pixels according to the user preference for size and file naming. Up to five previously discussed random number generators can be chosen from to generate the random colored pixels.

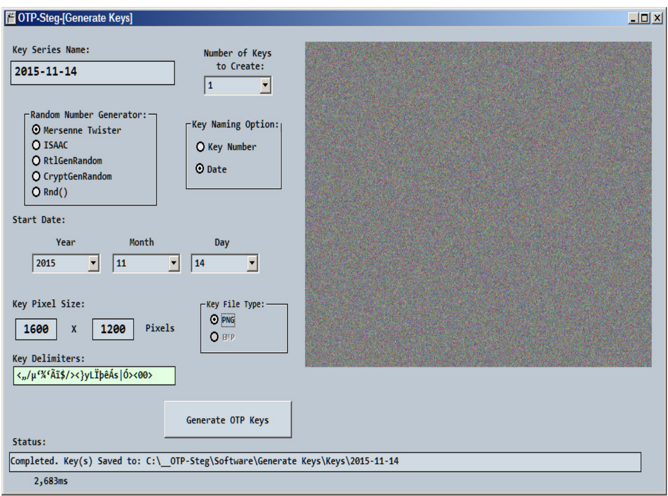



Figure 4. Key Generator executable program

### 4.2. Encryption and Embedding

The encryption program has by wide margin the most features and functionality built-in. Also included are functions for deleting and forensically wiping the key file used for encryption, as well as the original cover image. By comparing the encrypted payload file to an original cover image, steganography could easily be detected as the difference between the two images. It is an extremely important security measure to eliminate the original cover image and key as soon as possible after encryption takes place.

### 4.3. Message Hash Value

SHA-256 values of the message are calculated in both the encryption and decryption steps. In the encryption step, the hashed value is incorporated into the end of the message string. Upon decryption, the transmitted hash value is compared to the hashed value of the decrypted message, and displayed in the decryption program graphical user interface. If the values match the user is informed that the message has not been altered in any way since it was encrypted by the sender. This is also a double check that successful decryption has taken place and the message is authentic.

### 4.4. Text Compression:

Compression prior to embedding the message generally reduces message size by 50 to 80%. The zLib compression library DLL is utilized and called as a function within both the encrypt and decrypt programs. The result makes encryption and decryption quicker and also has the benefit of reducing the bit per pixel payload size in the cover image, increasing security against detection.

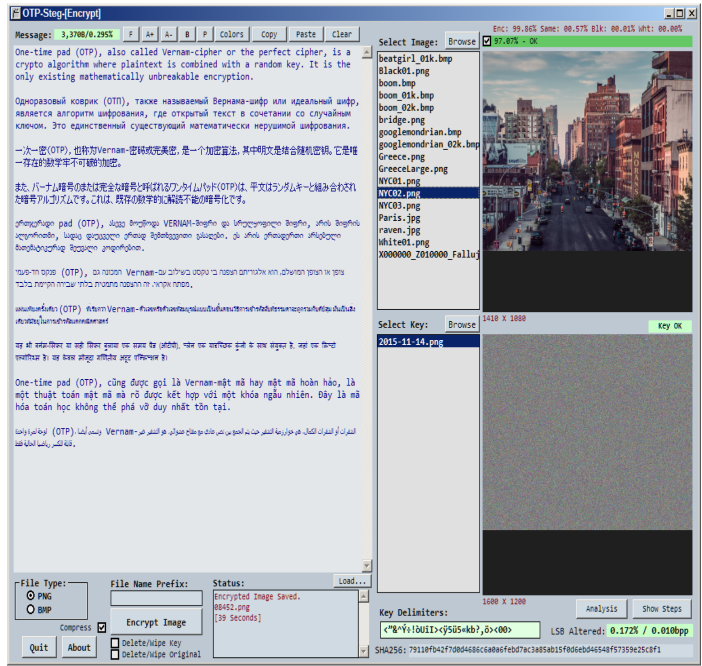



Figure 5. Encrypt/Embed executable program

### 4.5. Cover and key File Deletion and forensic Data Wiping

File wiping utilities are used to delete individual files from an operating system mounted drive. The advantage of file wiping utilities is that they can accomplish their task in a relatively short amount of time as opposed to disk cleaning utilities which take much longer and must be run separately.

### 4.6. Built-In Cover and Stego Image Analysis Tools

Several image statistical analysis features are built into the encryption program. Peak signal-to-noise ratio (PSNR), RS, Chi-Squared, LSB visual analysis, color changes, and color variations. The distortion in the stego-image can be measured by parameters such as mean square error (MSE) and peak signal-to-noise ratio (PSNR) (see Equation 1 and 2 below), and correlation. The lesser distortion means, the lesser MSE, but higher PSNR. If p is a M 譔 grayscale image and q is its stego-image, then the MSE and PSNR values are computed using (1) and (2). For color images a pixel comprises 3 or 4 bytes. Each byte can be treated as a pixel and the same equations can be used to calculate the MSE and PSNR.

$$\mathrm{MSE} = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} \left(p_{ij} - q_{ij}\right)^2 \tag{1}$$

$$\mathrm{PSNR} = 10 \times \log_{10} \frac{C_{max}^2}{\mathrm{MSE}} \tag{2}$$

The software image analysis window in the encryption program is shown below. Using this window, several operations can be performed to estimate effectiveness of message embedding. Least significant bit (LSB) color values can be investigated visually. Shown on the right of the analysis window are the least significant bit values of the photo on the left. If the least significant bit for red, green, and/or blue is set, this color is added at full intensity to the respective pixel in the image on the right. In Figure 7, individual least significant bit color values can be investigated as well in the red, green, and blue channels. In this image, it is obvious there is a problem with the blue channel — the sky has full intensity for all values. Encoding message data here would be risky, as the pixel variation is nonexistent. Steganography would be very easily detected by any encoding in this area. As a result, the software spreads out the message embedding adaptively, and ignores the blue channel in the area of the sky. Since the red channel has the most random variation throughout the image, it carries the largest brunt of the payload, leveraging its random character throughout the image.

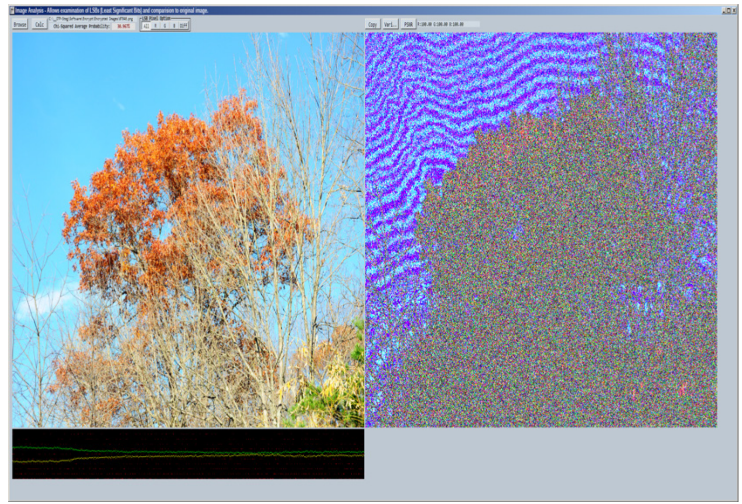


Figure 6. Image Analysis window, cover image on left, LSB analysis on the right

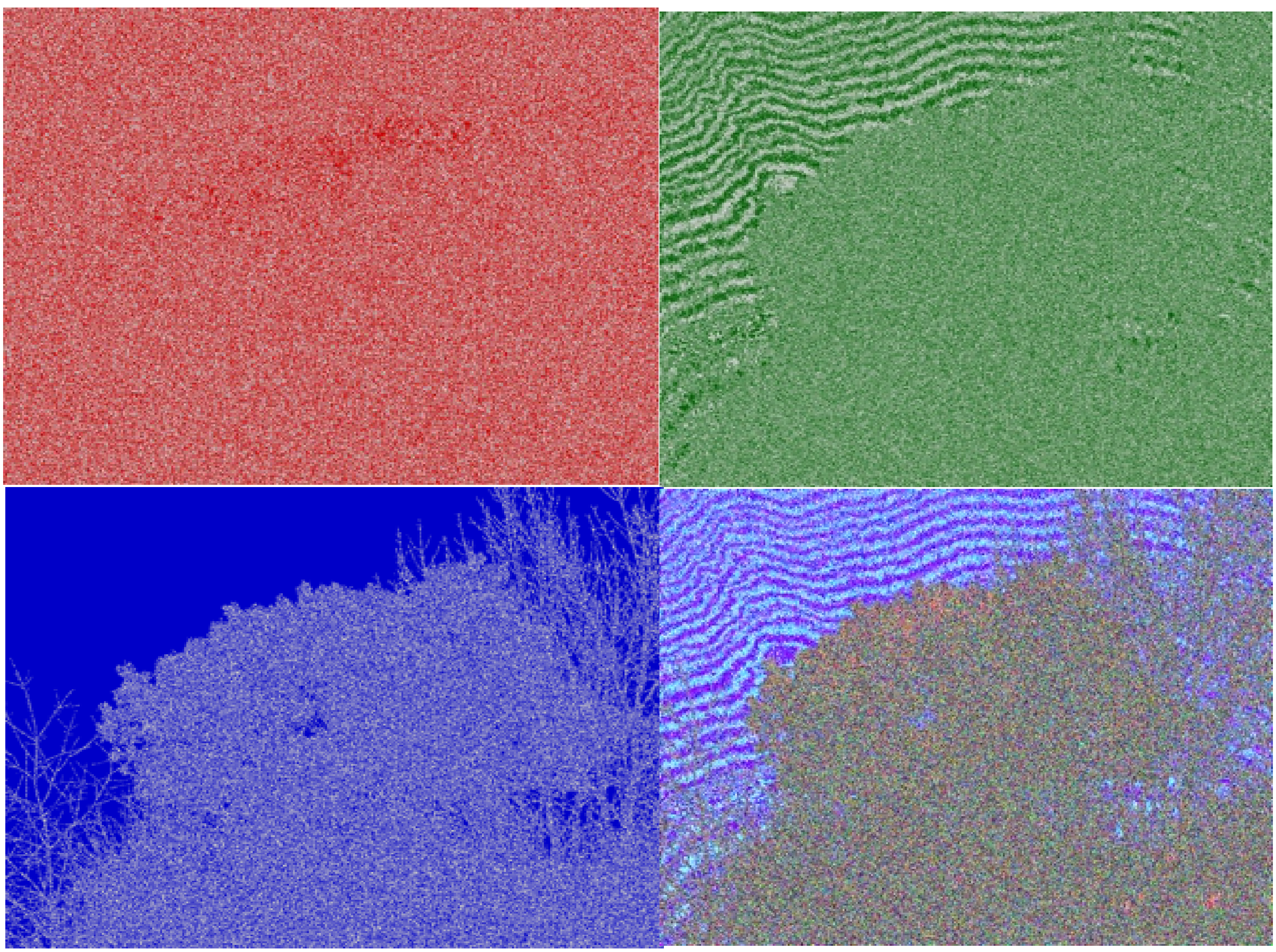


Figure 7. Analysis LSB color analysis graphics (Red, Green, Blue, All Colors).

Shown below are the variations in the red channel, the blue channel in Figure 9 shows the lack of variation in the sky area.

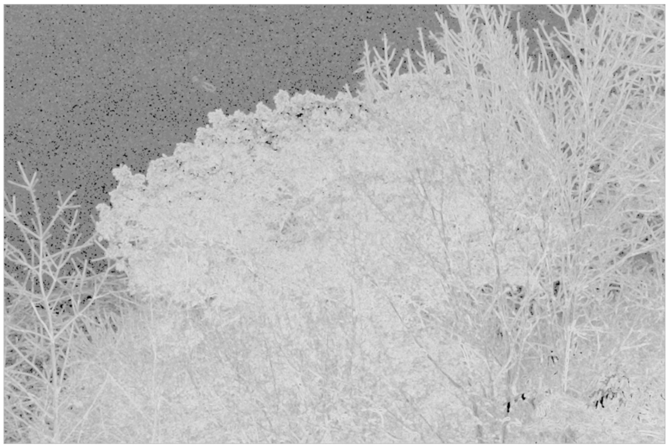
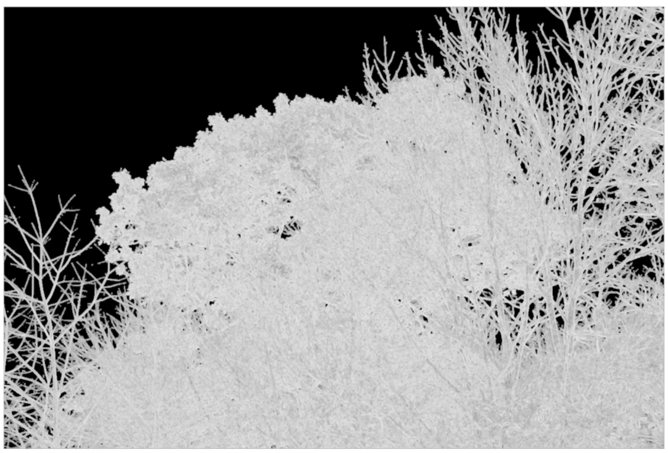


Figure 8. Red channel variation score (normalized to 0-255).



In Figure 10 below, the pixel least significant bit encodings are shown. Notice that in the area of the sky, only red pixels are encoded in the least significant bit. Other areas of the image vary between green and blue embedding depending on which color has the most variation in that pixel general area. Figure 11 shows a blowup of the pixel least significant coding in the area of the transition between the trees and the sky. Notice that the pixel encodings shift from primarily blue-green to the red color at this transition.

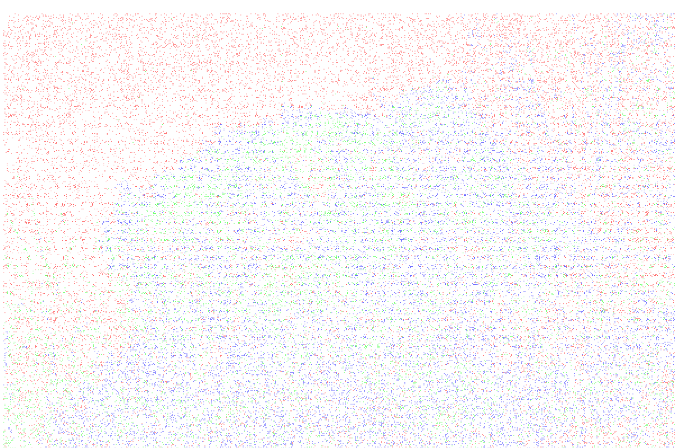
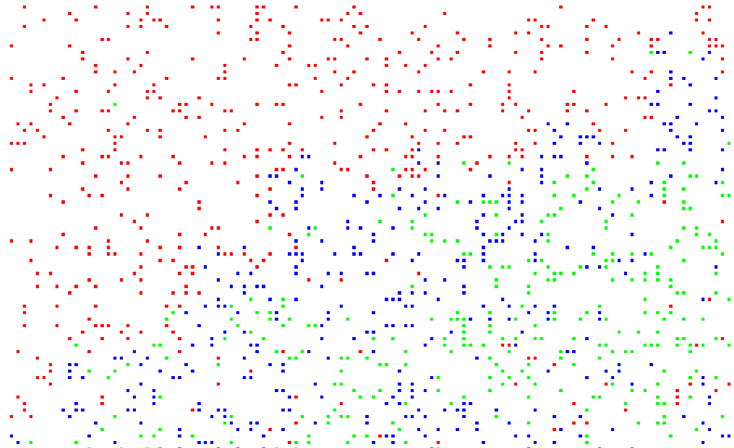


Figure 10. Pixel LSB modification encodings (Red, Green, or Blue).



### 4.7. Decryption

The decryption process largely reverses the encryption process using the decryption program. A SHA-256 hash value is computed from the decrypted message and compared to the hashed value contained within the payload file. If the two values matches, the hashed value is presented with a green colored background. If not, the background is reddish. A green value indicates to the receiver that the message has not been altered in any way since it was written.

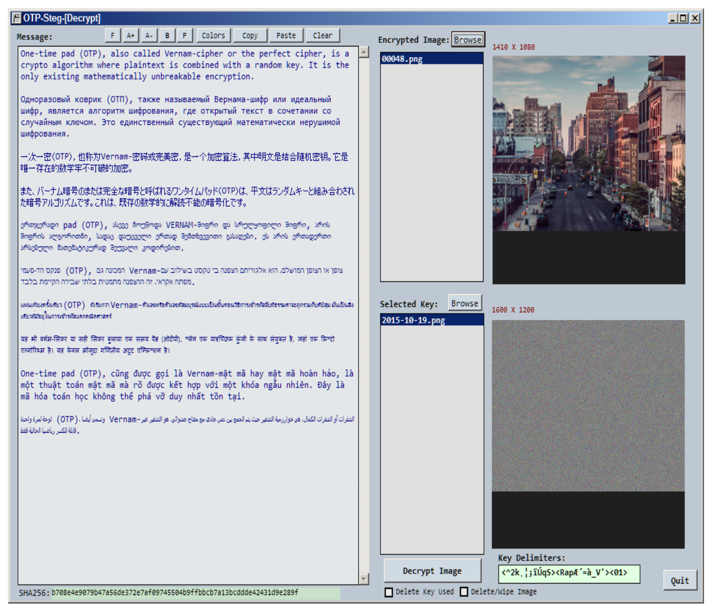



Figure 12. Decrypt/Extract executable program

### 5.1. Photo Selection

There are several general guidelines for photo image selection to increase security. Original photos taken with the user's own camera should be selected as cover images. This is to ensure that the duplicate of the original photo does not exist somewhere on the Internet for comparison. The photo should never have previously been encoded to JPEG to ensure full CMOS pixel sensor color variations throughout the image. As mentioned previously, once these criteria are satisfied, the user can evaluate an encodability score that is calculated by the encryption program that ranges from 0 to 100%.

### 5.2. *Encodability Score*

The user should choose in general images that score above 90% for encodability to enhance steganalysis security. The following is the weighting breakdown for the encodability score:

25% Overall PSNR (dB) variation score (0-100%) (more color variation = higher score)

25% Same colors next to each other (0-5%) (less same colors = higher score)

25% Black pixels (0-5%) (less black = higher score)

25% White pixels (0-5%) (less white = higher score)

100% - (100-90% = OK, 90-80% = Marginal, <80% = Unacceptable).

### 5.3. *Recommended Steganographic Practices*

Table 2. Recommended Steganographic Practices.

| No. | Practice | Description |
|---|---|---|
| 1 | Software Operation | Steganography software should be operated on a computer that is not connected to any network or the Internet. Files should be transferred using write only media such as DVD or CD, or less securely by USB drive. |
| 2 | Original Photos | Only original photos taken by the users high-quality camera should be considered as cover images. This is to ensure that the image does not have duplicates available on the Internet. Use RAW (original camera file format) images where possible. The software directly accepts all RAW image file types including Nikon, Canon, Sony, etc. |
| 3 | Software USB Loaded | The software should be run off of a USB drive plugged into the isolated computer. Further, USB drive containing software, keys, and cover images, should be located separately from the isolated computer in a safe and secure location. |
| 4 | Isolated Computers | The isolated computer used to run the software should be well secured and not networked in any way. The operating system should be directly installed from DVD, and antivirus and checks for malware should be regularly run to ensure there is no keystroke loggers, rootkits, or other security compromises installed. |
| 5 | "To" and "From" Keys | Both sender and receiver should have their own set of unique keys. Sender A to B, and B to A, each use their own one-way key series. This is to prevent key reuse. Each key must be used only one time, and one time only. Security using OTP depends on this precept. |
| 6 | Exchanging Keys | Key exchange should take place upon *physical meeting* using write only media such as DVD or CD. Key exchange must not take place over a network. Keys should be securely generated on isolated computers. Keys must be stored on removable USB drives separate from the isolated computer. |
| 7 | Deleting Files | All files including cover image files and key files should be forensically deleted and wiped once used. Forensic wiping utilities in the encryption and decryption programs can be used for this purpose. Wiping consists of randomly overwriting the previous file seven times with random data. |
| 8 | Sending Encrypted Files | Encrypted files should be sent as anonymously as possible. Direct email exchange should be avoided. A preferable alternative is to upload files periodically to gallery websites which have potentially thousands of viewers and downloaders daily. Identifying the specific receiver will be difficult in this situation. Each sender should upload to a different anonymous gallery. |
| 9 | Monitoring Windows Vulnerabilities | It should be known that just the act of plugging in a USB drive into a Windows computer creates a digital trail throughout the system registry. Installing software using a setup program also creates numerous records within the operating system registry. As a result, the software should be run off of a USB drive without running a separate install/setup utility. Windows must be isolated off of any network to ensure malware is not installed. |

| | | |
|---|---|---|
| 10 | Malware | Malware can cause a compromise in the steganography system at any time. A keystroke logger that is uploading typed messages is an instant fail. Users must be extremely cautious and knowledgeable about potential malware threats before using the software. In particular, any networked computer presents a point of vulnerability — the software and keys must never be used here. Only transfer of files previously encrypted on an isolated computer can be conducted over a network. |
| 11 | Usage Limitations | The biggest limitation is the human factor. Operational security must be observed that all times in addition to technical security. This means aggressive securing of the USB drive use for the software and keys, as well as limited knowledge by parties involved. People should be informed on a need to know basis only. |
| 13 | Encrypted Keys | For further security, keys can be encrypted for storage. As a result they will not be able to be used unless the user has knowledge of the encryption key. |
| 14 | Wipe Original Photos | Original photos must be deleted and erased from the camera, storage medium, and USB drive as soon as possible after they are used. |
| 15 | Wipe Used Keys | Keys must be deleted and erased as soon as they are used. |
| 16 | Internet Computer "Clean" | The computer connected to the Internet must be clean of viruses and malware or keystroke loggers. Special care must be taken in this area. |
| 17 | Camera Secured | The photo CMOS sensor output profile can be mapped to a particular individual camera. Photo sent on the Internet can be matched up to the users camera. As a result an effort should be made to keep the camera secure. |
| 18 | File Upload Galleries | File upload galleries should be selected for anonymity and high traffic volume. |
| 19 | Carefully Selected Cover Images | Cover images should be conforming to high encodability statistics and originality. Also they should be of subject matter that will not raise any suspicions. |
| 20 | Image File Format | Various image file formats can be chosen, leveraging the fact that steganalysis software will not run on many different types of image file types. Take advantage of other lossless file formats besides PNG and BMP such as TIF, J2K, EXR, WEBP, and JXR. |
| 21 | Must-Dos | 1: Keep software and keys in secure locations on USB drives. 2: Use software on isolated computer not connected to Internet. 3: Use keys and photos only once and be sure to erase files as soon as possible, especially original cover images and keys. |

### *5.4. Steganographic Communication Security State Level Estimation*

We envision certain levels of steganographic communication security levels that correspondents can use for planning, analysis, and security estimations. Thresholds can be established for protective measures using these security level guidelines.

Table 3. Notional Steganographic Security Levels.

| Security Level | Name | Description | Impact |
|---|---|---|---|
| 10 | Secure | Communication commencing securely. Operational security and human threat and insider threat must be strongly monitored and evaluated here. | None-success |
| 9 | Communication Suspected | Authorities suspect communication without knowledge of sender and receiver. | Low |
| 8 | Steg Statistically Detected | Positive steganography screening results indicating further investigation. | Moderate |

| | | | |
|---|---|---|---|
| 7 | Internet Computer Searched | If proper security measures recommended previously are followed, nothing should be derived. Duress codeword should be immediately used and communication ceased. | Moderate |
| 6 | Transmitted Files Discovered | If proper procedures are used, locating these files should not present much evidence. | Moderate |
| 5 | Software Computer Discovered | Traces of software use should be detectable in Windows registry. | High |
| 4 | Steg Known | Investigators conclude illicit communication has taken place, without acquiring USB drive(s). | High |
| | USB Drive discovered | User should make efforts to inform receiver communication is compromised. | Severe |
| 2 | Software Discovered/ Acquired | Knowledge of message text should be assumed at this point. | Severe |
| 1 | Key(s) acquired | Complete security compromise. | Severe |
| 0 | Suspect Detained | All communicating parties should make efforts to destroy any remaining evidence. | Failure |

Communicators should have a procedure in place to indicate ceasing of messages and also to destroy related software and keys systematically.

Steganographers should consider incorporating a duress codeword into their communication security protocol. The duress code word should be a predetermined word or phrase that indicates to the receiving party that communication security has been compromised. For example, capture by authorities may have created a situation where one party is succumbing to efforts to be "turned". The duress code word indicates such a situation and should be carefully chosen to arouse no suspicion should authorities have knowledge of its inclusion in a "trap" message.

### 5.5. *Software Availability*

The software is available as a free educational and research download to be used for digital forensics education and related projects. Please feel free to use the software for your own educational and research purposes. The software can be acquired here: http://199.175.52.196/OTP-Steg/.

## 6. Steganalysis Results

*StegExpose* is a Java based steganalysis tool heavily geared towards bulk analysis of lossless images. It is a steganalysis tool specialized in detecting LSB (least significant bit) steganography in lossless images such as PNG and BMP. It has a command line interface and is designed to analyze images in bulk while providing reporting capabilities and customization which is comprehensible for non-forensic experts. The *StegExpose* rating algorithm is derived from an intelligent and thoroughly tested combination of pre-existing pixel based steganalysis methods Two new fusion detectors, standard and fast fusion were derived from four well known steganalysis methods and successfully implemented in the tool. Standard fusion is more accurate than any of the component detectors it is derived from.

The following LSB steganalysis methods have been incorporated in *StegExpose*. RS analysis (Fridrich, Goljan, and Du 2001) detects randomly scattered LSB embedding in grayscale and color images by inspecting the differences in the number of regular and singular groups for the LSB and "shifted" LSB plane. Sample pair analysis (Dumitrescu, Wu, and Wang 2003) is based on a finite state machine whose states are selected multisets of sample pairs called trace multisets (Dumitrescu, Wu, and Wang 2003). The chi-square attack (Westfeld and Piltzmann 2000) is a statistical analysis of pairs of values (PoV's) exchanged during LSB embedding. PoV's are groups of binary values within a object's LSB's. Primary sets (Dumitrescu, Wu, and Memon 2002) is based on a statistical identity related to certain sets of pixels in an image. The difference histogram analysis (Zhang and Ping 2003) is a statistical attack on an image's histogram, measuring the correlation between the least significant and all other bit planes. Two new fusion detectors, standard and fast fusion, were derived from four well known steganalysis methods and successfully implemented in the tool. The standard fusion test is more accurate than any of the component detectors it is derived from.

*StegExpose* (the free open source download), was run on a batch of 27 image files that were encoded using the *OTP-Steg* software. Test specifications and results are shown below. None of the embedded files were detectable above the preset default threshold. Standard fusion was the test run which consists of all of the specific steganalysis tests mentioned above.

Table 4. StegExpose Steganalysis Test Specifications.

| **Test Spec** | **Description** |
|---|---|
| Embedded Text File: | U.S. Constitution; 52,782 Bytes Unicode (422,256 bits) |
| Images: | 27 Various landscape PNG photos, 1200?97 pixels (956,400 pixels) Nikon D90. |

| | |
|---|---|
| Uncompressed: | Approximate Bits per Pixel (bpp) 0.442 bpp |
| Compressed (zLib): | Approximate Bits per Pixel (bpp) 0.086 bpp |
| Alterations: | 1.445% LSBs altered, 4.335% of pixels |
| File Archive: | http://199.175.52.196/OTP-Steg/USConstitution/ |

Table 5. StegExpose Steganalysis Test Results using "Standard Fusion" test.

| File name | Above Stego Threshold? | Primary Sets | Chi Square | Sample Pairs | RS analysis | Fusion (mean) |
|---|---|---|---|---|---|---|
| 00247.png | FALSE | 0.023408176 | 0.00353364 | null | 0.02018579 | 0.01570926 |
| 02155.png | FALSE | 0.068625394 | 0.01936033 | null | 0.04494632 | 0.04431068 |
| 02664.png | FALSE | NaN | 5.03E-13 | null | 0.08658666 | 0.04329333 |
| 03090.png | FALSE | NaN | 0.0037011 | null | 0.23737088 | 0.12053603 |
| 03164.png | FALSE | 0.136200359 | 0 | null | 0.02282364 | 0.05300800 |
| 03504.png | FALSE | NaN | 0.00363950 | null | 0.19724031 | 0.10043991 |
| 03509.png | FALSE | 0.120022314 | 0.00140079 | null | 0.03595793 | 0.05246034 |
| 04031.png | FALSE | 0.004125309 | 3.57E-04 | null | 0.04380402 | 0.01609549 |
| 04095.png | FALSE | NaN | 0.00743453 | null | 0.09919615 | 0.05331534 |
| 04164.png | FALSE | NaN | 0.01840689 | null | 0.07674373 | 0.04757531 |
| 04378.png | FALSE | NaN | 4.83E-04 | null | 0.17958705 | 0.09003513 |
| 04479.png | FALSE | 0.047114348 | 0.00183215 | null | 0.06152043 | 0.03682231 |
| 04637.png | FALSE | NaN | 3.57E-04 | null | 0.09375770 | 0.04705742 |
| 05169.png | FALSE | 0.030743209 | 3.57E-04 | null | 0.03714153 | 0.02274730 |
| 05255.png | FALSE | NaN | 3.57E-04 | null | 0.11245120 | 0.05640417 |
| 05262.png | FALSE | 0.018022058 | 0.00206287 | null | 0.01099853 | 0.01036115 |
| 05777.png | FALSE | 0.017279631 | 6.59E-13 | null | 0.0070650 | 0.00811488 |
| 06202.png | FALSE | NaN | 4.25E-04 | null | 0.09364117 | 0.04703301 |
| 06672.png | FALSE | 0.06420808 | 0 | null | 0.06458346 | 0.04293051 |
| 07134.png | FALSE | 0.03542274 | 3.57E-04 | null | 0.017337435 | 0.01770577 |
| 07140.png | FALSE | NaN | 0.00142365 | null | 0.165817881 | 0.08362076 |
| 07946.png | FALSE | NaN | 1.02E-11 | null | 0.072127587 | 0.03606379 |
| 08145.png | FALSE | 0.033316358 | 2.77E-04 | null | 0.023061286 | 0.01888482 |
| 09061.png | FALSE | 0.014700003 | 0.00485040 | null | 0.025382546 | 0.01497765 |
| 09252.png | FALSE | 0.074362745 | 7.14E-04 | null | 0.01319539 | 0.02942414 |
| 09431.png | FALSE | 0.040878552 | 0.00328135 | null | 0.031193448 | 0.02511778 |
| 09988.png | FALSE | 0.062680713 | 3.54E-04 | null | 0.054694774 | 0.03924330 |

Test Results Summary: Zero (0%) steganalysis detections using the "Standard Fusion" detection algorithm in *StegExpose* software. *StegExpose* can be downloaded here: https://github.com/b3dk7/StegExpose.

## 7. Applications Regarding Crypto-Currency Cold-Storage and Transfer

Recently the global phenomenon of crypto-currencies has made headlines and changed the face of financial technology worldwide. The seminal block-chain algorithm has been applied to at least 1200 new crypto-currencies in addition to original, the Bitcoin. However, the future of this revolution is uncertain as security vulnerabilities and regulatory restrictions may make the future of ownership and transfer questionable.

To promote safe usage and encourage investment and speculation for the future, steganography applications such as the one discussed in this paper can be taken advantage of with robustly secure results. For example, Bitcoin public address and private key values can be securely encrypted and hidden in images uploaded to the cloud, transferred through e-mail, or stored locally. This allows applications in so-called cold storage scenarios as well as covert transfer of crypto-currency amounts when necessary.

Transfers using the standard Bitcoin network protocols are easily detectable, and blocked by firewalls and deep TCP/IP packet inspection. As a result, any activity in the clear over the network or protocol whatsoever can be used to flag the IP address and responsible parties--severely compromising one of the precepts of the crypto-currencies which is both anonymity and privacy. VPN traffic has been and is virtually entirely shut down in several entire nations, making this obvious alternative for network usage infeasible.

In the case of Bitcoin, TCP/IP ports 8332 and 8333 are two default ports used for Bitcoin peer-to-peer transfers and can be easily detected, inspected, and blocked, and the information derived used for further investigation into the origin and nature of the network traffic. As a result, it is highly desirable to have secondary means for engaging in transfers other than the standard protocol in situations where deep packet inspection or firewalls may be interfering with the transaction process. This is the case not only for Bitcoin but all of the other crypto currencies as well, now and most likely in the future.

One-Time-Pad Steganography offers this potential in a highly secure manner. Further, in the instance of cold storage with retrieval backup, several individuals can be entrusted with encrypted images and keys, and some of the information can be doubly or triply encrypted to ensure the agreement of several individuals to the release of crypto-currency funds. Encrypted images can be stored on the cloud in multiple locations, as well as locally on permanent storage media. Keys, however, must be stored with the person of the control authority. Additional customized protocols and policies should be designed in the future to address crypto-currency authentication and management; however, the steganography technology offers an ideal starting point for both hidden and theoretically unbreakable storage.

## 8. Conclusion

In this paper, we have presented a complete One-Time Pad encryption and steganography system including all software necessary to complete practical communication. We have compiled recommended best practices and identified potential security levels. Finally, we have tested the software using robust state-of-the-art steganalysis techniques and found the low payload threshold maintained in the software produces a high margin of communication security safety. No payload files were detected (0% detections), despite each file containing the entire content of the U.S. Constitution as embedded text.

## Conflict of Interest

The authors declare no conflict of interest regarding the publication of this paper.

## References


[1] R. J. Anderson, and F. A. P. Petitcolas, In the limits of steganography? IEEE Journal of Selected Areas in Communications, vol.16, no.4, pp.474-481, 1998.

[2] M. Bashardoust, G. B. Sulong, and P. Gerami, Enhanced LSB image steganography method by using knight tour algorithm, vignere encryption and LZW compression? International Journal of Computer Science Issues, vol.10, no.2, pp.221-227, 2013.

[3] Bhatacharya, I. Banerjee, and G. Sanyal, A survey of steganography and steganalysis techniques in image, text, audio and video cover carrier? Journal of Global Research in Computer Science, vol.2, no.4, pp.1-16, 2011.

[4] K. Chan, and L. M. Chang, Hiding data in images by simple LSB substitution? Pattern Recognition, vol.37, pp.469-474, 2004.

[5] Cheddad, J. Condell, K. Curran, and P.M. Kevitt, Digital image steganography: survey and analysis of current methods? Signal Processing, vol. 90, pp.727-752, 2010.

[6] S. M. Douiri, M. B. O. Medeni, S. Elbernoussi, and E. M. Souidi, A new steganographic method for gray scale image using graph coloring problem? Applied Mathematics & Information Sciences, vol.7, no.2, pp.521-527, 2013.

[7] Gangwar, and V. Srivastava, Improved RGB-LSB steganography using secret key? International Journal of Computer Trends and Technology, vol.4, no.2, pp.85-89, 2013.

[8] R. S. Gutta, Y. D. Chincholkar, and P. U. Lahane, Steganography for two and three LSBs using extended substitution algorithm? ICTAT Journal on Communication Technology, vol.4, no.1, pp.685-690, 2013.

[9] Gutub, M. Ankeer, M. Abu-Ghalioun, A. Shaheen, and A. Alvi, Pixel indicator high capacity technique for RGB image based steganography? in Proceedings of Fifth IEEE International Workshop on Signal Processing and its Applications, 2008, University of Sharjah, U.A.E.

[10] N. Hamid, A. Yahya, R.B. Ahmad, D. Nejim, and L. Kannon, Steganography in image files: a survey? Australian Journal of Basic and Applied Sciences, vol.7, no.1, pp.35-55, 2013.

[11] J. He, S. Tang, and T. Wu, In adaptive steganography based on depth-varying embedding? in Proceedings of 2008 Congress on Image and Signal Processing, 2008, pp.660-663.

[12] M. Hussain, and M. Hussain, A survey of image steganography techniques? International Journal of Advanced Science and Technology, vol. 54, pp.113-123, 2013.

[13] Y. K. Jain, and R. R. Ahirwal, A novel image steganography method with adaptive number of least significant bits modification based on private stego-keys? International Journal of Computer Science and Security, vol.4, no.1, pp.40-49, 2010.

[14] M. Juneja, and P.S. Sandhu, Designing of robust image steganography technique based on LSB insertion and encryption? in Proceedings of International Conference on Advances in Recent Technologies in Communication and Computing, 2009, pp.302-305.

[15] Kamaldeep, Image steganography techniques in spatial domain, their parameters and analytical techniques: a review article?, IJAIR, vol.2, no.5, pp.85-92, 2013.

[16] H. B. Kekre, A. A. Athawale, and P. N. Halarnkar, Increased capacity of information hiding in LSB's method for text in image? International Journal of Electrical, Computer and System Engineering, vol.2, no.4, pp.246-249, 2008.

[17] Y. K. Lee, G. Bell, S.Y. Huang, R.Z. Wang, and S.J. Shyu, In advanced least-significant-bit embedding scheme for steganographic encoding? LNCS, vol.5414, 2009, pp.349-360.

[18] Li, J. He, J. Huang, and Y.Q. Shi, A survey on image steganography and steganalysis? Journal of Information Hiding and Multimedia Signal processing, vol.2, no.2, pp.142-172, 2011.

[19] C. Lou, and C. H. Hu, LSB steganographic method based on reversible histogram transformation function for resisting statistical steganalysis? Information Sciences, vol.188, pp.346-358, 2012. Application of a large key cipher in image steganography by exploring the darkest and brightest pixels? International Journal of Computer Science and Communication, vol. 3, no.1, pp.49-53, 2012.

[20] R. S. Marcal, and P.R. Pereira, A steganographic method for digital images robust to RS steganalysis? LNCS, vol.3656, 2005, pp.1192-1199.

[21] Martin, G. Sapiro, and G. Seroussi, Is image steganography natural? IEEE Transactions on Image Processing, vol.14, no.12, pp.2040-2050, 2005.

[22] M. K. Meena, S. Kumar, and N. Gupta, Image steganography tool using adaptive encoding approach to maximize image hiding capacity? International Journal of Soft Computing and Engineering, vol.1, no.2, pp.7-11, 2011.

[23] Mishra, A. Gupta, and D. K. Vishwakarma, Proposal of a new steganography approach? in Proceedings of International Conference on Advances in Computing, Control, and Telecommunication Technologies, 2009, pp.175-178.

[24] H. Mathkour, G. M. R. Assassa, A. A. Muharib, and I. Kiady, A novel approach for hiding messages in images? in Proceedings of International Conference on Signal Acquisition and Processing, 2009, pp.89-93.

[25] H. Motameni, M. Norouzi, and A. Hatami, Labeling method in steganography? World Academy of Science, Engineering and Technology, vol. 24, pp.349-354, 2007. vol. 270, part II, 2012, pp.479-488.

[26] M. T. Parvez, and A. A. Gutub, RGB intensity based variable-bits image steganography? in Proceedings of IEEE Asia-pacific Services Computing Conference, 2008, pp.1322-1327. Gandharba Swain et al. / International Journal of Computer Science & Engineering Technology (IJCSET)

[27] P. S. Pharwaha, Secure data communication using moderate bit substitution for data hiding with three-layer security? IE(I) Journal-ET, vol.91, pp.45-50, 2010., International Journal of Security and Its Applications, vol.6, no.2, pp.1-12, 2012.

[28] G. Swain, and S. K. Lenka, RSB array based image steganography technique by exploring the four least significant bits? CCIS

[29] G. Swain, D. R. Kumar, A. Pradhan, and S. K. Lenka, A technique for secure communication using message dependent steganography? International Journal of Computer and Communication Technology, vol.2, no. 2- 4, pp.177-181, 2010.

[30] G. Swain, and S. K. Lenka, Steganography using the twelve-square substitution cipher and an index variable? in Proceedings of ICECT, 2011, vol.3, pp.84-88.

[31] G. Swain, and S. K. Lenka, A robust image steganography technique using dynamic embedding with two least significant bits? Advanced Materials Research, vols. 403-408, pp.835-841, 2012.

[32] G. Swain, and S. K. Lenka, A dynamic approach to image steganography using the three least significant bits and extended hill cipher? Advanced Materials Research, vols. 403-408 pp.842-849, 2012.

[33] G. Swain, and S. K. Lenka, A technique for secret communication by using a new block cipher with dynamic steganography"

[34] G. Swain, and S. K. Lenka, A hybrid approach to steganography- embedding at darkest and brightest pixels?, in Proceedings of International Conference on Communication and Computational Intelligence, 2010, pp.529-534.

[35] M. A. B. Younes, and A. Jantan, A new steganography approach for image encryption exchange by using least significant bit insertion, International Journal of Computer Science and Network Security, vol.8, no.6, pp.247-254, 2008.

[36] H. J. Zhang, and H. J. Tang, "A novel image steganography algorithm against statistical analysis", in Proceedings of Sixth International Conference on Machine Learning and Cybernetics, 2007, pp.3884-3888.