Jonathon Meney

Professor Cezar Campeanu

Cryptography and Codes

8 December 2023

Steganography, Encryption in Plain Sight

Steganography is an incredibly unique form of cryptography. Its methods involve embedding hidden messages in simple plain files. Examples include hiding a secret message in an audio file, a video, or most commonly hiding images inside other images. Steganography can also be used for malicious intent by encrypting malware into something as simple as an image. This paper will examine the three main types of steganography, along with the mathematics behind the embedding process will also be observed in further detail. Lastly, we'll compare steganography to other methods of encryption, discuss its security, and look at some applications of steganography, as well as some examples.

1. Types of Steganography

The three main types of steganography include pure, public key, and private key. Pure steganography is the simplest form of steganography requiring no exchange of information such as keys (AL-Ani et al.). A pure steganography system is defined as a quadruple (C, M, E, D). The first element, C, is the set of possible covers (the medium that data is to be embedded in, image, video, audio, etc.) for M to be embedded into with E. The element M is the set of possible messages that we can embed into some cover, and E is the embedding function to do so. Also note, in pure steganography methods, the size of the message must be less than or equal to the size of the cover. If the message was larger than the cover, it would result in a block appearing as random noise being appended to the bottom of the cover, therefore defeating the purpose of trying to hide a message making it partially visible. Lastly, in pure steganography, we have the element D which is the extraction function to retrieve the original message from the cover.

The second type of steganography, secret key, involves a single key shared between transmitter and receiver. A secret key steganography system consists of five elements (C, M, K, E_K, D_K). As like pure steganography, elements C, M, E_K, and D_K, are the set of covers, the set messages, the embedding function, and the extraction function, respectively. The new fifth element is the set of possible keys K, when embedding with some key k, using the same k in the extraction function will result in the original message back (AL-Ani et al.). Typically the secret key used is important only to the embedding function but the message could also be encrypted in some encryption scheme using the same key.

The third and final type of steganography is public key, this functions like a typical encryption system. It consists of six elements C, M, KU, KR, E_{KU}, and D_{KR}. Once again like pure, and secret key steganography, C and M are the set of covers and the set of messages respectively. KU is the public key of the system, and KR is the private key of the system, and E_{KU}, and D_{KR} are the embedding and extraction functions. Public key steganography systems involve encrypting the message first before embedding it in some way (AL-Ani et al.). What this causes is the image to be dispersed more evenly across the image as noise and is much harder to detect, and the encryption adds a level of security.

2. Steganography Methodology

In addition to the three main forms of steganography there is also a wide variety of embedding methods that can be used to hide a message. There are five main methods ranging from most common to difficult to implement. They include substitution systems, the transform domain technique, the spread spectrum technique, the distortion technique, and the cover generating method (AL-Ani et al.). The most common method is substitution systems which can be broken down into several variations.

The most common substitution system is, Least Significant Bit. To embed a message inside of a cover, the bits of the message are substituted into the least significant bits of each byte of the cover. The number of least significant bits used varies but is typically between one and 3 bits. We are able to use these bits as changing the value of them will result in no noticeable change in the image. Since a color value ranges from 0 - 255 a change of between 1 and 8 can’t be seen. The same is true for video and audio, causing no visual or auditory difference between the original cover and the embedded cover (Zakaria et al.). To extract the message the receiver simply extracts the least significant bits and concatenates them back together to generate the message.

Another substitution system is Random Interval. This substitution system is also an example of secret key steganography. To embed a message using Random Interval, a sequence of random numbers is generated from some seed. Bits of the message are then embedded based on this sequence (Zakaria et al.). For example, if the sequence generated was 5, 10, 4, 3, then the first bit would be embedded at position 5, then the next 10 positions later, the next 4 positions later, and so on. The receiver can extract the original message by generating the random interval sequence again based on the seed the sender used, and extracting the bits at those intervals. The bits can then be concatenated back together to see the original message.

Another relative of Random Interval substitution is Pseudorandom Permutation. This method involves generating a random sequence and assigning the sequence to the bits of the secret message. The message bits are then embedded in ascending or descending order based on this sequence, and the same number of positions into the cover (Zakaria et al.). For example, if we generate a sequence 5, 3, 17, and a message, 011, we assign 5 to 0, 3 to 1 and 17 to 1. We then embed 1 at position 3, 0 at position 5, and 1 at position 17. The receiver can then extract the message by using the seed to generate the sequence, retrieve the bits at those positions and then rearrange them to be in the matching order to the random sequence.

Image downgrading is another simple embedding method that involves only the most significant bits of the message and the least significant bits of the cover. This method is also special as it can only be done when both the cover and message are images. The embedding process involves taking the four most significant bits of each byte of the message image, and substituting them into the four least significant bits of each byte of the cover image. This method will slightly distort the cover image and there will be a small loss in the message image. However, the loss in the secret image will not be large enough to make the secret image completely unreadable, and will be very similar to the human eye. To extract the secret image, the receiver can extract the four least significant bits of each byte and right zero pad them with four zeros, then concatenate each byte together to retrieve the secret image (Zakaria et al.).

The last kind of substitution system is Cover Regions & Parity Bits. This method is the most complex substitution system requiring a multi step process. To embed a message, we first find the length of a region by dividing the length of the cover by the length of the message. Then we break the cover into the regions of that size. We then choose some bit of the region to act as a parity bit. A bit from the message is then given to each region and the parity bit is either flipped or kept the same based on what the new parity of the region would become to hide the message (Zakaria et al.). Then to retrieve the message, we take the original cover and subtract (bitwise) the cover with an embedded message.

The second kind of embedding process is known as Transform Domain. The first technique of this form is using the Discrete Cosine Transform. When using Discrete Cosine Transform we also work in the color space of YCbCr, and only care about the Y part which is luminance. An image is first broken down into groups of eight by eight pixels. The Discrete Cosine Transform of two dimensions is then applied to the Y value of each pixel, the formula for this is $X_k = \sum_{n=0}^{N-1} x_n \cos\left[\frac{\pi}{N}\left(n+\frac{1}{2}\right)k\right] \qquad \text{for } k = 0, \ \dots \ N-1 .$. The coefficients generated by the Discrete Cosine Transform of two dimensions are then divided by some quantization table that has been chosen for use, then in the least significant bits of these coefficients the secret message is embedded. Then the image is compressed. To retrieve the hidden message, it is first decompressed, and the inverse of the Discrete Cosine Transform of two dimensions is applied given by $X_k = \frac{1}{2}x_0 + \sum_{n=1}^{N-1} x_n \cos\left[\frac{\pi}{N}\left(k+\frac{1}{2}\right)n\right] \qquad \text{for } k = 0, \ \dots \ N-1 .$. Then the least significant bits can be extracted and put back together to retrieve the secret message (Vaishali et al.).

The other form of Transform Domain, uses Discrete Wavelet Transform and is not very commonly used at all. To embed a message into some cover image, the image is first broken down into four sub band frequencies and the message is embedded in the least significant bits of one of the four subbands. The image is then reconstituted from these four sub bands. To extract the message the image can simply be broken into its four sub bands again and extracted from the least significant bits of the band it was embedded in (Vaishali et al.). Another form of the Discrete Wavelet Transform method also exists where we decompose each pixel of the cover into its RGB values and embed a secret message in only one color space (Vaishali et al.).

Another technique of steganography is known as Spread Spectrum. This technique first involves encoding the message by adding redundant bits through some error correcting code. The encoded message is then padded to make it the same size as the cover. This is done to make it harder to detect an embedded message. A seed is then used to generate a pseudorandom sequence, and this sequence is used to modulate the message into random noise (Brundick et al.). The modulated message of noise is then embedded in the least significant bits of the cover. To extract the message the steps are repeated in reverse using the same seed to regenerate the pseudorandom sequence for modulating the message into noise.

Distortion technique is another steganography method most commonly used when embedding messages into a piece of text as the cover. To embed a message using this method it is similar to the Discrete Wavelet Transform method. The cover is first broken into its four sub bands and then the message is embedded across all subbands according to some distortion function (Holub et al.). This method is not widely implemented and distortion functions are not commonly found as well. To extract the hidden message the original cover and the cover with embedded message must both be looked at and their difference must be taken. This difference will result in the original message.

The last technique used in steganography is the Cover Generating method. This method is the simplest to understand, yet the most complex to implement. In this system we begin only with a message to embed in some cover we don't yet have. We generate this cover based off of the message to be encoded (Chen et al.). Currently this method is only achievable by hiding some message into a piece of text. A good example of this is SpamMimic, a website online, that will generate a long message from some message to also be hidden inside of this generated text. With the advent of AI this method may prove easier to implement over time as AI machines are capable of generating images, audio, and even video.

3. Security of Steganography

The security of steganography varies from technique to technique. It can also be as secure as you want it to be, by first encrypting the secret message. Doing so will make it harder to detect as well as if it is retrieved from the cover it will still need to be decrypted. Encrypting the message makes it harder to detect as it will appear more as noise when analyzing the image. Choosing a cover that is very close in size will also make it harder to detect as when analyzing the least significant bits, if encoded with some substitution system, will result in an image of just noise. If the message is significantly smaller than the cover, and encoded the same way it will appear, when examining the least significant bits, as a block of noise that abruptly cuts off before the end of the image. Transform domain methods are the most secure as the changes in the least significant bits of a cover are very hard to notice when analyzing them (AL-Ani et al.). A notable downside of steganography is malware can be embedded into an image. When a user opens this image, if embedded in such a way, the message will be executed by the image loader. To the common computer user, a malware embedded cover is indistinguishable to them, which makes this attack on a user very easy to execute.

4. Comparisons to Data Encryption

Steganography is a very different form of encryption. Steganography is more about information concealment than data encryption. The reason this is possible, is if someone intercepted a cover with an embedded message, it is not immediately known that there is some hidden message inside of what was intercepted. Data is also not always encrypted in steganography, though as mentioned before, encrypting the data before embedding it makes it much harder to detect. Lastly, Random Interval Substitution and Pseudorandom Permutation Substitution are very similar to a Diffie Hellman key exchange.

5. Applications

The number of applications of steganography is quite small. The most common application is in printers. When a printer prints a document, it will embed data small dots into the printed document that are only visible under incredibly close inspection or microscope. This embedded data holds information about the printer, commonly the model and serial number of the printer, and the timestamp of the print (Gibson). Another common application of steganography is digital watermarking. A watermark can be embedded throughout the entire image over and over again and it will not be visible to the human eye. This allows the owner of the image to have concrete proof that they own this image. Since the watermark is repeated through the whole image it will not be lost if the image is cropped, and if embedded correctly will not be lost on compression. Another good example is money which contains a series of different watermarks embedded into bills to detect counterfeit cash. Once again as mentioned before malware is another application, though not particularly safe or useful.

6. Cicada 3301

Lastly, Cicada 3301 was a cryptic internet puzzle that was revealed in 2012, containing a series of complex challenges and cryptographic puzzles attached to it. One intriguing aspect of Cicada 3301's puzzles was the incorporation of steganography, a technique used to conceal messages within seemingly ordinary content. Participants discovered hidden messages within images, audio files, and other media by meticulously analyzing the data. Steganography allowed Cicada 3301 to embed clues in plain sight, requiring a keen eye and a deep understanding of various encoding methods to unveil the concealed information. This use of steganography added an extra layer of complexity to the puzzle, contributing to the intrigue surrounding Cicada 3301 and attracting a large community. To this day the community is still trying to solve the Liber Primus, a 57 page book written in a runic language that was uncovered in 2014. Cicada 3301 has also been radio silent since 2017 ("Uncovering Cicada Wiki.").

7. Conclusion

To conclude, steganography comes in many types, and has many different embedding schemes. The security of steganography varies heavily based on the methods used to embed a secret message, and is vastly different compared to regular encryption of data. Steganography does not have many applications, however, in the niche use cases where it is used, we can see it to be very useful. Cicada 3301 is just one famous example where steganography was used.

Works Cited

AL-Ani, Zaidoon Kh., et al. “Overview: Main Fundamentals for Steganography .” *Journal Of Computing*, vol. 2, no. 3, Mar. 2010, pp. 158–165, https://arxiv.org/ftp/arxiv/papers/1003/1003.4086.pdf.

Brundick, Frederick S, and Lisa M Marvel. “Implementation of Spread Spectrum Image Steganography.” *Army Research Laboratory*, Mar. 2001, apps.dtic.mil/sti/pdfs/ADA389095.pdf.

Chen, Kejiang, et al. “Cover Reproducible Steganography via Deep Generative Models.” *IEEE TRANSACTIONS ON DEPENDABLE SECURE COMPUTING* , 26 Oct. 2022, https://doi.org/https://doi.org/10.48550/arXiv.2210.14632.

Gibson, Ryan. “Steganography: Hiding Data In Plain Sight.” University of North Carolina. https://www.cs.unc.edu/~lin/COMP089H/LEC/steganography.pdf. Accessed 26 Nov. 2023.

Holub, Vojtěch, et al. “Universal distortion function for steganography in an arbitrary domain.” *EURASIP Journal on Information Security*, vol. 2014, no. 1, 3 Jan. 2014, https://doi.org/10.1186/1687-417x-2014-1.

P, Vaishali, and Pradyumna Bhat. “Transform Domain Techniques for Image Steganography.” *INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN ELECTRICAL, ELECTRONICS, INSTRUMENTATION AND CONTROL ENGINEERING*, 3`, no. 1, Apr. 2015, pp. 65–68, https://ijireeice.com/wp-content/uploads/2015/05/T1_131.pdf.

“Uncovering Cicada Wiki.” *Fandom*, uncovering-cicada.fandom.com/wiki/Uncovering_Cicada_Wiki. Accessed 26 Nov. 2023.

Zakaria, Abdul Alif, et al. “Analysis of Steganography Substitution System Methods Using New Testing Techniques Proposed for Strict Avalanche Criterion.” *International Journal of Cryptology Research*, vol. 1, no. 5, 2015, pp. 61–76, https://www.cybersecurity.my/data/content_files/53/1648.pdf.