Stegware Destruction Using Showering Methods

Saurabh Choudhury, Amritha P.P, M. Sethumadhavan

Abstract: ***Stegware is a steganographic technique used to hide malware in images. Once the malware embedded image is loaded on the victim side, an attacker can execute these malwares remotely and potentially steal information from the end user. Because of its nature, Stegware cannot be detected by usual methods, but it can be destroyed before it executes. In this paper we introduce showering mechanism which will destroy the malware and only innocent or cleaned image will be received by the end user. Experimental results show that Stegware is even vulnerable to histogram attack. This destruction method can completely block the execution of Stegware and preserves the quality of image which is measured using PSNR.***

Index Terms: Stegware, malware, steganography.

I. INTRODUCTION

The one challenge that all cyber-attackers must overcome first is how to move their malicious tool from one place to another i.e. to a victim's system undetected. Post attack analysis of many of the recent attacks has shown that attackers are turning to steganography to move their payload around. What started as a means to hide secret information within seemingly unsuspicious files has evolved into a way for attackers to embed their malwares. This embedding of malware into files is known as Stegware.

Unlike cryptography, steganography has proven itself to be a versatile method of transporting information without being detected by anyone who might be monitoring the channel, it has become the go to method for attackers to use it as a delivery system. Steganography hides the existence of anything in contrast to cryptography which only makes the information unreadable.

Stegware can used to both infiltrate and exfiltrate data from a victim's PC. Moreover, with enough sophistication it can also be turned into a Command and Control server [1].

The advantage of Stegware in short is its use of steganography which hides the attack from normal prying eyes. Since, Stegware in its basic form is still steganography, it can still be detected or destroys using anti steganography techniques. Moving forward, we show some of these techniques.

II. RELATED WORKS

Since stegware basically uses steganography methods, we have a lot of resource materials to start with.

A. Soria-Lorente and S. Berres in their paper demonstrate a novel steganographic method which appears to be highly resistant against the Chi-square attack. Based on the Joint Photographic Expert Group compression standard and an Entropy Thresholding technique. The steganographic algorithm uses a pair of keys, one private and the other public to generate a binary sequence of pseudorandom numbers, which indicates the position as to where the secret message needs to be embedded [2].

G.Umamaheswari and Dr.C.P.Sumathi in their work a method where the cover image remained unchanged. Here, the LSB pixel values that match with the secret message bit values are saved to a separate position file which is further protected by a secret key [3].

Aos.A.Z., et al., in their work studied the different types of steganography systems, design and implementation of steganography system which embeds information in an .EXE files. After which they propose a novel system for bypassing antivirus detection [4].

In [5] authors used double stegging methods to destroy the content from audios. There was not much work done on destroying stego content from audios. Double stegging method showed that stego content was destroyed while the quality of audio was preserved.

Wojciech Mazurczyk and Luca Caviglione in their paper discussed how information hiding techniques are being increasingly utilized by current malware to hide all trails of its existence. They highlight this new trend by reviewing the most notable examples of malicious software that shows this capability [6].

In [7] authors have used image processing operations to destroy the stego content from images. They showed that their method could destroy 80 % of the stego and maintained the quality of the image.

III. PROPOSED WORK

Here we propose two showering method to destroy stegware and was subjected to RS steganalysis [6] to verify the performance of the showering methods. At first using various embedding techniques we embed a malware in a cover image. We create a pool of images that contains of mixture of normal images and several harmful images. Once an image has been flagged as suspicious under a steganalysis method we proceed accordingly. For example, say an image has been flagged as suspicious by the LSB embedding detector we will either flip the LSBs or embed some arbitrary data into the LSBs to destroy the embedded data [8].

We have used two showering methods for destroying the stegware. They are –

Revised Manuscript Received on April 12, 2019.

Saurabh Choudhury, TIFAC-CORE in Cyber Security, Amrita School of Engineering, Coimbatore, Amrita Vishwa Vidyapeetham, India.

Amritha P.P, TIFAC-CORE in Cyber Security, Amrita School of Engineering, Coimbatore, Amrita Vishwa Vidyapeetham, India.

M. Sethumadhavan, TIFAC-CORE in Cyber Security, Amrita School of Engineering, Coimbatore, Amrita Vishwa Vidyapeetham, India.

A. LSB Bit Flipping

This is a simple method in which we check the value of pixels or the RGB component and flip its LSB. We do this for all the pixels of the image. This method assures that we overwrite any data that might have been embedded using any type of LSB embedding.

B. Double Stegging

As our aim is to destroy the embedded data and not extract and analyze it, the complexity of our work decreases. We re-embed random data into the LSBs of the harmful image. Here, we use the same method to embed data into the image two times. The second time is considered as double stegging. Once this step is done, we try to extract the information which is the malware and try to execute it.

Steganalysis techniques used for analysis –

A. Histogram analysis

This is a rather simple and very effective analysis technique for detection of steganography. Here, we draw a histogram to depict the overall pixel distribution. For a RGB image we draw three histograms for each of the planes.

B. StegExpose

StegExpose is a steganalysis tool developed by Benedikt Boehm specialized in detecting LSB steganography in images. It has a command line interface and is designed to analyze images in bulk while providing reporting. StegExpose rating algorithm is derived from an intelligent and thoroughly tested combination of pre-existing pixel based steganalysis methods.

C. Regular Singular (RS) steganalysis

It has been observed that the smoothness of an image is influenced by randomizing the LSB of the images. It has been found that LSB plane is not completely random rather that it is related to every other bit plane. Message embedding in the LSB plane can be considered to be the same as randomizing the LSB plane. The one advantage that this analysis has over others is that not only determines the existence of a message but also gives us its length [6].

IV. EXPERIMENTAL RESULTS

To conduct the experiments, we used 10 images from the Caltech dataset. These images were then embedded with varying payload (100%, 50%, 30%, 5%) of image size to create a diverse dataset. All stego images were subjected to RS steganalysis. There values were recorded for future comparison with the images after showering was done.

We created the stegware by using Stegosuit, InvokePSI [10] and also by writing our own code in JAVA and MATLAB. InvokePSI allowed us to embed a PowerShell script in it and execute it with a one liner. It also has the capability to execute script when the infected image is open on a web browser on a victim's system. After the creation of the stegware, we used StegExpose to determine the nature of the images and RS steganalysis to determine the length of the message. Once this was done, we proceeded with the destruction of the embedded data using showering methods. After this process we again applied StegExpose and RS Steganalysis to evaluate the performance of our methods.

Fig. 1. Normal Image

Fig. 2. Image with hidden data made using Stegosuite

Fig. 2. Shows the image created using StegoSuite and we found that the quality of the image is 92.165 dB.

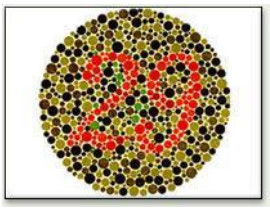

Fig. 3. Normal Image

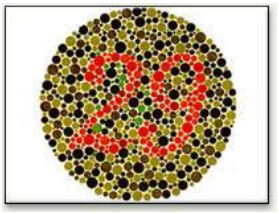

Fig. 4. Image hidden with data using InvokePSI

Fig. 4. shows the image created using InvokePSI and we found that the quality of the image is 31.625 dB.

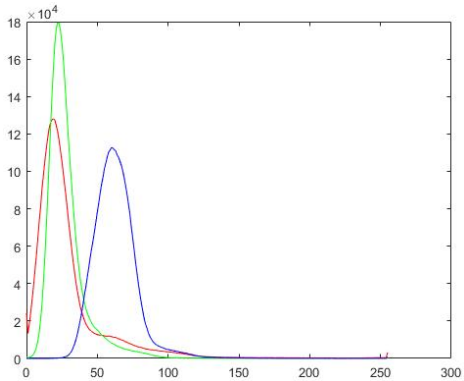

Fig. 5. RGB Histogram of normal image

Fig. 5. gives us the histogram for a normal image. We take this diagram as a standard to compare with the histogram of the infected images.

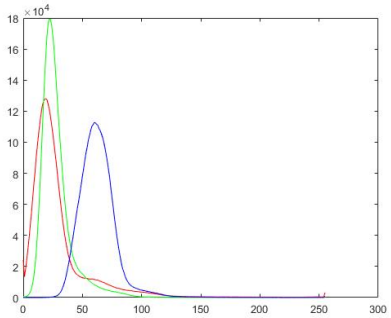

Fig. 6. RGB Histogram after embedding using StegoSuite

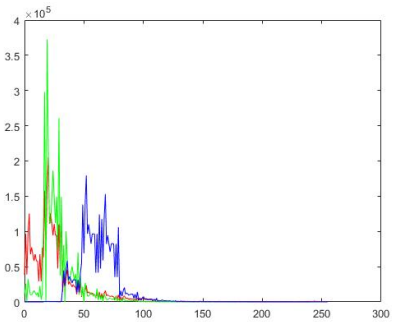

Fig. 7. RGB Histogram after embedding using InvokePSI

In Fig. 7. we see the histogram of the stegware created by using InvokePSI. When compared with Fig. 5. we can easily deduce that something is wrong with the image in Fig. 7.

Table I and II lists the PSNR values of four images before and after using double stegging and LSB flipping. Results show that the image quality is preserved above 30 dB after showering. Table III records the percentage of payload embedded in an image along with their RS Steganalysis percentage. These are images which might be sent to a victim. Table IV lists the RS Steganalysis values after applying the showering methods. It was observed that the RS steganalysis failed to detect the presence of the hidden malware data

Table I. Image Quality Measured Using PSNR

Before Showering (dB)		After Showering Using Double Stegging (dB)	
StegoSuite	InvokePSI	StegoSuite	InvokePSI
92.1585	31.125	92.2199	31.667
92.9308	31.225	92.2199	31.896
92.6079	31.889	92.1099	31.885
92.4878	31.855	92.1099	31.855

Table II. Image Quality Measured Using PSNR

Before Showering (dB)		After Showering Using LSB Flipping (dB)	
StegoSuite	InvokePSI	StegoSuite	InvokePSI
92.1585	31.125	92.7857	31.456
92.9308	31.225	92.2657	31.159
92.6079	31.889	92.1968	31.753
92.4878	31.855	92.1523	31.458

Table III. RS Steganalysis of STEGO Images

Payload %	*RS Steganalysis %*
100	66.32
50	32.45
30	26.23
5	17.18

Table IV. RS Steganalysis Of showered Images

Payload %	*Showering Method*	*RS Steganalysis %*
100	LSB Embedding	<1
100	Double Steg	<5
50	LSB Embedding	<1
50	Double Steg	<5
30	LSB Embedding	<1
30	Double Steg	<3
5	LSB Embedding	<1
5	Double Steg	<3

Even though the result of StegExpose classified the showered images as harmful, we were not able to execute the stegware.

After the bit – flipping, we extracted the malware that was embedded in it and tried to execute it. We noticed thet the malware failed to execute. In case of double stegging, we were able to execute the script that we embedded and thereby destroying the malicious script.

Moving on from static images we also considered how attackers might use animated GIFs as stegware. First we went over the structure of a GIF file. [11] nicely breaks down how a GIF looks like internally. After going over it, it was found that all GIFs have a comment section where data could be stored in plain text. An attacker might include the binary of a malicious program here and send it to a unsuspecting victim. On the victim side, the binary can be easily executed after extracting it. The attacker does not even need a very sophisticated extraction method, he can just open the GIF in a text editor and get the required data. Destroying this type of stegware is as simple as purging the comment section.

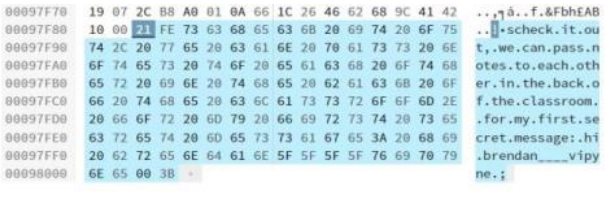

Fig. 8. Comment section of the GIF showing hidden data viewed in a hex editor

Next we tried to determine how videos could be used for the same purpose. Videos are a series a frames which are basically static images. We used the openly available ffmpeg suit of tools which lets us handle video, audio and various other forms of multimedia [12].

Using ffmpeg we first extracted the frames from 4 second video randomly chosen from YouTube. We can choose to store the frames losslessly or in lossless format. We can also opt to either extract all the frames or frames after an interval of time. This gives us our set of static images to work with. Similarly, as we had created stegware in static images, we used the same procedure to create stegware from this frame. After this we recreated the video the using ffmpeg using various codecs. Some of the codecs that were used are huffyuv, ffv1, libx264, libx265 and libvpx-vp9. The last one being a lossless codec. After this we once again extracted the frames and compared their MD5 hashes to check if the data was preserved or not. Using a lossless codec, the data was preserved but other codecs did not guarantee this. So re-creation of the video using different types of codecs with different configurations also work as a showering method. Moreover, we can apply the previously mentioned showering methods to get similar results for the stegware destruction

V. CONCLUSION

We conclude that showering methods were able to stop the stegware from executing its hidden malware. We found that RS Steganalysis and Histogram analysis which normally work for detecting hidden messages also work for hidden malwares in images. Our destruction method was also able to preserve the quality of image with PSNR above 30 dB.

REFERENCES

1. Wiseman, S. (2017). Stegware – Using Steganography for Malicious Purposes. [online] Research Gate. Available at: https://www.researchgate.net/publication/321623657_Stegware_-_Using_Steganography_for_Malicious_Purposes [Accessed 18 Mar. 2018].
2. Soria Lorente and S. Berres, “A secure steganographic algorithm based on frequency domain for the transmission of hidden information," vol. 2017, pp. 1-14, 01 2017.
3. Umamaheswari, G & Sumathi, C.P., “A New Steganographic Technique Matching the Secret Message and Cover image Binary Value”, 2017.
4. A. Z., A. W. Naji, S. Hameed, F. Othman, and B. B. Zaidan, “Approved undetectable-antivirus steganography for multimedia information in pe-file,” vol. 0, pp. 437-441, 2009.
5. Srinivas, T. M., and P. P. Amritha. “Real Time Audio Steganographic Countermeasure”. Data Engineering and Intelligent Computing. Springer, (2018), 293-300.
6. W. Mazurczyk and L. Caviglione, “Information hiding as a challenge for malware detection,” vol. 13, pp. 89-93, 2015.
7. Amritha, P. P., M. Sethumadhavan, and R. Krishnan. “On the Removal of Steganographic Content from Images”. Defence Science Journal (2016). 66.
8. Archana O. Vyas and Dr. Sanjay V. Dudul, “A Technique for Steganalysis of Object Oriented LSB Steganography”. International Journal of Advanced Research in Computer Science, Vol 6, No. 8, 2015
9. Hsien-Wen Tseng and Hui-Shih Leng, “A Steganographic Method Based on Pixel-Value Differencing and the Perfect Square Number,” Journal of Applied Mathematics, 2013.
10. GitHub. (2018). peewpw/Invoke-PSImage. [online] Available at: https://github.com/peewpw/Invoke-PSImage [Accessed 18 Jul. 2018].
11. Flickinger, M. (2005). What's in A GIF - Bits and Bytes. [online] Giflib.sourceforge.net. Available at: http://giflib.sourceforge.net/whatsinagif/bits_and_bytes.html [Accessed 21 Jul. 2018]
12. Ffmpeg.org. (2018). FFmpeg. [online] Available at: https://www.ffmpeg.org/ [Accessed 21 Oct. 2018]

AUTHORS PROFILE

Saurabh Choudhury is currently pursuing his M. Tech (Cyber Security) from Amrita Vishwa Vidyapeetham. His current research interests include: Steganography.

Ms Amritha P.P. received her M. Tech (Cyber Security) from Amrita Vishwa Vidyapeetham, currently pursuing her PhD at Amrita Vishwa Vidyapeetham. Her current research interests include: Steganography and code obfuscation.

Dr M. Sethumadhavan received his PhD (Number Theory) from Calicut Regional Engineering College. Currently, he is working as a Professor in the Centre for Cyber Security, Amrita Vishwa Vidyapeetham, Coimbatore. His current research interests include: Cryptography and Boolean functions.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication