Indonesian Journal of Electrical Engineering and Computer Science
Vol. 12, No. 2, November 2018, pp. 716~721
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v12.i2.pp716-721

Concealment of Files Blocked by Gmail with EOF-Based Image Steganography

Ichsan Taufik[1], Undang Syaripudin[2], Faiz M. Kaffah[3], Nanang Ismail[4], Jaka Giri Sobirin[5], Teddy Surya Gunawan[6]
[1,2,3,5]Informatic Engineering Department, Faculty of Science and Technology, UIN Sunan Gunung Djati
Jalan A.H Nasution 105, Bandung, Indonesia 40614, +62 22 7800525
[4]Electrical Engineering Department, Faculty of Science and Technology, UIN Sunan Gunung Djati
Jalan A.H Nasution 105, Bandung, Indonesia 40614, +62 22 7800525
[6]Department of Electrical and Computer Engineering, Kulliyyah of Engineering
International Islamic University Malaysia,
Jalan Gombak, 53100 Kuala Lumpur, Malaysia, (+603) 6196 4521

Article Info

Article history:

Received May 12, 2018
Revised Jul 13, 2018
Accepted Jul 27, 2018

Keywords:

End of File
Gmail
Image cover file
Insertion
Steganography

ABSTRACT

Nowadays, due to security concern, not all the process of sending files via email runs smoothly. There are several types of file extensions that are blocked when sent via email. For examples, there are several file extensions blocked by Gmail. This paper discusses steganographic implementation using End of File (EOF) algorithm to insert special file into image cover file with JPG and PNG format so that files with these extensions can be sent via email. Before a special extension file is inserted into the cover file, a compression process should be conducted first to make the file size smaller. The proposed algorithm is implemented on Visual Basic.Net software. Based on the tests performed, the application can insert Gmail-blocked file system to the image cover file, without changing the physical bit of the image cover file or file system that inserted with 100% success rate. The stego-image file is also successfully sent via email without being blocked.

Corresponding Author:

Ichsan Taufik,
Informatic Engineering Department, Faculty of Science and Technology,
UIN Sunan Gunung Djati,
Jalan A.H Nasution 105, Bandung, Indonesia 40614, +62 22 7800525
Email: ichsan@uinsgd.ac.id

1. INTRODUCTION

Recently, various threats in the virtual world such as hackers, crackers, carders make people worry about the security of information it sends [1], [2]. Various data security techniques have been developed, one of which is steganography which is defined as an invisible communication study. Steganography is usually associated with hiding the existence of communicated data. This will keep the confidentiality between the two communicating parties [3]. Various steganography techniques have been developed, including the Least Significant Bit (LSB) [4], Discrete Fourier transformation technique (DFT) [5], Discrete cosine transformation technique (DCT) [5], [6], Discrete Wavelet transformation technique (DWT) [6], pixel value differencing (PVD) [7], edges based data embedding method (EBE) [3], random pixel embedding method (RPE) [8], mapping pixel to hidden data method, labeling or connectivity method, and pixel intensity based [1], [3], [9]. If the hidden file is inserted into a cover file, then there are several types of steganography, including text steganography, image steganography, audio steganography, video steganography, and network or protocol steganography [3], [9]-[11]. Image-steganography is one of the popular method due to the prevalent of image sharing among users. In image-steganography, confidentiality is achieved by inserting data into the image cover file and producing a stego-image [3], in which pixel intensity is used to hide data.

In terms of data security, it is well known norm that security is inversely proportional to convenience. Therefore, added security features will most likely cause user inconvenience. One of the perceived cases is the blocking of some file types to be sent through the Gmail service. Some types of files that are forbidden to be sent or received through the Gmail service are .ade, .adp, .bat, .chm, .cmd, .com, .cpl, .exe, .hta, .ins, .isp, .jar, .jse , .lib, .ink, .mde, .msc, .msp, .mst, .pif, .scr, .sct, .shb, .sys, .vb, .vbe, .vbs, .vxd, .wsc,. wsf, and .wsh. Messages containing these file types will be blocked and Gmail will not accept these file types even if sent in a compressed form. This preventive measure by Gmail is executed automatically due to the potential security threat, i.e. virus or malicious code, in those files. In addition, Gmail does not allow users to send or receive damaged files, or files that do not work properly. To resolve this problem, it is possible to use steganographic techniques to insert files potentially blocked by Gmail by hiding the content in an image file. This paper proposes an insertion technique of special extension files blocked by Gmail with the End of File (EOF) method, in which the file to be sent is inserted at the end of the cover file [12], [13].

2. PROPOSED METHOD

As explained in the introduction, that the technique used in the research is image steganography. Where files with extensions blocked by Gmail will be hidden in a cover file of images. Figure.1 illustrates the typical image-steganography. End-of-file (EOF) is a special character used in any operating systems to mark the end of a file. If operating system found this special character, it will stop reading the file as it assumes that no more data is available. The flowchart of image-steganography process is shown in Figure. 2.

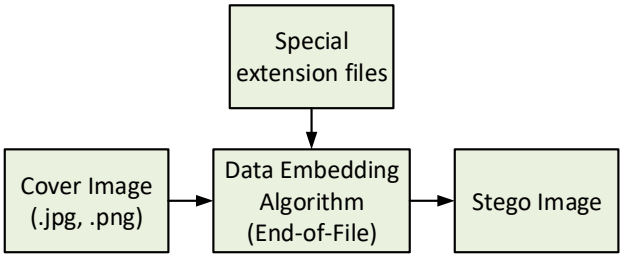

Figure 1. Typical image-steganography

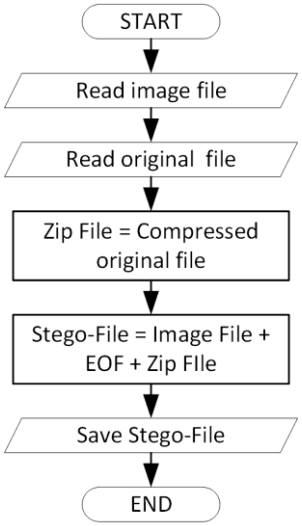

Figure 2. Flowchart of image-steganography process

The cover file is an image file with .jpg or .png format which serves to be a file insertion container with special extension. The original file will be inserted with some formats. .ade, .adp, .bat, .chm, .cmd, .com, .cpl, .exe, .hta, .ins, .isp, .jar, .jse , .lib, .lnk, .mde, .msc, .msp, .mst, .pif, .scr, .sct, .shb, .sys, .vb, .vbe, .vbs, .vxd, .wsc,. wsf, and .wsh. Compressed files are compressed extension files into a zip file to make the

process faster because the file size is smaller. Figure. 3 shows the one-direction end-to-end steganography process. Users input image file as cover, then do steganography process by using application developed using VB.Net.

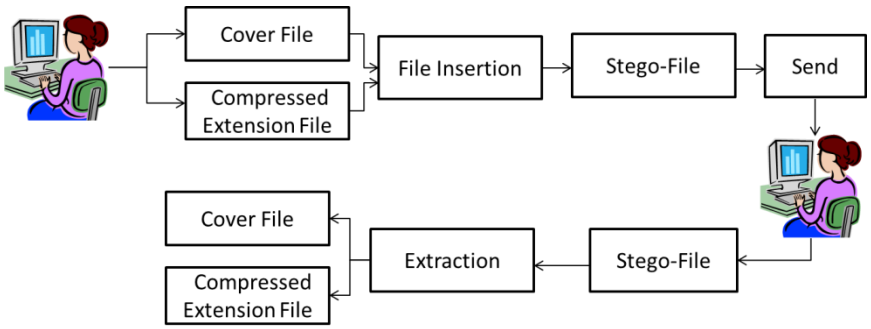

Figure 3. End-to-end process

3. SOFTWARE IMPLEMENTATION

3.1. Application Function

The functionality of developed applications can be illustrated with se case diagrams such as shown in Figure. 4. There are three main functions of the application, namely:

a) Stegano is a function to insert a file after compressing into a .zip file
b) Unstegano is a function to separate back the inserted files from the image file and extract the compressed extension file.
c) Sending email is a function to send email. This function is integrated with Gmail services.

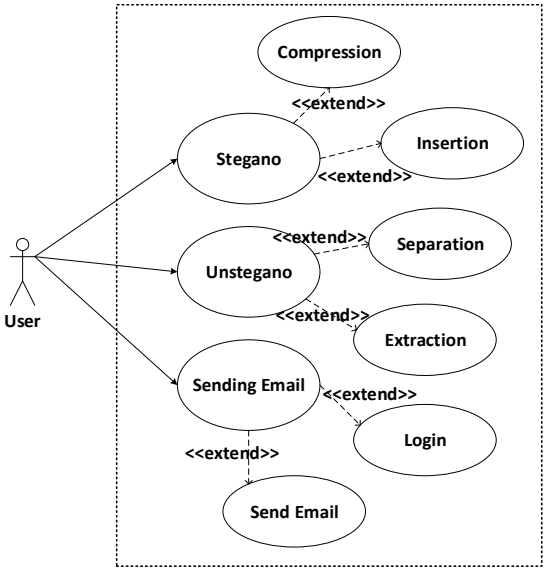

Figure 4. Use case diagram

3.2. Application Interface

The main page is the page that will appear first when the program is run and is shown in Figure. 5. On this page there are already menus for steganography, un-steganography and email delivery. Through the interface, we can perform the process of inserting the file into the image file and sending it to the destination email address. On the receiving end, this application is used to restore the inserted file to the original file.

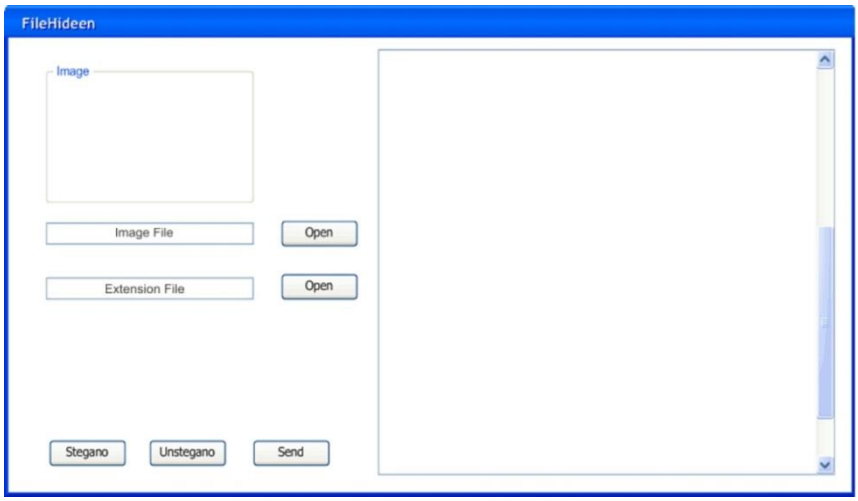

Figure 5. The application interface

4. RESULTS AND DISCUSSION

4.1. Testing Application Interface

Testing the application interface is done by trying all the elements of the function found on each page to find the errors that exist in the application. Test results show that all application interfaces are running without error. The test scenario and the results can be seen in Table 1.

Table 1. Testing Application Interface

ID	Scenario	Output		Information
		Success	Failed	
	Page *Stegano*			
A1	Inputting *file* image type .jpg, .png	√	-	Successfully input the cover file for insertion container
A2	Inputting *extention file* types .ade, .adp, .bat, .chm, .cmd, .com, .cpl, .exe, .hta, .ins, .isp, .jar, .jse, .lib, .lnk, .mde, .msc, .msp, .mst, .pif, .scr, .sct, .shb, .sys, .vb, .vbe, .vbs, .vxd, .wsc, .wsf, .wsh.	√	-	Successfully input file extension as object to be inserted
A3	Clicking the Stegano button	√	-	The process of inserting the extension file into a cover file is a "success" image and appears in the Stegano folder
	Page *Unstegano*			
A4	Input Steganographic file	√	-	Successfully input image file that has been inserted file extension
A5	Clicking the unstegano button	√	-	The process of extraction "successful" and appear in the stegano folder
	Page *Send*			
A6	Input email and password Gmail	√	-	successfully login to Gmail for the preparation of sending stenographic files
A7	Destination email input, message title, message body	√	-	Successfully input data as a companion stenographic file
A8	Input steganographic file	√	-	Successfully retrieved stenographic files in the form of image files and Images that appear on the form
A9	Clicking the send button	√	-	Successfully sending messages and stenographic files "sent messages"

4.2. Testing File Insertion (Stegano Process)

Testing is done by trying the insertion function for all types of files that are provided. Cover file used is image file extention .jpg and .png. While the files that are inserted are files that have a type that is blocked by Gmail, which is the extension .ade, .adp, .bat, .chm, .cmd, .com, .cpl, .exe, .hta, .ins, .isp, .jar, .jse, .lib, .ink, .mde, .msc, .msp, .mst, .pif, .scr, .sct, .shb, .sys, .vb, .vbe,. vbs, .vxd, .wsc, .wsf, and .wsh.

The extension files are tested, and the cover file used has various sizes. The combination of file extensions, cover files and the results can be seen in Table 2.

Table 2. Testing of the File Insertion Process

ID	*Extension File*	Size (MB)	Cover *File*			Output		
			JPG	PNG	Size (KB)	Success	Failed	Size (MB)
A1	.ade	7.400	√	-	19	√	-	7.24
A2	.adp	0.209	-	√	216	√	-	0.310
A3	.bat	0.095	√	-	61	√	-	0.155
A4	.chm	0.464	-	√	74	√	-	0.537
A5	.cmd	0.501	√	-	56	√	-	0.518
A6	.com	8.335	√	-	1303	√	-	9.39
A7	.cpl	6.782	-	√	127	√	-	6.62
A8	.exe	1.463	-	√	158	√	-	1.49
A9	.hta	0.636	-	√	26	√	-	0.661
A10	.ins	0.656	√	-	218	√	-	0.853
A12	.isp	1.877	-	√	1066	√	-	2.84
A13	.jar	1.839	√	-	739	√	-	2.40
A14	.jse	0.253	√	-	430	√	-	0.560
A15	.lib	0.831	-	√	161	√	-	0.991
A16	.lnk	0.910	√	-	77	√	-	0.907
A17	.mde	8.376	-	√	102	√	-	8,23
A18	.msc	7.4	-	√	18	√	-	7.24
A19	.msp	0.91	√	-	92	√	-	0.921
A20	.mst	8.376	√	-	189	√	-	8.31
A21	.vbe	0.831	-	√	352	√	-	1.15
A22	.wsf	0.253	-	√	217	√	-	0.347
A23	.wsh	0.209	-	√	622	√	-	0.715

The result of the insertion test of the file in Table 2 shows the success rate of 100%. it is for all types of embedded system files. The file size of the extension is the size of the file that has been compressed into .zip. While the output size shows the size of stego-image. The insertion speed is also affected by the file size.

4.3. Testing Sending File

Stego-image is further tested with file sending, and test results prove all stego-image successfully sent via email without any blocking. The results of the test delivery by email are shown by Table 3.

Table 3. Sending Stenographic Files

ID	Cover file	Size (MB)	Status
A1	.JPG	7.24	Sent
A2	.PNG	0.310	Sent
A3	.JPG	0.155	Sent
A4	.PNG	0.537	Sent
A5	.JPG	0.518	Sent
A6	.JPG	9.39	Sent
A7	.PNG	6.62	Sent
A8	.PNG	1.49	Sent
A9	.PNG	0.661	Sent
A10	.JPG	0.853	Sent
A12	.PNG	2.84	Sent
A13	.JPG	2.40	Sent
A14	.JPG	0.560	Sent
A15	.PNG	0.991	Sent
A16	.JPG	0.907	Sent
A17	.PNG	8.23	Sent
A18	.PNG	7.24	Sent
A19	.JPG	0.921	Sent
A20	.JPG	8.31	Sent
A21	.PNG	1.15	Sent
A22	.PNG	0.347	Sent
A23	.PNG	0.715	Sent

4.4. Testing Unstegano Process

Extraction test that runs on the stegano form that aims to restore the file extension which has been hidden in the image cover file can be seen in Table 4.

Table 4. Extraction file

ID	Stego-file		Extraction		Extension file	
	Type	Size (MB)	Success	Failed	Type	Size (MB)
A1	.JPG	7.24	√	-	.zip	7.400
A2	.PNG	0.310	√	-	.zip	0.209
A3	.JPG	0.155	√	-	.zip	0.095
A4	.PNG	0.537	√	-	.zip	0.464
A5	.JPG	0.518	√	-	.zip	0.501
A6	.JPG	9.39	√	-	.zip	8.335
A7	.PNG	6.62	√	-	.zip	6.782
A8	.PNG	1.49	√	-	.zip	1.463
A9	.PNG	0.661	√	-	.zip	0.636
A10	.JPG	0.853	√	-	.zip	0.656
A12	.PNG	2.84	√	-	.zip	1.877
A13	.JPG	2.40	√	-	.zip	1.839
A14	.JPG	0.560	√	-	.zip	0.253
A15	.PNG	0.991	√	-	.zip	0.831
A16	.JPG	0.907	√	-	.zip	0.910
A17	.PNG	8,23	√	-	.zip	8.376
A18	.PNG	7.24	√	-	.zip	7.4
A19	.JPG	0.921	√	-	.zip	0.91
A20	.JPG	8.31	√	-	.zip	8.376
A21	.PNG	1.15	√	-	.zip	0.831
A22	.PNG	0.347	√	-	.zip	0.253
A23	.PNG	0.715	√	-	.zip	0.209

5. CONCLUSION

This paper has presented an image-steganography technique to conceal file blocked by Gmail. The EOF algorithm is implemented by inserting the original file to cover image file. Results showed that the generated stego-image delivered successfully via Gmail without being blocked. To reduce the computational time and the final file size, first the original file was compressed so that the file size will be smaller. Various testing were conducted on the application interface, stegano process, file sending, and unstegano process.

REFERENCES

[1] F. J. Mabry, J. R. James, and A. J. Ferguson, "Unicode Steganographic Exploits: Maintaining Enterprise Border Security," *IEEE Security & Privacy* vol. 5, pp. 32-39, 2007.

[2] T. S. Gunawan, M. K. Lim, N. F. Zulkurnain, and M. Kartiwi, "On the Review and Setup of Security Audit using Kali Linux," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 11, pp. 51-59, 2018.

[3] J. Kour and D. Verma, "Steganography Techniques –A Review Paper," *International Journal of Emerging Research in Management &Technology*, vol. 3, pp. 132-135, 2014.

[4] M. Kaur and V. K. Sharma, "Encryption based LSB Steganography Technique for Digital Images and Text Data," *IJCSNS International Journal of Computer Science and Network Security*, vol. 16, pp. 90-97, 2016.

[5] M. Ramkumar and A. N. Akansu, "Capacity estimates for data hiding in compressed images," *IEEE Transactions on Image Processing*, vol. 10, pp. 1252 - 1263, 2001.

[6] R. J. Mstafa, K. M. Elleithy, and E. Abdelfattah, "A Robust and Secure Video Steganography Method in DWT-DCT Domains Based on Multiple Object Tracking and ECC," *IEEE Access*, vol. 5, pp. 5354 - 5365, 2017.

[7] M. Khodaei and K. Faez, "New adaptive steganographic method using least significant- bit substitution and pixel-value differencing," *IET Image Processing* vol. 6, pp. 677 - 686, 2012.

[8] A. Tiwari, S. R. Yadav, and N. K. Mittal, "A Review on Different Image Steganography Techniques," *International Journal of Engineering and Innovative Technology (IJEIT)*, vol. 3, pp. 121-124, 2014.

[9] Rakhi and S. Gawande, "A Review on Steganography Methods," *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*, vol. 2, pp. 4635-4638, 2013.

[10] V. Sharma and R. Thakur, "LSB modification based Audio Steganography using Trusted Third Party Key Indexing method," in Third International Conference on Image Information Processing (ICIIP), Waknaghat, India, pp., 2015.

[11] R. Amirtharajan and J. B. B. Rayyappan, "STeganography-Time to Time: a Review," *Research Journal on Information Technology*, vol. 5, pp. 53-66, 2013.

[12] Muslih and E. H. Rachmawanto, "Multimedia File Security With End-of-File Steganography Method for Maintaining a Confidential Message (in Bahasa Indonesia) " *Techno.COM*, vol. 15, pp. 1-6, 2016.

[13] Martono and Irawan, "A Use of End-of-File Steganography on Digital Watermarking (in bahasa Indonesia)," *Jurnal TICOM*, vol. 2, pp. 36-42, 2013.