Secret Communication on Facebook Using Image Steganography: *Experimental Study*

Budoor S. Edhah
Department of Information Systems
King Abdulaziz University, Saudi Arabia
beidhah@stu.kau.edu.sa

Daniyal M. Alghazzawi
Faculty of Computing & Information Technology
Department of Information Systems
King Abdulaziz University, Saudi Arabia
dghazzawi@kau.edu.sa

Li Cheng
Xinjiang Technical Institute of Physics & Chemistry
Chinese Academy of Sciences
chengli@ms.xjb.ac.cn

***Abstract*—Facebook is a popular online social network that provides the means for connecting people all over the world to communicate together on one venue through chatting, sharing photos, documents, and videos. The wide penetration of Facebook globally makes it a very attractive medium for image steganography, especially with millions of images uploaded daily, which further obscure steganography in the uploaded images. Transmitting photos through Facebook enforces the application of image processing to the uploaded photos prior to their publication, which has the consequence of altering the original features of the uploaded images. This paper presents a set of experiments for exploring Facebook image processing schemes. Also, it explores several methods for successfully applying steganography over Facebook.**

Keywords-steganography; online social network; Facebook; JPEG

I. INTRODUCTION

Online social networks (OSN) such as Facebook allow people to communicate easily worldwide. The great popularity of Facebook and its availability on many platforms on desktop and mobile computing platforms that are running different operating systems such as Windows, Unix, Android, and iOS enables Facebook to be a very attractive medium for performing steganography. Steganography is the discipline of hiding secret messages in another medium to extract it at the destination point, with the result that, among all the observers of the involved media, only the intended recipient is aware of the existence of the hidden message [1] [2]. However, transmitting hidden messages using image steganography via Facebook runs the risk of losing the embedded secret messages due to the imposed image processing applied to the uploaded photos. This paper explores Facebook image processing and examines several methods for successfully applying steganography via Facebook in which stego images can survive Facebook's image processing.

The paper is structured as follows. Section II provides background information about Facebook and steganography. Section III summarizes previous related work. Section IV tackles the selected steganography tools that we used throughout our experiments. Section V details our efforts in conducting the preliminary experiments for assessing Facebook image processing in different environments. Section VI describes our steganography experiments on a carrier photo selected from section V. Section VII compares the selected steganography tools based on our experiments' findings, while section VIII covers the discussion and section IX provides the concluding remarks of our work.

II. BACKGROUND

The following subsections provide an overview of Facebook, steganography, the history of steganography, steganography classification, steganography techniques, and steganalysis.

A. Facebook

Online social networks are defined as web-based services that enable users to create their own profile within a bounded system, communicate with other users with whom they share a connection, and view and navigate their list of connections along with the connections of other users within the system [3]. Facebook is a leading online social network with over 1.49 billion monthly active users worldwide as of August 2015 [4]. Its mission is to make the world more open and connected by allowing users to control and share information [5]. Through Facebook, users can share photos, videos, documents, and much other information. Besides offering the possibility for users to create their own pages and groups, it gives them the option to customize their privacy. Hence, users are able to make their pages public to everyone, public to friends only, public to specific users, or private. Moreover, it allows users to control who can follow, view, post, and comment on their pages [6]. With the outstanding popularity of Facebook and the number of photos shared daily, it would seem to be the perfect place to conduct steganography.

B. Steganography

Information security is one of the most important factors required for information communication through the Internet. Securing the secrecy of communication is known as cryptography while securing the existence of the message is known as steganography [7]. Steganography means "covered writing" and derives from the Greek word "stegos," which means "covered" and the word "graphia," which means "writing" [8]. It is the science of concealing information of which only the sender and the intended recipient are aware of the existence and the retrieval of the hidden message [9]. The process of applying steganography requires a set of elements, which are the cover object (C) that acts as a carrier to hold the secret message, the secret message (M) that is embedded in the cover object, the selected steganographic technique, and the stego key (K) that is used to encode and decode the secret message [10]. Steganography is gaining attraction due to the pressing issue of security over the Internet [7]. Fig. 1 illustrates the generic steganography process.

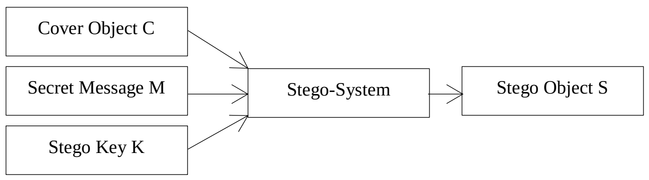


Figure 1. Generic steganography process

C. History of Steganography

While steganography has existed since 440 BC, its terminology was invented at the end of the fifteenth century [8]. Historically, steganography was used in messages that were hidden inside waxed wooden tablets, written on the stomachs of rabbits, or tattooed on slaves' scalps: after the hair had grown back, the secret messages would be undetected until the heads were shaved again [11]. Also, invisible ink and microdots were used as means for steganography; early in World War II, steganographic technology consisted mainly of those inks in which innocent letters contain a very different secret message between the lines [12] [13]. Moreover, five hundred years ago, the Italian mathematician Jerome Cardan adapted an ancient Chinese method of secret writing. This method consists of two parties who would each have a copy of the same paper mask that contains holes. The sender places the paper mask over a blank sheet of paper and writes the secret message through those holes; then the paper mask is taken off and the sender would continue writing on the blank paper in which the paper is viewed as having innocuous text [14]. Nowadays with the emergence of technology, steganography has begun to assume many new forms.

D. Steganography Classification

With digitization, steganography is used on digital objects. Such objects include images, music, videos, programs, and networks. All digital file formats can be used for steganography, and the most suitable formats are those that contain large amounts of redundant bits [15]. A steganography system can be classified based on two general approaches. The first approach is based on the type of the cover media, while the second one is based on the type of the embedding method. These consist of the insertion-based method, substitution-based method, or generation-based method [8]. An insertion-based method embeds the secret message in areas of the cover object that are ignored by applications; therefore, after the embedding process, the stego file size will be greater than the size of the cover object. The substitution method embeds the secret message by replacing insignificant bits from the cover object with bits of the secret message, hence this method maintains the same size of the original cover object, but in some cases, the quality of the cover object can be altered. Finally, the generation method uses the secret message to generate the stego file, hence, detecting this method is hard since no cover object exists [16] [17]. Fig. 2 illustrates the steganography classification.

E. Steganographic Techniques

There are two types of domains in which steganography is performed, the spatial domain and the frequency domain. In the spatial domain, the processing is performed directly on the pixel values of the photo, while in the frequency domain, the processing is executed indirectly; the pixel values are first transformed and then the processing will take place on the transformed coefficients [12]. Steganography can be applied through different techniques, which include the Least Significant Bit (LSB), Discrete Cosine Transform (DCT), or Discrete Wavelet Transform (DWT) [18]. The LSB technique is implemented in the spatial domain in which the bits of the hidden information are embedded into the least significant bits of the cover photo [19]. On the other hand, DCT and DWT steganographic techniques are implemented in the frequency domain in which the payload bits are embedded into the frequency components of the cover image after the image has been transformed from the spatial domain to the frequency domain [20] [21].

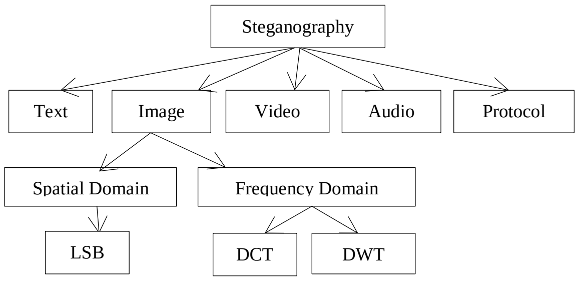


Figure 2. Steganography classification

F. Steganalysis

Steganalysis is the art of discovering the secret information in a cover object. Steganography is considered secure if the stego image does not reveal any detectable artifacts of the existence of the embedded message [22]. The stego images should have the same statistical properties as the cover image [23]. Steganography can be discovered through a set of comparisons between the original photo and the stego photo, or through detecting the steganography program's signatures on which most steganalysis programs depend [24]. Moreover, steganalysis is applied by checking the statistical abnormalities in the suspected photo in which the mean, the chi-square test, the linear analysis, and the variance are examined to measure the amount of departure from the expected norm and thereby reveal the distortion [25].

III. RELATED WORK

This section discusses several efforts that have complemented or otherwise inspired and influenced our research.

Castiglione, Cattaneo, and De Santis [26] analyzed the image processing of several online social networks: Facebook, Badoo, and Google+, which is imposed on the uploaded photos. The analysis mostly focused on the published images' properties on those OSNs and the changes that arose due to the processing with regard to the JPEG quantization table, pixel resolution, and related metadata. Of particular interest to our research is which photo formats Facebook uses for the uploaded photos and how it processes those images.

As proposed by Nagaraja, Houmasadr, Piyawongwisal, Singh, Agrawal, and Borisov [27], Stegobot is a bot network that communicates over unobservable communication channels. Stegobot is a social network botnet on Facebook for stealing data from users and sending those stolen data to the botmaster. It is based on a model of covert communication over a social network in which bots use Facebook image steganography to conceal the presence of communication within the image-sharing activity of user interaction. Commands are directed from one infected node to another, and the stolen data is returned to the botmaster, node after node. Through their work, a database of 116 different photos was used to identify the maximum JPEG resolution not altered by Facebook image processing. Then photos were resized to a lower size than the maximum Facebook constraint. After that, the bot communications were embedded through the use of the Yet Another Stenographic System (YASS) steganography method. Once the carrier images had been uploaded to Facebook, a YASS decoder was employed to extract the communications.

The YASS method [28] embeds data in randomized locations from which it will hinder blind steganalysis and the malicious processing of steganographic signals by rivals. YASS embeds data randomly in 8×8 blocks in which the embedding does not correspond with the 8×8 grid used during JPEG compression. YASS resists blind steganalysis and corruption from JPEG compression. Therefore, this method is better suited to be used on Facebook since it resists the JPEG compression applied by Facebook. However, Facebook image compression methods can still distort some bits of the steganography, but the majority of the message can be recovered and the lost data can be inferred when the data is in the form of text or photos. In our research, we also use Facebook to apply image steganography and investigate the Facebook image processing in order to provide better steganographic results over Facebook. Throughout the research, we have used multiple steganography tools to carry out our experiments, in which we aim to hide and transmit the secret message rather than hiding and transmitting bot network communications.

One of the most relevant works for steganography on Facebook was done by Owen Campbell Moore in April 2013 for his master's degree project [29]. Owen developed Secretbook, which is a Goggle Chrome web browser extension using Modified Linear Block Code method to encrypt up to 140 ASCII characters into an image. Facebook users can manually upload the image on their account and decrypt the message from Facebook images through the extension. Owen determined that Facebook compresses JPEG images with a quality factor of 75. Beckhusen in [30] demonstrates that Secretbook automatically compresses a JPEG image as Facebook would and then embeds the hidden message. In addition, it adds redundancy so any lasting alteration can be corrected by restoration from the copies. Our research approach is related to the work of Campbell-Moore, in which, for some part of our experiments, we have tried to investigate the image formats accepted and the image processing imposed by Facebook to photos uploaded a series of 50 times; we have attempted to allow photos to be processed by Facebook prior to applying steganography in order to minimize some of the Facebook image processing, which may destroy the embedded secret messages. Also, we proved that it is possible to use JPEG images to transmit secret messages longer than 140 characters through Facebook.

The work of Amsden, Chen, and Yuan [31] investigated a technique for applying steganography on Facebook cover photos. Their experiments demonstrated that Facebook cover photos can successfully hold hidden information to a capacity of at least 20% using the DCT coefficient embedding method. They concluded from their experiments that manual interaction with Facebook should be used for steganographic purposes instead of an automated Facebook integrated application. This research is also related to our work in which we have experimented applying steganography on Facebook cover photos by embedding secret messages up to 20 KB using different steganography tools. In addition, we have investigated the Facebook image processing on different platforms and different browsers, and we have concluded that applying steganography using Facebook's mobile application is not applicable due to the high intensity of image processing applied to the uploaded photos. In addition, we proved that we can apply steganography on Facebook post photos and Facebook profile photos using SilentEye tool.

A recent research [32] by Hiney, Dakve, Szczypirski, and Gaj investigated the compression that Facebook imposes on photos uploaded to the site through a set of experiments. Their work explored a method for minimizing the level of Facebook compression so that JPEG images can be used as steganography carriers on Facebook. Their experiments attempted to preprocess JPEG images through resizing them, converting them to 2048 * yyy and 960 * yyy resolutions, and compressing them prior to uploading to Facebook so that the Facebook processing on the uploaded photos will be minimized. Moreover, they selected photos that have Facebook upload to download file size ratios closest to 1.0 to start applying steganography on them. As they tested multiple steganography tools prior to their Facebook steganography test, they discovered that JPHide and JPSeek (JPHS) steganography tool yielded a success rate of 50% for recovery attempts. This research is very close to our work in which we have investigated the Facebook image processing on a selected JPEG photo for 50 times. In addition, we have tested different operating systems and browsers to determine whether these factors would have an influence on Facebook image processing or not. In our work, we also used different steganography methods to apply steganography on Facebook using different steganography tools.

To summarize our efforts, we have attempted in this research to explore Facebook image processing on uploaded photos for 50 times, and on various platforms, operating systems, and browsers. We have selected certain steganography tools that have success rates for applying steganography on Facebook based on the literature. Through this research, we have explored different methods for applying steganography on Facebook cover photos, post photos, and profile photos with various text file sizes of 77 bytes, 134 bytes, 1 KB, 10 KB, and 20 KB. Also, we have examined the steganography persistency on photos uploaded to Facebook and explored the possibility of applying nested steganography. Finally, we have demonstrated comparisons between the selected tools based on our findings. The following section explains the steganographic tools we selected and our motives behind this selection.

IV. STEGANOGRAPHY TOOLS

There are multiple steganographic tools available to apply steganography on different file formats. Since we interested in image steganographic tools that can be used to hide and extract secret messages from uploaded and downloaded Facebook photos respectively, we were very selective in choosing those tools that we planned to use to carry out our steganography experiments. Those selected tools are SilentEye, JPHide and JPSeek (JPHS), and Secretbook. We chose to select those three tools specifically because, after reviewing the literature, we learned that experiments have already been conducted using those three tools to apply steganography on Facebook. These experiments revealed some level of successful attempts in extracting the hidden messages after downloading the stego images from Facebook [31] [33]. This is especially the case for Secretbook, which has been designed specifically for steganography on Facebook. The following three subsections provide more details about these tools along with the reasons behind selecting them to carry out our experiments.

A. SilentEye

SilentEye is a cross-platform steganography application. This software combines a new steganography method and cryptography process by using a plug-in system. SilentEye allows information to be hidden in photos or sounds using the least significant bit (LSB) steganography technique [34]. Based on the experiment in [33], after downloading steganographic images (post photos) from Facebook, out of several steganographic tools that had been tested, which are JPHide and JPSeek, StegHide, F5, SteganPEG, and SilentEye, the researchers found SilentEye was the only tool that was able to extract the hidden messages from the downloaded Facebook stego post images; at some level, SilentEye was able to survive the Facebook compression process. Based on the test result of the aforementioned research, we selected SilentEye tool as one of the steganography tools that we used to carry out our experiments.

B. JPHide and JPSeek (JPHS)

JPHide and JPSeek (JPHS) is an open source steganography software that respectively embeds and recovers files in/from JPEG images. JPHide uses a DCT coefficient-embedding method to encrypt the secret message into the cover medium. The user of JPHide is required to set a passphrase to encode the data, while the user of JPSeek needs to have the exact passphrase to decode the secret message successfully [35]. Referring to the two experiments carried out in [31] and [32], both papers selected JPHide and JPSeek tool to perform their experiments of communicating secret messages via Facebook. In the first study, the researchers used JPHide and JPSeek tool to hide and extract secret messages from Facebook's cover photo successfully. While in the second study, the researchers tested several steganography tools to be used for Facebook, which are Open Puff, Outguess Rebirth, JPHide and JPSeek, Steg, F5, Our Secret, StegHide, Incognito, and Steganography. Out of all those tools tested, JPHide and JPSeek was the only tool that succeeded in applying steganography on Facebook with a 50% success rate of secret message retrieval from the downloaded Facebook stego post photo. Therefore, we have selected JPHide and JPSeek to be the second steganography tool we used in our experiments.

C. Secretbook

Secretbook is a Google Chrome extension that is designed to apply steganography on Facebook. Secretbook uses a Modified Linear Block Code method to encrypt up to 140 ASCII characters into photos [29] [36] [37]. We have selected Secretbook as one of the tools to be used in our experiments since this tool is designed specifically for applying steganography on Facebook, so including this tool will add value to our work and provide a wider perspective to our findings. The following section details our efforts in conducting the preliminary experiments for assessing Facebook image processing.

V. PRELIMINARY EXPERIMENTS

Facebook resizes and formats the uploaded photos and eliminates any redundant data on those photos prior to their publication. This allows photos to use minimum space and bandwidth when posted on Facebook pages. Facebook resizes regular photos to 720 pixels, 960 pixels, or 2048 pixels and cover photos to 851 pixels by 315 pixels. In addition, Facebook changes the format of all pictures uploaded to JPEG format and ensures that the size of the cover photo is less than 100 KB; otherwise, Facebook compresses the uploaded photo to the aforementioned sizes [38]. The following subsections demonstrate our efforts in investigating and assessing Facebook image processing on the uploaded photos through a set of preliminary experiments. The first section covers our first experiment of investigating Facebook image processing on different platforms. The second and the third sections cover our experiments using different operating systems and browsers respectively.

A. Assessing Facebook's Photo Processing Scheme

As a start and before applying steganography on photos and uploading them to Facebook, we attempted to pretest and explore Facebook environment and interaction with regular photos uploaded to it to investigate image processing, compression, or any relative changes that might alter the regular images' properties after uploading those images to Facebook. The purpose of this experiment was to check the possibility of any photo processing by Facebook that could have potential critical consequences on the secret messages embedded in the carrier's photos uploaded to Facebook when we applied steganography on them later. We conducted this test on Facebook in different platforms (mobile and personal computer [PC]), operating systems, and browsers and in diverse possible Facebook photo uploading locations. We have done the test on Facebook mobile application and Facebook website; and the test included Facebook cover photo, Facebook profile photo, and Facebook post photo category.

In our preliminary experiments, we chose a random regular photo that had no secret message on it and uploaded it to Facebook. The features of the photo that we selected to carry out the test are in the JPEG format, with a size of 92.3 KB, 94,607 bytes, dimensions of 705 pixels × 856 pixels, horizontal and vertical resolution of 72 dpi, bit depth of 24, resolution unit of 2, and color representation of sRGB. Fig. 3 illustrates the original photo selected to be used throughout all our experiments and Fig. 4 presents Facebook page highlighting the three different photo uploading locations. We have noticed that as we uploaded the original photo to Facebook, the photo's properties with regard to the size, dimensions, and even perceptibility have changed. We have recorded the observed variations between the original photo before uploading to Facebook and the changes that occurred after uploading; accordingly, we kept uploading the previous uploaded photo to Facebook several times to check if Facebook image processing would continue to be applied to the previous already processed image, or would it be stopped at a certain point in time?

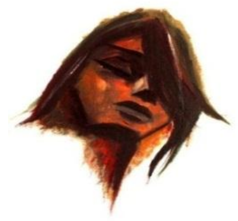


Figure 3. The original photo

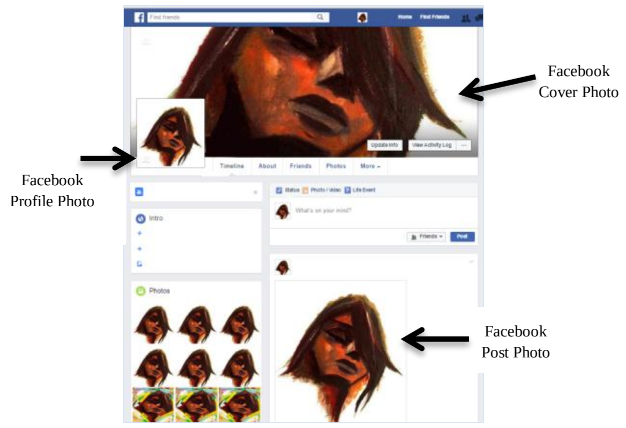


Figure 4. Facebook photo uploading locations

We repeated the experiment 50 times in the following sequence: we first uploaded the original photo, downloaded it, uploaded the former original downloaded photo, uploaded it again, downloaded it and so on for 50 times. Fig. 5 illustrates the 50-times uploading process. From this experiment's results, we learned that Facebook image processing will continue to be applied even if the photo has already gone through processing for all of the 50 uploading tries on Facebook, and this is applied on both Facebook mobile application and Facebook website. For the case of Facebook mobile application, the size of the uploaded original photo kept changing for all of the 50 tries of the experiment. However, for the case of uploading photos manually by interacting directly with Facebook website through the PC instead of uploading through Facebook mobile application, the photo processing, with regard to any change of the photo's size in bytes, occurred during the early tries of the uploading; and it was fixed at a certain point regardless of the number of times the photo was re-uploaded and downloaded again. To give specific details, as we started uploading the original photo of size 94,607 bytes and dimensions of 705 × 856 pixels using the PC, for the case of Facebook cover photo, the photo's size changed from the first uploading try to hold at the new size of 66,305 bytes and this size was fixed for the rest of the 50 uploading tries of the test. For the case of Facebook post photo, the image processing altered the size of the uploaded photo for the first 11 tries and on the 12th try the size of Facebook post photo got fixed at 14,250 bytes for the rest of the 50 test tries. On the other hand, the image processing with regard to the change in size of Facebook profile photo stopped at the 14th try of the test and proceeded to provide the same fixed size, which is 14,249 bytes, for the rest of the 50 test tries. We can clearly notice that Facebook image processing through a mobile application has a higher intensity than the processing through a PC. Also, the findings indicate that the level of photo compression through the PC is higher than the photo compression through the mobile application. Moreover, the uploaded photos though the PC appear very close to the original photo, while the uploaded photos through Facebook mobile application look distorted and different than the original photo. Table I demonstrates the series of size changes in bytes due to Facebook image processing, which occurred on the uploaded original photo using both a Facebook mobile application and a PC for 50 times; and Fig. 6 illustrates those findings on a chart.

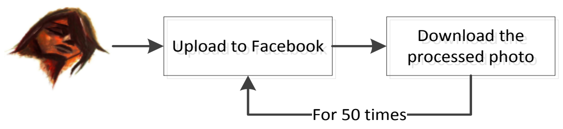


Figure 5. The 50 uploading process

TABLE I
Facebook Image Processing on the Original Photo Uploaded for 50 Times

	Facebook					
#	Mobile Application			Personal Computer (PC)		
	Cover	Post	Profile	Cover	Post	Profile
0	94,607	94,607	94,607	94,607	94,607	94,607
1	89,652	49,003	89,607	66,305	46,891	46,891
2	89,658	95,253	84,903	66,305	46,879	15,894
10	83,557	92,048	71,646	66,305	14,251	14,249
20	77,925	93,790	65,288	66,305	14,250	14,249
30	75,150	100,431	62,541	66,305	14,250	14,249
40	73,803	103,093	61,587	66,305	14,250	14,249
50	73,184	102,274	61,640	66,305	14,250	14,249

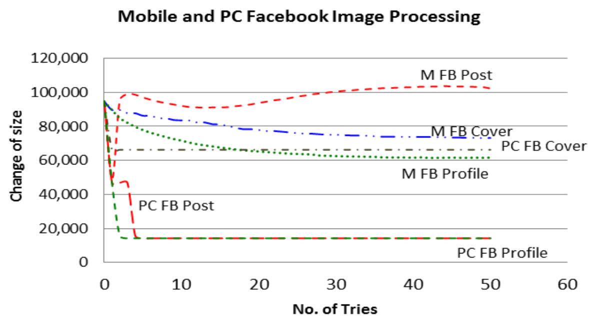
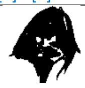
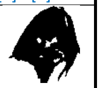


Figure 6. Facebook mobile and PC image processing

The experiment's outcomes from uploading the original photo to Facebook for 50 times are illustrated in Table II. Out of these 50 tries of uploading the previous downloaded photo to Facebook, the selected tries included in this table are try 0, which is the original photo before uploading to Facebook and try 1, which is the first time we uploaded the original photo to Facebook. This try reflects how the original photo's size was changed for the first time due to its first encounter with Facebook image processing. Also, we included try 2 in the table, which is the re-uploading of the first downloaded photo from Facebook. We selected this try because, as we noticed, most of the leaps of the original photo's change of size usually occurred on the 1st and 2nd tries of uploading to Facebook, and the rest of the 50 tries usually had a gradual change of sizes if they were not fixed. Moreover, tries 10, 20, 30, 40, and 50 are also included in the table below to give a view of the gradual changes of the original photo uploaded to Facebook due to the continuous image processing. This table also demonstrates the size in bytes and the dimensions in pixels for each uploaded photo. All photos in Table II are illustrated in black and white tones to highlight the gradual visual changes. Only in the case of Facebook mobile post photos, the black and white theme did not reflect all the visual changes; therefore, we presented a second version of Facebook Mobile post photos with a grayscale theme in Table III. To provide a comprehensive perspective about Facebook image processing that was imposed on the 50 uploading tries of the original photo, we attempted to use a photo comparison tool that makes visual and binary comparisons between files and tracks differences in order to investigate the continuous Facebook image processing on the preceding processed uploaded photos. The tool that we selected to track the differences between consecutive photos of the 50 tries is Araxis Merge [39]. This tool provides image comparisons in the changed pixels as well as illustrating binary comparisons, thereby demonstrating the block of bytes removals, insertions, and changes that occurred between photos. We used this tool in our experiment to make comparisons between each two consecutive uploaded photos from the 50 tries in which we have identified the block of bytes removals [-], bytes insertions [+], and bytes changes [#] for them. The findings are also illustrated in Table II.

TABLE II Mobile and PC Facebook Image Processing on the Original Photo Uploaded for 50 Times and the Binary Comparisons

	0	1	2	10	20	30	40	50
FB (M) Cover photo*	49,607 Bytes 705x856 Pixels	89,652 Bytes 705x856 Pixels [-]78 [+]74 [#]186	89,658 Bytes 705x856 Pixels [-]43 [+]83 [#]787	83,557 Bytes 705x856 Pixels [-]49 [+]45 [#]105	77,925 Bytes 705x856 Pixels [-]33 [+]31 [#]154	75,150 Bytes 705x856 Pixels [-]33 [+]30 [#]184	73,803 Bytes 705x856 Pixels [-]41 [+]38 [#]167	73,184 Bytes 705x856 Pixel [-]87 [+]81 [#]241
FB (M) Post photo*	49,607 Bytes 705x856 Pixels	94,003 Bytes 705X856 Pixels [-]99 [+]93 [#]232	95,253 Bytes 705X856 Pixels [-]70 [+]64 [#]483	92,048 Bytes 705X856 Pixels [-]4 [+]1 [#]143	93,790 Bytes 705X856 Pixels [-]2 [+]1 [#]405	100,431 Bytes 705X856 Pixels [-]6 [+]4 [#]433	103,093 Bytes 705X856 Pixels [-]8 [+]3 [#]441	102,274 Bytes 705X856 Pixels [-]2 [+]2 [#]422
FB (M) Profile photo*	49,607 Bytes 705x856 Pixels	89,607 Bytes 705X705 Pixels [-]84 [+]77 [#]112	84,903 Bytes 705X705 Pixels [-]57 [+]51 [#]291	71,646 Bytes 705X705 Pixels [-]46 [+]43 [#]73	65,288 Bytes 705X705 Pixels [-]23 [+]21 [#]84	62,541 Bytes 705X705 Pixels [-]35 [+]33 [#]104	 61,587 Bytes 705X705 Pixels [-]69 [+]62 [#]159	 61,640 Bytes 705X705 Pixels [-]49 [+]46 [#]142
FB (PC) Cover photo**	49,607 Bytes 705x856 Pixels	66,305 Bytes 705x856 Pixels [-]20 [+]21 [#]23	66,305 Bytes 705x856 Pixels [-]0 [+]0 [#]1	66,305 Bytes 705x856 Pixels [-]0 [+]0 [#]1	66,305 Bytes 705x856 Pixels [-]0 [+]0 [#]1	66,305 Bytes 705x856 Pixels [-]0 [+]0 [#]1	66,305 Bytes 705x856 Pixels [-]0 [+]0 [#]1	66,305 Bytes 705x856 Pixels [-]0 [+]0 [#]1
FB (PC) Post photo**	49,607 Bytes 705x856 Pixels	46,891 Bytes 705x856 Pixels [-]23 [+]10 [#]20	46,879 Bytes 705x856 Pixels [-]0 [+]1 [#]209	14,251 Bytes 705x856 Pixels [-]9 [+]21 [#]58	14,250 Bytes 705x856 Pixels [-]2 [+]2 [#]36	14,250 Bytes 705x856 Pixels [-]0 [+]0 [#]1	14,250 Bytes 705x856 Pixels [-]0 [+]0 [#]1	14,250 Bytes 705x856 Pixels [-]0 [+]0 [#]1
FB (PC) Profile photo**	49,607 Bytes 705x856 Pixels	46,891 Bytes 705x856 Pixels [-]23 [+]10 [#]20	15,894 Bytes 325x395 Pixels [-]9 [+]14 [#]60	14,249 Bytes 325x395 Pixels [-]15 [+]11 [#]86	14,249 Bytes 325x395 Pixels [-]1 [+]0 [#]19	14,249 Bytes 325x395 Pixels [-]0 [+]0 [#]1	14,249 Bytes 325x395 Pixels [-]0 [+]0 [#]1	14,249 Bytes 325x395 Pixels [-]0 [+]0 [#]1

*FB (M): Facebook through a Mobile Application, * *FB (PC): Facebook through a Personal Computer
Binary Comparisons Highlighting [-]: Number of Removed Blocks, [+]: Number of Inserted Blocks, and [#]: Number of Changed Blocks

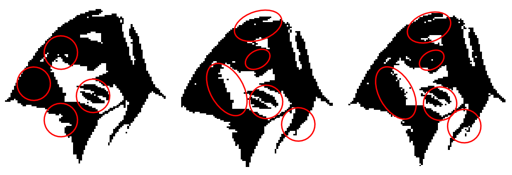
From the table above, we can view the gradual visual changes that occurred to the original photo uploaded to Facebook (FB) using both the personal computer (PC) and Facebook mobile application (M). We have noticed that the dimensions of Facebook profile photos using the Facebook mobile application are different than Facebook profile photos using the PC. This is due to the imposed crop feature in FB mobile app, whereas we didn't crop the photos when we uploaded them through Facebook website. Furthermore, we repeated the same experiment of the 50 uploading tries on Facebook mobile application and PC for three times to check if we are going to have the same outcomes if we upload the same original photo. Surprisingly, as we repeated the same experiment using the same original photo on the same app or PC, we found that every time we somehow received different results. Table IV demonstrates the different outcomes of repeating the same experiment using Facebook mobile application. Also, we clearly noticed that most of the changes, image processing, or distortions occurred when we uploaded photos using Facebook mobile app rather than a PC and especially when we uploaded photos as a Facebook post photo rather than a Facebook cover or profile photo. Fig. 7 illustrates the visual changes of three selected Facebook post photos throughout the 50 tries using a PC. Also, to provide a more accurate conclusion about those visual differences between the uploaded photos, we used with Araxis Merge tool, another image comparison tool (Image Comparer 3.8), which makes comparisons between photos and indicates the percentage of differences between them. We have selected two photos, try 5 and try 15, from the 50 tries of Facebook PC post photos and compared between them using Image Comparer; this tool indicated that 5% of differences exist between the two photos selected, which supports our observations; however, we depended on Araxis Merge more since this tool demonstrates the photos' differences with greater details. For Facebook cover photos, we noticed that the size of the original photo uploaded changed only on the 1st try of the experiment and then it remained fixed for the rest of the 50 tries. We suspect that although the rest of the 50 tries of Facebook cover photos held the same size, dimensions, and the perceptibility, other, invisible photo's properties might have existed that got changed during the experiments; this also applies to the late tries of Facebook post and profile photos in which the photos' sizes became fixed at the 12th and the 14th try respectively. To validate our assumption, we examined similar photos selected from the 50 tries of Facebook's cover, post, and profile photos in which they share the same size, dimensions, and manifestation and compared them using Araxis Merge to confirm our assumption, which is that although those photos hold the same size, dimensions, and manifestation, they are different and contain invisible changes. By doing so, we discovered through the binary comparison of each two consecutive photos of the 50 uploaded cover, post, and profile photos that they actually hold some binary differences. For the case of Facebook cover photo, we learned that a single block of bytes was changed between the consecutive uploaded Facebook cover photos, which confirmed our assumption. For the case of Facebook post photos that shared the same sizes, starting from try 12, the binary comparison proved that there are some differences with regard to the number of blocks of bytes removed, inserted, and changed; and these differences are minimized gradually till they reach to only 1 constant block of bytes changed for the remainder of the 50 tries starting precisely from the 19th try and forward. Furthermore, although Facebook profile photos had fixed sizes starting from try 14, the binary comparison revealed many changes on those photos; those changes gradually decreased for each proceeding uploading try until they stabilized at only 1 change for the remaining part of the 50 tries, starting also from try number 19. Fig. 8 illustrates the Araxis Merge photo comparison of try 2 and try 10 of the PC Facebook profile photos and Fig. 9 demonstrates their binary comparisons.

TABLE III Facebook Mobile Post Photos with Grayscale Theme

1	2	10	20	30	40	50
94,003 Bytes 705X856 Pixels [-]99 [+]93 [#]232	95,253 Bytes 705X856 Pixels [-]70 [+]64 [#]483	92,048 Bytes 705X856Pixels [-]4 [+]1[#]143	93,790 Bytes 705X856 Pixels [-]2 [+]1 [#]405	100,431 Bytes 705X856 Pixels [-]6 [+]4 [#]433	103,093 Bytes 705X856 Pixels [-]8 [+]3 [#]441	102,274 Bytes 705X856 Pixels [-]2 [+]2 [#]422

TABLE IV Different Facebook Processing Using the Facebook Mobile Application

Processing of Facebook Mobile Application			
#	***Experiment 1***	***Experiment 2***	***Experiment 3***
0	94,607	94,607	94,607
1	94,003	94,003	49,003
2	95,253	95,253	95,253
10	92,048	92,048	92,048
20	93,823	93,790	93,790
30	98,779	100,277	100,431
40	79,822	103,068	103,093
50	76,461	103,270	102,274



Try#1
46,891 Bytes
705x856 Pixels

Try#10
14,251 Bytes
705x856 Pixels

Try#50
14,250 Bytes
705x856 Pixels

Figure 7. Visual changes of PC Facebook uploaded post photos

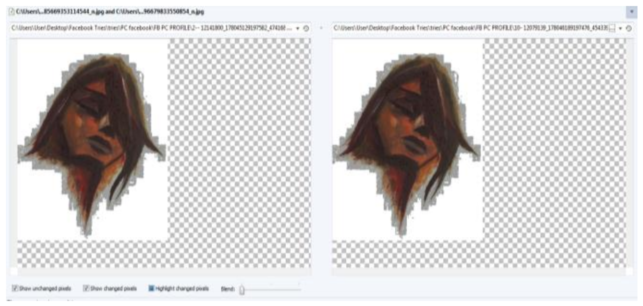


Figure 8. Araxis Merge photo comparison of tries 2 and 10 of PC Facebook profile photos

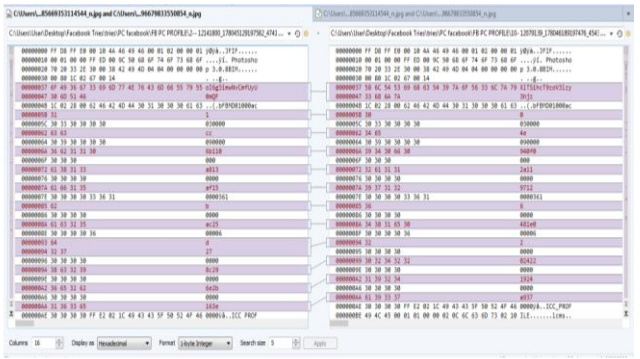


Figure 9. Araxis Merge binary comparison of try 2 and 10 of PC Facebook profile photos

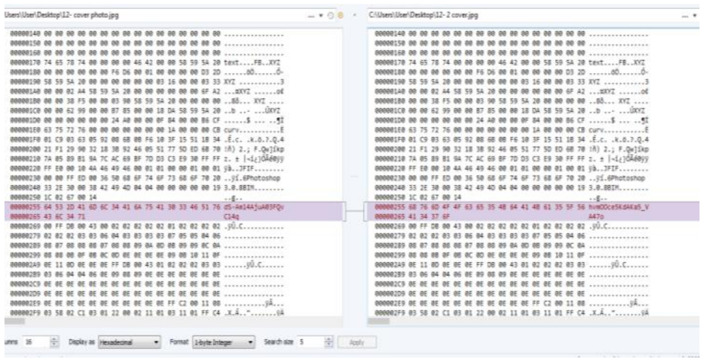


Figure 10. Binary comparison of the downloaded Facebook cover photos number 11

Moreover, after we had uploaded the original photo to Facebook 50 times, from among these 50 tries we selected, at random, a single photo and ran a small experiment: we attempted to upload this photo to Facebook to check if the uploaded version would have the same features each time as the one from our previous experiment or not. We repeated this experiment three times, during which we uploaded the same photo and checked the photo's features once we downloaded it. For this experiment, we selected try number 11 from Facebook cover, post, and profile photos, uploaded them to Facebook, downloaded them as photo number 12, and finally checked their features. The previous size of Facebook cover photo number 11 was 66,305 bytes and the size of cover photo number 12 was also 66,305 bytes. Whereas the previous size of Facebook post photo number 11 was 14,248 bytes and photo number 12 was 14,250 bytes. For the case of the Facebook profile photo, the size of the 11th photo was 14,247 bytes and for the 12th photo the size was 14,252 bytes. The purpose of this experiment was to check if uploading the same photo will lead exactly to the next photo being the same and to check if the downloaded photo from each try is also the same, confirming the outcomes with a binary comparison using Araxis Merge. From this experiment, we learned that as we upload the cover photo number 11, which holds the size of 66,305 bytes, we always get a photo number 12, which holds the same size of 66,305 bytes. The same thing applies to Facebook post and profile photos in which we get the same sizes as the previous experiment. The findings of repeating this experiment for three times were also the same. Checking the differences between each consecutive downloaded photo using a binary comparison, we discovered that the downloaded photos are different although they share the same size; hence, all of them hold one block of bytes changed between each consecutive photo, which indicates that the downloaded photos are different although they share the same size. Table V illustrates the findings of this experiment and Fig. 10 illustrates the single block of bytes changed between downloaded cover photo number 11 of the 1st and the 2nd tries. The following subsection discusses our next experiment using different operating systems.

TABLE V Binary Comparison of the Same Downloaded Photo

#	FB Cover Photo			FB Post Photo			FB Profile Photo		
	Try 1	*Try 2*	*Try 3*	*Try 1*	*Try 2*	*Try 3*	*Try 1*	*Try 2*	*Try 3*
11	66, 305	66, 305	66, 305	14,248	14,248	14,248	14,247	14,247	14,247
12	66, 305 [-]0 [+]0 [#]1	66, 305 [-]0 [+]0 [#]1	66, 305 [-]0 [+]0 [#]1	14,250 [-]0 [+]0 [#]1	14,250 [-]0 [+]0 [#]1	14,250 [-]0 [+]0 [#]1	14,252 [-]0 [+]0 [#]1	14,252 [-]0 [+]0 [#]1	14,252 [-]0 [+]0 [#]1

[-]: Number of Removed Blocks, [+]: Number of Inserted Blocks, and [#]: Number of Changed Blocks

B. *Different Operating Systems*

We repeated the same experiment in which we uploaded the same original photo to different photo uploading locations on Facebook for 10 times; however, this time we used different operating systems (OSs) to check if different operating systems have an effect on the same experiment's findings or not with regard to the uploaded photos' size in bytes and the binary comparisons. The purpose of this experiment was to confirm if repeating the same experiment using different operating systems would lead to an exact result or not. For this experiment we selected various operating systems, which were Windows, Mac, Fedora, iOS, and Android; we compared the findings based on Windows OS since the previous experiment had been conducted using Windows. We conducted the experiment on Facebook cover photo, Facebook post photo, and Facebook profile photo for 10 times, for each of which we uploaded the same original image to Facebook, downloaded it, uploaded it again, and so on for 10 times. Tables VI, VII, and VIII demonstrate the findings of this experiment and Figures 11, 12, and 13 illustrate those findings in charts. We noticed that for Facebook cover photo, Windows and Fedora operating systems led to the same results while each one of the other operating systems led to different findings. Also, according to Fig. 11, we noticed that in all operating systems, the curves of the Facebook cover photos started with a decreasing manner and around the second try the curves approximately started to have a fixed pattern. Checking Facebook post photo, we observed that Windows, Fedora, and Mac OSs had the same results for the first three uploading tries and on the fourth try they started to provide different outcomes from Windows in which Fedora and Mac have approximately the same outcomes. This is clearly illustrated in Table VII and Fig. 12. For the case of Facebook profile photo, we discovered that each operating system led to different outcomes from Windows, and we noticed that Fedora and Mac had approximately the exact results. Checking Fig. 13, we noticed that the curves of Facebook profile photos in all the operating systems started with a decreasing mode and, around the second try, the curves started to have an approximately regular pattern. From this experiment, we concluded that Facebook processes the uploaded photo differently based on the operating system.

TABLE VI Facebook Cover Photos through Different Operating Systems

Facebook Cover Photo					
#	***Windows***	***Fedora***	***Mac***	***iOS***	***Android***
0	94,607	94,607	94,607	94,607	94,607
1	66,305 [-]20 [+]21 [#]23	66,305 [-]20 [+]21 [#]23	46,891 [-]23 [+]10 [#]20	89,652 [-]78 [+]74 [#]186	40,721 [-]21 [+]1 [#]18
2	66,305 [-]0 [+]0 [#]1	66,305 [-]0 [+]0 [#]1	46,879 [-]0 [+]1 [#]209	89,658 [-]43 [+]38 [#]787	40,721 Identical
3	66,305 [-]0 [+]0 [#]1	66,305 [-]0 [+]0 [#]1	46,870 [-]2 [+]1 [#]197	87,772 [-]24 [+]23 [#]207	40,721 Identical
4	66,305 [-]0 [+]0 [#]1	66,305 [-]0 [+]0 [#]1	46,878 [-]1 [+]1 [#]185	87,762 [-]115 [+]112 [#]723	40,721 Identical
5	66,305 [-]0 [+]0 [#]1	66,305 [-]0 [+]0 [#]1	46,890 [-]5 [+]3 [#]191	86,273 [-]49 [+]90 [#]386	40,721 Identical
6	66,305 [-]0 [+]0 [#]1	66,305 [-]0 [+]0 [#]1	46,875 [-]3 [+]3 [#]162	86,218 [-]90 [+]82 [#]724	40,721 Identical
7	66,305 [-]0 [+]0 [#]1	66,305 [-]0 [+]0 [#]1	46,879 [-]1 [+]1 [#]131	84,833 [-]21 [+]18 [#]347	40,604 [-]4 [+]5 [#]185
8	66,305 [-]0 [+]0 [#]1	66,305 [-]0 [+]0 [#]1	46,880 [-]4 [+]4 [#]134	84,770 [-]65 [+]61 [#]831	40,604 Identical
9	66,305 [-]0 [+]0 [#]1	66,305 [-]0 [+]0 [#]1	46,879 [-]1 [+]1 [#]104	83,778 [-]44 [+]39 [#]443	40,604 Identical
10	66,305 [-]0 [+]0 [#]1	66,305 [-]0 [+]0 [#]1	46,869 [-]0 [+]0 [#]83	83,557 [-]33 [+]24 [#]662	40,604 Identical

[-]: Number of Removed Blocks, [+]: Number of Inserted Blocks, and [#]: Number of Changed Blocks

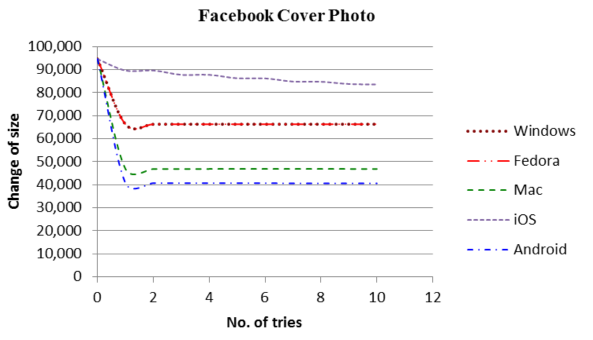


Figure 11. Facebook cover photos through different operating systems

TABLE VII Facebook Post Photos through Different Operating Systems

Facebook Post Photo					
#	***Windows***	***Fedora***	***Mac***	***iOS***	***Android***
0	94,607	94,607	94,607	94,607	94,607
1	46,891 [-]23 [+]10 [#]20	46,891 [-]23 [+]10 [#]20	46,891 [-]23 [+]10 [#]20	49,003 [-]99 [+]93 [#]232	40,721 [-]21 [+]1 [#]18
2	46,879 [-]0 [+]1 [#]209	46,879 [-]0 [+]1 [#]209	46,879 [-]0 [+]1 [#]209	95,253 [-]70 [+]64 [#]483	40,604 [-]4 [+]5 [#]185
3	46,870 [-]3 [+]0 [#]195	46,870 [-]3 [+]0 [#]195	46,870 [-]3 [+]0 [#]195	99,044 [-]9 [+]8 [#]316	40,586 [-]0 [+]0 [#]36
4	15,897 [-]14 [+]17 [#]52	46,878 [-]1 [+]1 [#]185	46,878 [-]1 [+]1 [#]185	98,641 [-]4 [+]3 [#]666	40,580 [-]0 [+]0 [#]19
5	14,258 [-]14 [+]8 [#]81	46,890 [-]5 [+]3 [#]191	46,890 [-]5 [+]3 [#]191	97,012 [-]3 [+]3 [#]733	40,582 [-]0 [+]0 [#]12
6	14,251 [-]4 [+]3 [#]93	46,875 [-]3 [+]3 [#]162	46,875 [-]3 [+]3 [#]163	95,665 [-]6 [+]2 [#]761	40,582 [-]0 [+]0 [#]1
7	14,248 [-]2 [+]3 [#]95	46,879 [-]1 [+]1 [#]131	46,879 [-]1 [+]1 [#]131	94,544 [-]4 [+]4 [#]720	40,582 [-]0 [+]0 [#]1
8	14,252 [-]3 [+]3 [#]90	46,880 [-]4 [+]4 [#]134	46,880 [-]4 [+]4 [#]134	93,585 [-]17 [+]4 [#]722	40,582 [-]0 [+]0 [#]1
9	14,250 [-]1 [+]1 [#]62	46,879 [-]1 [+]1 [#]104	46,879 [-]1 [+]1 [#]104	92,869 [-]11 [+]3 [#]786	40,582 [-]0 [+]0 [#]1
10	14,251 [-]0 [+]0 [#]45	15,948 [-]6 [+]21 [#]48	15,948 [-]6 [+]21 [#]48	92,048 [-]1 [+]2 [#]914	40,582 [-]0 [+]0 [#]1

[-]: Number of Removed Blocks, [+]: Number of Inserted Blocks, and [#]: Number of Changed Blocks

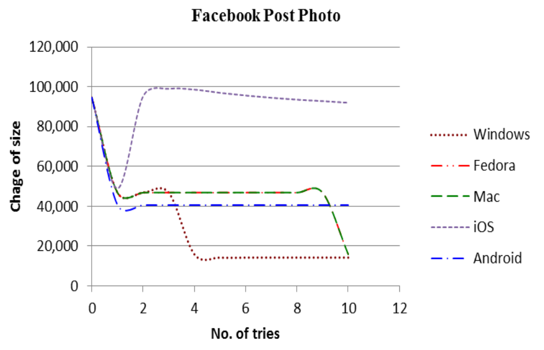


Figure 12. Facebook post photos through different operating systems

TABLE VIII Facebook Profile Photos through Different Operating Systems

Facebook Profile Photo					
#	***Windows***	***Fedora***	***Mac***	***iOS***	***Android***
0	94,607	94,607	94,607	94,607	94,607
1	46,891 [-]23 [+]10 [#]20	46,891 [-]23 [+]10 [#]20	46,891 [-]23 [+]10 [#]20	89,607 [-]84 [+]77 [#]112	35,162 [-]25 [+]1 [#]11
2	15,894 [-]9 [+]14 [#]60	46,879 [-]0 [+]1 [#]209	46,879 [-]0 [+]1 [#]209	84,903 [-]57 [+]51 [#]291	35,162 Identical
3	14,249 [-]17 [+]11 [#]83	46,870 [-]2 [+]0 [#]196	46,870 [-]3 [+]0 [#]195	82,159 [-]60 [+]55 [#]391	35,162 Identical
4	14,246 [-]0 [+]0 [#]99	46,878 [-]1 [+]1 [#]185	46,878 [-]1 [+]1 [#]186	79,713 [-]0 [+]1 [#]132	35,162 Identical
5	14,246 [-]1 [+]1 [#]108	46,890 [-]5 [+]3 [#]199	46,890 [-]5 [+]3 [#]191	77,853 [-]52 [+]49 [#]301	35,162 Identical
6	14,247 [-]3 [+]3 [#]86	46,875 [-]3 [+]3 [#]162	46,875 [-]3 [+]3 [#]162	76,287 [-]41 [+]38 [#]334	35,162 Identical
7	14,246 [-]5 [+]2 [#]90	46,879 [-]1 [+]1 [#]131	46,879 [-]1 [+]1 [#]131	74,973 [-]32 [+]27 [#]392	34,793 [-]8 [+]5 [#]208
8	14,250 [-]2 [+]2 [#]67	46,880 [-]4 [+]4 [#]134	46,880 [-]4 [+]4 [#]134	73,721 [-]39 [+]35 [#]301	34,793 Identical
9	14,248 [-]0 [+]0 [#]39	46,879 [-]1 [+]1 [#]104	46,879 [-]1 [+]1 [#]104	72,583 [-]37 [+]34 [#]338	34,793 Identical
10	14,249 [-]1 [+]1 [#]38	46,869 [-]0 [+]0 [#]83	46,869 [-]0 [+]0 [#]83	71,646 [-]9 [+]8 [#]328	34,793 Identical

[-]: Number of Removed Blocks, [+]: Number of Inserted Blocks, and [#]: Number of Changed Blocks

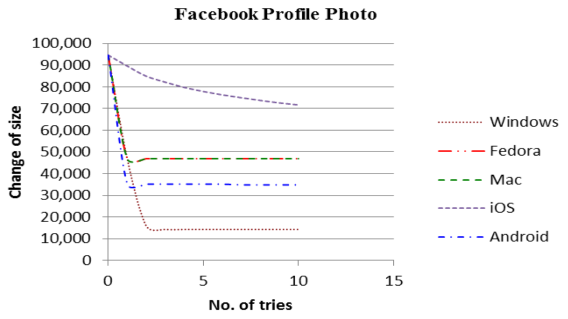


Figure 13. Facebook profile photos through different operating systems

C. *Different Web Browsers*

We again conducted the same previous experiment but this time examining four different Internet browsers in which we uploaded the same original photo to Facebook website for 10 times to check if the type of the browser has an effect on the experiment findings of Facebook image processing or not; the experiment included the photos' size in bytes and the binary comparison. The four browsers that we selected to carry out this experiment were Google Chrome, Internet Explorer, Mozilla Firefox, and Safari. The findings of this experiment are presented in Tables IX, X, and XI. From the experiment findings, we noticed that for the case of Facebook cover photo, all the browsers led to the exact same results. For the case of Facebook post photo, all the browsers led to the same results except for the case of Google Chrome, which led to the exact same result as the rest of the browsers until it started to have different results starting from try number 6. Finally, for the case of Facebook profile photo, again all the browsers led to the same exact results except for Google Chrome, which had the exact same results as the other browsers until it started to change starting from try number 7. We concluded from this experiment that the type of the browser will not have a great effect on the future steganography experiments, at least in the first five tries for all the browsers. Also, through this experiment, we observed that in all the browsers there is only one block of bytes changes between consecutive uploaded photos when we upload them as cover photos rather than post or profile photos. Thus, we predicted that applying steganography on Facebook cover photos could lead to the desired results, which we will explore in the following section. The subsequent section discusses steganography experiments using different methods.

TABLE IX Facebook Cover Photos through Different Internet Browsers

Windows				
Facebook Cover Photo				
#	***Google Chrome***	***Firefox***	***Internet Explorer***	***Safari***
0	94,607	94,607	94,607	94,607
1	66,305 [-]20 [+]21 [#]23	66,305 [-]20 [+]21 [#]23	66,305 [-]20 [+]21 [#]23	66,305 [-]20 [+]21 [#]23
2	66,305 [-]0 [+]0 [#]1	66,305 [-]0 [+]0 [#]1	66,305 [-]0 [+]0 [#]1	66,305 [-]0 [+]0 [#]1
3	66,305 [-]0 [+]0 [#]1	66,305 [-]0 [+]0 [#]1	66,305 [-]0 [+]0 [#]1	66,305 [-]0 [+]0 [#]1
4	66,305 [-]0 [+]0 [#]1	66,305 [-]0 [+]0 [#]1	66,305 [-]0 [+]0 [#]1	66,305 [-]0 [+]0 [#]1
5	66,305 [-]0 [+]0 [#]1	66,305 [-]0 [+]0 [#]1	66,305 [-]0 [+]0 [#]1	66,305 [-]0 [+]0 [#]1
6	66,305 [-]0 [+]0 [#]1	66,305 [-]0 [+]0 [#]1	66,305 [-]0 [+]0 [#]1	66,305 [-]0 [+]0 [#]1
7	66,305 [-]0 [+]0 [#]1	66,305 [-]0 [+]0 [#]1	66,305 [-]0 [+]0 [#]1	66,305 [-]0 [+]0 [#]1
8	66,305 [-]0 [+]0 [#]1	66,305 [-]0 [+]0 [#]1	66,305 [-]0 [+]0 [#]1	66,305 [-]0 [+]0 [#]1
9	66,305 [-]0 [+]0 [#]1	66,305 [-]0 [+]0 [#]1	66,305 [-]0 [+]0 [#]1	66,305 [-]0 [+]0 [#]1
10	 66,305 [-]0 [+]0 [#]1	 66,305 [-]0 [+]0 [#]1	 66,305 [-]0 [+]0 [#]1	 66,305 [-]0 [+]0 [#]1

[-]: Number of Removed Blocks, [+]: Number of Inserted Blocks, and [#]: Number of Changed Blocks

TABLE X Facebook Post Photos through Different Internet Browsers

Windows				
Facebook Post Photo				
#	***Google Chrome***	***Firefox***	***Internet Explorer***	***Safari***
0	94,607	94,607	94,607	94,607
1	46,891 [-]23 [+]10 [#]20	46,891 [-]23 [+]10 [#]20	46,891 [-]23 [+]10 [#]20	46,891 [-]23 [+]10 [#]20
2	46,879 [-]0 [+]1 [#]209	46,879 [-]0 [+]1 [#]209	46,879 [-]1 [+]2 [#]208	46,879 [-]0 [+]1 [#]209
3	46,870 [-]3 [+]0 [#]195	46,870 [-]2 [+]0 [#]196	46,870 [-]2 [+]0 [#]196	46,870 [-]3 [+]0 [#]195
4	46,878 [-]14 [+]17 [#]52	46,878 [-]1 [+]1 [#]186	46,878 [-]1 [+]1 [#]185	46,878 [-]1 [+]1 [#]185
5	46,890 [-]14 [+]8 [#]81	46,890 [-]5 [+]3 [#]191	46,890 [-]5 [+]3 [#]191	46,890 [-]5 [+]3 [#]191
6	15,922 [-]4 [+]3 [#]93	46,875 [-]3 [+]3 [#]162	46,875 [-]3 [+]3 [#]162	46,875 [-]3 [+]3 [#]162
7	14,276 [-]2 [+]3 [#]95	46,879 [-]1 [+]1 [#]131	46,879 [-]1 [+]1 [#]132	46,879 [-]1 [+]1 [#]132
8	14,274 [-]3 [+]3 [#]90	46,880 [-]4 [+]4 [#]134	46,880 [-]4 [+]4 [#]134	46,880 [-]4 [+]4 [#]134
9	14,273 [-]1 [+]1 [#]62	46,879 [-]1 [+]1 [#]104	46,879 [-]1 [+]1 [#]104	46,879 [-]1 [+]1 [#]104
10	 14,273 [-]0 [+]0 [#]45	 86,869 [-]0 [+]0 [#]83	 86,869 [-]0 [+]0 [#]83	 86,869 [-]0 [+]0 [#]83

[-]: Number of Removed Blocks, [+]: Number of Inserted Blocks, and [#]: Number of Changed Blocks

TABLE XI Facebook Profile Photos through Different Internet Browsers

Windows				
Facebook Profile Photo				
#	***Google Chrome***	***Firefox***	***Internet Explorer***	***Safari***
0	94,607	94,607	94,607	94,607
1	46,891 [-]23 [+]10 [#]20	46,891 [-]23 [+]10 [#]20	46,891 [-]23 [+]10 [#]20	46,891 [-]23 [+]10 [#]20
2	46,879 [-]9 [+]14 [#]60	46,879 [-]0 [+]1 [#]209	46,879 [-]0 [+]1 [#]209	46,879 [-]0 [+]1 [#]209
3	46,870 [-]17 [+]11 [#]83	46,870 [-]2 [+]0 [#]196	46,870 [-]3 [+]0 [#]195	46,870 [-]2 [+]0 [#]196
4	46,878 [-]0 [+]0 [#]99	46,878 [-]1 [+]1 [#]186	46,878 [-]1 [+]1 [#]186	46,878 [-]1 [+]1 [#]185
5	46,890 [-]1 [+]1 [#]108	46,890 [-]5 [+]3 [#]191	46,890 [-]5 [+]3 [#]191	46,890 [-]5 [+]3 [#]191
6	46,875 [-]3 [+]3 [#]86	46,875 [-]3 [+]3 [#]162	46,875 [-]3 [+]3 [#]162	46,875 [-]3 [+]3 [#]162
7	66,305 [-]5 [+]2 [#]90	46,879 [-]1 [+]1 [#]131	46,879 [-]1 [+]1 [#]131	46,879 [-]1 [+]1 [#]131
8	46,892 [-]2 [+]2 [#]67	46,880 [-]4 [+]4 [#]134	46,880 [-]4 [+]4 [#]134	46,880 [-]4 [+]4 [#]134
9	46,890 [-]0 [+]0 [#]39	46,879 [-]1 [+]1 [#]104	46,879 [-]1 [+]1 [#]104	46,879 [-]1 [+]1 [#]104
10	 46,879 [-]1 [+]1 [#]38	 46,869 [-]0 [+]0 [#]83	 46,869 [-]0 [+]0 [#] 83	 46,869 [-]0 [+]0 [#]83

[-]: Number of Removed Blocks, [+]: Number of Inserted Blocks, and [#]: Number of Changed Blocks

VI. STEGANOGRAPHY EXPERIMENTS

This section covers steganography experiments on Facebook through adopting several methods. All the experiments are applied on the original photo previously selected and tested in different photo posting locations of Facebook (Facebook cover photo, Facebook post photo, and Facebook profile photo). Through these experiments, we attempted to explore different methods for applying steganography on Facebook in order to have a wider perspective of various possibilities by which stego images at some level would survive the Facebook image processing, and the secret embedded messages would be extracted successfully at the destination point. We generated two main different scenarios or methods for applying steganography on Facebook. Moreover, we experimented with hiding different payload capacities; hence, we prepared different sizes of secret messages in a text file format that would be embedded in the original image in order to test the capability of the message retrieval after uploading the stego image on Facebook. The secret messages' capacities that we selected to carry out our steganography experiments were 77 bytes, 134 bytes, 1 kilobyte, 10 kilobytes, and 20 kilobytes. The purpose of choosing such sizes was to investigate the possibilities for the secret message retrieval; hence, if it was successful in the cases that have smaller sizes such as 77 bytes, 134 bytes, and 1 Kilobyte, would it also survive holding greater sizes such as 10 and 20 Kilobytes? More precisely, we selected payload capacities of 77 and 134 bytes in particular for our steganography experiments, because one of the steganography tools adopted in this research (Secretbook) suggested those sizes for the original photo that we had selected; thus, we aimed to unify the payload sizes for all the tools for the purpose of the subsequent comparison of tools in section VII. We discuss more details of the motives behind selecting 77 bytes and 134 bytes in the following subsection. Moreover, we selected 1 kilobyte for the test because this size had already been tested in [31]; and through the aforementioned research, the researchers proved the ability to retrieve secret messages from Facebook cover photo at a capacity of at least 20%, which inspired us to include more sizes in the test, such as 10 and 20 kilobytes. The steganography experiments were conducted using the three aforementioned tools, which are SilentEye, JPHide and JPSeek (JPHS), and Secretbook. We have tested (SilentEye and JPHide and JPSeek) tools with regards to their abilities for hiding and retrieving the hidden messages prior to uploading the stego photos on Facebook; thus, any obstacles that arise to retrieving the secret message from the downloaded photos will be due solely to Facebook image processing. The following subsections provide more details for the two main methods we employed during our steganography experiments on Facebook.

A. *Experiment 1: Steganography without Facebook Image Processing*

The first method we followed for transmitting secret messages using Facebook is considered a traditional one. We used the steganographic tools (SilentEye, JPHide and JPSeek (JPHS), and Secretbook) for embedding secret messages of different sizes in the original photo, uploading the stego photo on Facebook, downloading the uploaded stego photo, and finally using the same tools to check the ability for retrieving the embedded message. For the case of Secretbook tool, the maximum size of the embedded secret messages was up to 134 bytes only, which is explained comprehensively in the following paragraph. Through using SilentEye tool, we discovered we were able to retrieve the secret messages from Facebook cover photo in all cases of the different message sizes. For the case of Facebook post photo, SilentEye was successful in retrieving the embedded messages when their sizes were only 77 bytes, 134 bytes, and 1 kilobyte and it failed to retrieve messages that held the sizes of 10 or 20 kilobytes. For the case of Facebook profile photo, SilentEye was able to retrieve messages from the downloaded profile photos that held the size of 77 bytes, 134 bytes, and 1 kilobyte and failed in the rest of the tries. On the other hand, the finding from using the second tool, which is JPHide and JPSeek (JPHS), is that this tool was able to retrieve the secret messages from all Facebook cover photos with different payload capacities whereas it failed in all the tries of retrieving secret messages from Facebook post and profile photos. For the case of Secretbook, this tool was successful in retrieving secret messages of a size only of 77 and 134 bytes from all Facebook cover, post, and profile photos; and for other message sizes, it was not applicable for Secretbook to embed messages of such sizes. Table XII illustrates the findings of the 1st steganography experiment.

TABLE XII Findings of Experiment 1 Steganography without Facebook Processing

Steganography without Facebook Image Processing															
	FB Cover Photo					***FB Post Photo***					***FB Profile Photo***				
Tool	77 B	*134 B*	1 KB	10 KB	20 KB	77 B	134 B	1 KB	10 KB	20 KB	77 B	134 B	1 KB	10 KB	20 KB
SilentEye	✓	✓	✓	✓	✓	✓	✓	✓	×	×	✓	✓	✓	×	×
JPHS	✓	✓	✓	✓	✓	×	×	×	×	×	×	×	×	×	×
Secretbook	✓	✓	N/A	N/A	N/A	✓	✓	N/A	N/A	N/A	✓	✓	N/A	N/A	N/A

N/A: Not Applicable

Secretbook tool hides and retrieves the secret message while it is on Facebook page. Secretbook automatically determines the max payload capacity that can be hidden in the photo; therefore, we could not apply the same experiment where we hide the same predetermined aforementioned sizes of texts (1 KB, 10 KB, and 20 KB) since it is not applicable to this tool. As we uploaded the original photo to Secretbook, this tool determined the max payload capacity that could be embedded in the uploaded photo, which was identified as 77 characters. Therefore, we tried this experiment with the same size that Secretbook suggested and we attempted to include this proposed size in all our steganography experiments with the two tools (SilentEye and JPHide and JPSeek [JPHS]) for the sake of the comparison between all of the three tools. As we embedded 77 characters on the original photo using Secretbook and posted the photo on Facebook to extract the hidden message through Secretbook, we found that no message had been hidden in the uploaded photo, as Secretbook stated. We inferred that Secretbook made an error in embedding the secret message due to the white background of the original photo that we had uploaded; hence, according to Secretbook, it is clearly stated before hiding any message in any photo, it is important for the user to choose high-quality images with dimensions of 960×720 pixels for best outcomes; moreover, the user needs to avoid images that have large areas of sky or any single color since these sorts of images are prone to errors [30] [40]. Based on this information, we at first tried to add more colors to the white background of the original photo and used this first modified photo for our steganography experiment via Secretbook, as is displayed in Fig. 14 image (b). Uploading this adjusted stego photo to Facebook, Secretbook failed to indicate any secret message existed on the first modified photo. Consequently, we attempted to re-modify the original photo, which previously had the dimensions of 705×856 pixels to the dimensions suggested by Secretbook, which were 960×720 pixels and re-modified the background from light colors to a rich colorful background, as is illustrated in Figure 14 (c). Figure 11 displays the modification series of the original photo illustrating the first and the second modified photos. As we modified the original photo for the second time to match the Secretbook recommendation of steganography, we repeated the same experiment where we uploaded the new modified original photo to Secretbook, in which the tool suggested a new max payload capacity for embedding the secret message, which was indicated by 134 characters. Posting this stego photo to Facebook, Secretbook was able to extract the secret message.

Thus, this new modified original photo is the one selected to perform our steganography experiments with Secretbook and with the size of 134 bytes suggested; whereas for the other tools (SilentEye and JPHS), we used the same original photo that has no modification with different payload capacities ranging from 77 bytes to 20 kilobytes including the new size of 134 bytes identified by Secretbook in our steganography experiments.

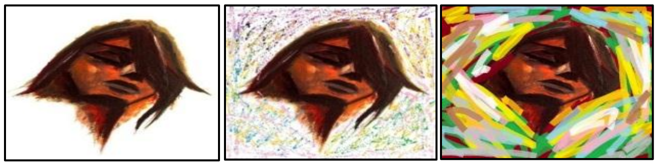

(a) Original photo (b) First modified (c) Second modified photo

Figure 14. The original photo after modification suggested by Secretbook

For all the failed attempts at retrieving the embedded secret messages from the previous experiment, we attempted to repeat the same experiment on the failed cases for three rounds. The purpose of applying this method was to check the possibility of retrieving the re-embedded secret messages from all previous failed attempts at extracting hidden messages from Facebook stego downloaded photos. To provide a clearer view of the mechanism of this method, we reviewed the findings of the first experiment and took all downloaded Facebook photos of failed message retrieval attempts and embedded the same secret messages again on them. We again uploaded those stego photos to Facebook, downloaded them, and checked if whether it was possible this time to retrieve the secret message or not. We repeated this experiment three times. Referring to the findings of the first experiment, SilentEye failed to retrieve the secret message from Facebook post and profile photos when the message size was 10 and 20 kilobytes, whereas JPHS failed to retrieve secret messages from Facebook post and profile photos for all message sizes. After embedding the secret messages again on those photos, except for the case of the 20-kilobyte message using SilentEye tool in which SilentEye indicated that there was not enough space on the photo to embed new messages; and subsequently uploading them to Facebook again for the second time, then downloading them, SilentEye failed to retrieve any secret messages on the first round, whereas JPHS was able to retrieve secret messages from Facebook post and profile photos when the size was 77 and 134 bytes only. Repeating the same experiment again, SilentEye was able to retrieve the secret message from the Facebook post photo that had a secret message of size 10 kilobytes and failed for the rest of the downloaded Facebook photos, whereas JPSeek failed to retrieve secret messages from all the downloaded Facebook photos, except for the ones retrieved from the second round. Repeating this experiment for the third time just to confirm our findings, we got the same results as the second round in which SilentEye was able to retrieve the secret message of 10 kilobytes in size from Facebook post photo and failed in the other cases, whereas JPSeek failed in all retrieval cases except for the 77 and 134 bytes messages for both Facebook post and profile photos.

For all the successful attempts at extracting secret messages from the first time, we tended to repeat the same experiment on those photos where we re-embedded the same secret messages again on those stego photos to investigate the possibility of retaining secret messages from stego photos that previously had contained a secret message. We conducted this experiment on the findings of the first experiment that had successful cases of message retrieval for three rounds. Those cases were SilentEye for all Facebook cover photos and Facebook post and profile photos when the message sizes were 77 bytes, 134 bytes, and 1 kilobyte. For the case of JPHS, we implemented the experiment in all Facebook cover photos since they were the only ones succeeded in retrieving a secret message from them. Finally, for the case of Secretbook, this tool succeeded in retrieving the secret messages of size 77 and 134 bytes from all Facebook cover, post, and profile photos. The findings from this experiment are that SilentEye and JPHS were able to extract the payloads successfully from all Facebook cover photos in all the three rounds. Also, SilentEye succeeded in retrieving a message in all the three rounds of Facebook profile photo when the message sizes were 77 bytes, 134 bytes, and 1 kilobyte. Moreover, SilentEye was able to retrieve messages from Facebook post photos when the message size was 134 bytes in all the three rounds, whereas it failed to retrieve secret messages from Facebook post photos of size 77 bytes in the first and the third rounds, and of size 1 kilobyte in the third round. For the case of the third tool, Secretbook succeeded in retrieving the secret messages of size 77 and 134 bytes from all Facebook cover, post, and profile photos in all the three rounds. We concluded from all those experiments that we were able to enhance some of the failed cases of message retrievals by re-embedding messages again on them. Also, we proved that we could use stego photos as new carriers for new secret messages. Fig. 15 illustrates the workflow of the first experiment including the successful and failed cases and Table XIII demonstrates the findings of the successful and failed cases of experiment 1, steganography without Facebook processing, using SilentEye, JPHide and JPSeek (JPHS), and Secretbook steganography tools.

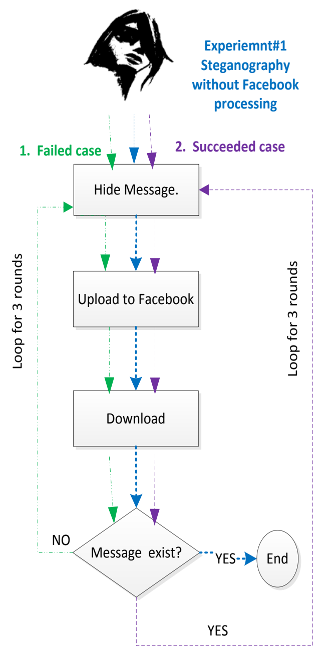


Figure 15. Experiment 1 of steganography without Facebook processing

TABLE XIII Steganography Findings of Successful and Failed Cases of Experiment 1

Steganography without Facebook Image Processing																
Tool		***FB Cover Photo***					***FB Post Photo***					***FB Profile Photo***				
		77 B	134 B	1 KB	10 KB	20 KB	77 B	134 B	1 KB	10 KB	20 KB	77 B	134 B	1 KB	10 KB	20 KB
SilentEye	*Initial*	✓	✓	✓	✓	✓	✓	✓	✓	×	×	✓	✓	✓	×	×
	Round1	✓	✓	✓	✓	✓	×	✓	✓	×	N/A	✓	✓	✓	×	×
	Round2	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓	×	×
	Round3	✓	✓	✓	✓	✓	×	✓	×	✓		✓	✓	✓	×	×
JPHS	*Initial*	✓	✓	✓	✓	✓	×	×	×	×	×	×	×	×	×	×
	Round1	✓	✓	✓	✓	✓	✓	✓	×	×	×	✓	✓	×	×	×
	Round2	✓	✓	✓	✓	✓	✓	✓	×	×	×	✓	✓	×	×	×
	Round3	✓	✓	✓	✓	✓	✓	✓	×	×	×	✓	✓	×	×	×
Secretbook	*Initial*	✓	✓	N/A	N/A	N/A	✓	✓	N/A	N/A	N/A	✓	✓	N/A	N/A	N/A
	Round1	✓	✓				✓	✓				✓	✓			
	Round2	✓	✓				✓	✓				✓	✓			
	Round3	✓	✓				✓	✓				✓	✓			

N/A: Not Applicable

B. *Experiment 2: Steganography with Facebook Image processing*

The second method we followed in our experiment was to impose Facebook image processing prior to applying steganography. This method is inspired by the work in [31]. It is about allowing the original photo to undergo Facebook image processing prior to applying steganography. To give a broader perspective about the mechanism of this method, we first uploaded the original photo to Facebook and downloaded the photo. By doing so, this allowed the photo to be processed by Facebook. Then, we attempted to use the steganography tools (SilentEye, JPHS, and Secretbook) to embed a secret message of different sizes and upload those stego images to Facebook again. Thereafter, we downloaded the stego images from Facebook and checked them using the first two tools to investigate the possibility of retrieving the secret messages. For the case of Secretbook, the checking was done while the photo was posted on Facebook; hence, Secretbook is a Google Chrome extension. We learned that using SilentEye tool, the secret message can be extracted successfully for all sizes in the case of Facebook cover photo. Checking for Facebook post photo, SilentEye was able to retrieve the secret message when the message sizes were 77 bytes, 134 bytes, 1 kilobyte and 10 kilobytes; however, SilentEye was not able to retrieve the secret message from Facebook post photo when the secret message size was 20 kilobytes. For the case of Facebook profile photo, SilentEye was able to extract the secret message in all the different sizes assigned except for the message that has a size of 20 kilobytes, as it failed in the message retrieval of that size. On the other hand, using the JPHide and JPSeek tool, JPSeek was able to retrieve the secret message of all sizes for the case of Facebook cover photo and failed in all secret message sizes for the case of Facebook post and profile photos except for the message size of 77 and 134 bytes. Moreover, Secretbook succeeded in retrieving the secret messages of size 77 and 134 bytes from all Facebook cover, post, and profile photos. From this experiment, we concluded that applying Facebook processing on photos prior to using steganography provides better results; specifically, SilentEye tool was able to retrieve all messages from all cases except when the message size was 20 Kilobytes for both post and profile photos. JPHS succeeded in all cases except when the message size was 1, 10, and 20 kilobytes for both Facebook post and profile photos. Secretbook succeeded in all the applicable cases (77 bytes and 134 bytes). Fig. 16 illustrates the workflow of this experiment and Table XIV demonstrates our findings.

TABLE XIV Findings of Experiment 2 Steganography with Facebook processing

Steganography with Facebook Image Processing															
Tool	***FB Cover Photo***					***FB Post Photo***					***FB Profile Photo***				
	77 B	*134 B*	1 KB	10 KB	20 KB	77 B	134 B	1 KB	10 KB	20 KB	77 B	134 B	1 KB	10 KB	20 KB
SilentEye	✓	✓	✓	✓	✓	✓	✓	✓	✓	×	✓	✓	✓	✓	×
JPHS	✓	✓	✓	✓	✓	✓	✓	×	×	×	✓	✓	×	×	×
Secretbook	✓	✓	N/A	N/A	N/A	✓	✓	N/A	N/A	N/A	✓	✓	N/A	N/A	N/A

N/A: Not Applicable

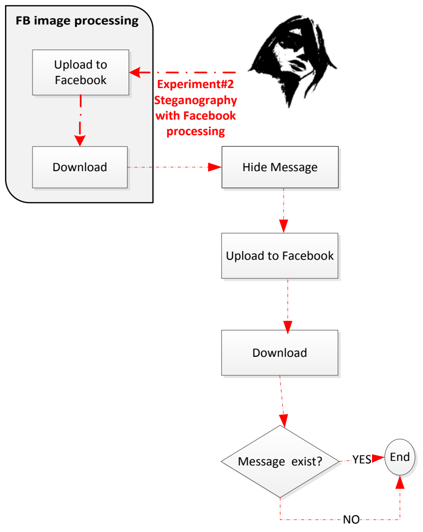


Figure 16. Experiment 2 of steganography with Facebook (FB) processing

C. *Steganography Persistency*

In this section, we describe our attempts to examine the persistency of the secret messages in photos downloaded from Facebook from which, previously, secret messages were successfully extracted; the method will check the results when we uploaded those stego photos several times on Facebook and then downloaded them. The question then arises: will steganography tools still be able to extract the same secret messages from those photos or not? We have applied this method to both findings from experiments 1 and 2. To elaborate more about the mechanism of this method, we at first checked all the successful attempts of SilentEye at retrieving secret messages from experiment 1, which were Facebook cover photos in all secret message sizes and Facebook post and profile photos when the message sizes were 77 bytes, 134 bytes, and 1 kilobyte. Then we tried to upload those photos to Facebook again and checked if the old secret message could still be extracted successfully or not. We repeated this experiment three times. The findings indicated that for all the three rounds, all Facebook cover photos were able to maintain the secret messages on them even if the same stego photos were uploaded to Facebook several times. While for the case of Facebook post photos that have a message size of 77 bytes and 1 kilobyte, the secret message survived Facebook image processing on the first uploading round. Moreover, for the Facebook profile photo that has a message size of 1 kilobyte, the secret message survived and could be extracted from the first round of the experiment, while for the second round, the secret message could not be extracted. On the other hand, based on SilentEye's findings from experiment 2, the secret messages were successfully extracted from all of the downloaded Facebook photos except from Facebook post and profile photos that have a message size of 20 kilobytes. Uploading those photos that have successful secret message retrievals again to Facebook, we discovered that all cover photos had maintained the secret messages on them for the whole three rounds. While for Facebook post and profile photos, only when the message sizes were 77 bytes, 134 bytes, and 1 kilobyte, the persistency of the secret messages maintained for 2, 2, and 1 round respectively. On the other hand, repeating the same experiment using JPHS tool, we found that all Facebook cover photos maintained the secret messages on them for all the three rounds and for both findings of experiments 1 and 2. Moreover, Facebook post and profile photos of sizes 77 and 134 bytes using JPHS maintained the secret messages on them for all the three rounds, except for the case of Facebook profile photo of size 77 bytes, in which the messages were maintained for only 2 rounds. For the case of Secretbook, we were able to maintain the embedded secret messages in all the three rounds; however, some of the characters of the secret messages were altered. More details about Secretbook case are discussed in the following paragraph. From this experiment, we concluded that stego messages have the possibility to be retained again for several times even if the photos encountered Facebook processing more than one time. We also concluded that the persistency of secret messages is higher when applying this method on the findings of experiment 2, which is steganography with Facebook processing, than if applying it on the findings of the first steganography experiment. Tables XV and XVI demonstrate the persistency findings from experiments 1 and 2 respectively.

Performing a steganography persistency experiment using Secretbook, we recorded a set of observations regarding the retrieved embedded message. As we embedded a message size of 134 bytes on the original photo and uploaded it to Facebook several times, we found that some of the characters of the retrieved message had either been altered, removed, inserted, changed, or undergone a combination of different alteration processes. This experiment directly linked us to our first preliminary experiments where we performed the binary comparisons of each consecutive uploaded photo to check the number of blocks of bytes removed, inserted, and changed between the uploaded photos. Fig. 17 demonstrates our observation regarding the altered characters in the embedded message in Facebook cover photo. The following section discusses our attempt to perform nested steganography.

TABLE XV Steganography Persistency based on Experiment 1

Steganography Persistency																
Tool		FB Cover Photo					FB Post Photo					FB Profile Photo				
		77 B	134 B	1 KB	10 KB	20 KB	77 B	134 B	1K B	10 KB	20 KB	77 B	134 B	1K B	10 KB	20 KB
SilentEye	Initial	✓	✓	✓	✓	✓	✓	✓	✓	×	×	✓	✓	✓	×	×
	Round1	✓	✓	✓	✓	✓	✓	×	✓			×	×	✓		
	Round2	✓	✓	✓	✓	✓	×		×					×		
	Round3	✓	✓	✓	✓	✓										
JPHS	Initial	✓	✓	✓	✓	✓	×	×	×	×	×	×	×	×	×	×
	Round1	✓	✓	✓	✓	✓										
	Round2	✓	✓	✓	✓	✓										
	Round3	✓	✓	✓	✓	✓										
Secretbook	Initial	✓	✓	N/A	N/A	N/A	✓∗	✓∗	N/A	N/A	N/A	✓∗	✓∗	N/A	N/A	N/A
	Round1	✓	✓				✓∗	✓∗				✓∗	✓∗			
	Round2	✓	✓				✓∗	✓∗				✓∗	✓∗			
	Round3	✓	✓				✓∗	✓∗				✓∗	✓∗			

N/A: Not Applicable, ✓∗: Altered Retrieved Message

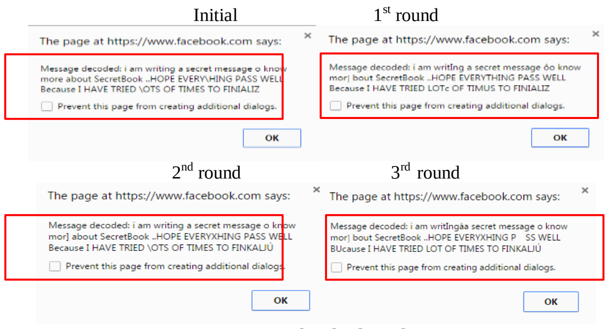
TABLE XVI Steganography Persistency based on Experiment 2

Steganography Persistency																
Tool		FB Cover Photo					FB Post Photo					FB Profile Photo				
		77 B	134 B	1 KB	10 KB	20 KB	77 B	134 B	1 KB	10 KB	20 KB	77 B	134 B	1 KB	10 KB	20 KB
SilentEye	Initial	✓	✓	✓	✓	✓	✓	✓	✓	✓	×	✓	✓	✓	✓	×
	Round 1	✓	✓	✓	✓	✓	✓	✓	✓	×		✓	✓	✓	×	
	Round 2	✓	✓	✓	✓	✓	✓	✓	×			✓	✓	×		
	Round 3	✓	✓	✓	✓	✓	×	×				×	×			
JPHS	Initial	✓	✓	✓	✓	✓	✓	✓	×	×	×	✓	✓	×	×	×
	Round 1	✓	✓	✓	✓	✓	✓	✓				✓	✓			
	Round 2	✓	✓	✓	✓	✓	✓	✓				✓	✓			
	Round 3	✓	✓	✓	✓	✓	✓	✓				×	✓			
Secretbook	Initial	✓	✓	N/A	N/A	N/A	✓∗	✓∗	N/A	N/A	N/A	✓∗	✓∗	N/A	N/A	N/A
	Round 1	✓	✓				✓∗	✓∗				✓∗	✓∗			
	Round 2	✓	✓				✓∗	✓∗				✓∗	✓∗			
	Round 3	✓	✓				✓∗	✓∗				✓∗	✓∗			

N/A: Not Applicable, ✓∗: Altered Retrieved Message

The embedded secret message

" i am writing a secret message to know more about SecretBook ..HOPE EVERYTHING PASS WELL Because I HAVE TRIED LOTS OF TIMES TO FINIALIZ"



Initial:
The page at https://www.facebook.com says:
Message decoded: i am writing a secret message o know more about SecretBook ..HOPE EVERY\HING PASS WELL Because I HAVE TRIED \OTS OF TIMES TO FINIALIZ
Prevent this page from creating additional dialogs.

1st round:
The page at https://www.facebook.com says:
Message decoded: i am writIng a secret message ôo know mor] bout SecretBook ..HOPE EVERYTHING PASS WELL Because I HAVE TRIED LOTc OF TIMUS TO FINIALIZ
Prevent this page from creating additional dialogs.

2nd round:
The page at https://www.facebook.com says:
Message decoded: i am writing a secret message o know mor] about SecretBook ..HOPE EVERYXHING PASS WELL Because I HAVE TRIED \OTS OF TIMES TO FINKALJÚ
Prevent this page from creating additional dialogs.

3rd round:
The page at https://www.facebook.com says:
Message decoded: i am writIngàa secret message o know mor] bout SecretBook ..HOPE EVERYXHING P SS WELL BUcause I HAVE TRIED LOT OF TIMES TO FINKALJÚ
Prevent this page from creating additional dialogs.

Figure 17. Secretbook altered message

D. *Nested Steganography*

In this section, we describe our attempts to perform nested steganography, embedding secret messages one over the other in Facebook cover photo category. The experiment involves all three of the tools (SilentEye, JPHide and JPSeek (JPHS), and Secretbook) conducted on a processed Facebook cover photo. We selected Facebook cover photo specifically for this experiment because, according to all our previous steganography experiments, Facebook cover photo is the only photo category in Facebook that led to a successful outcome in retrieving and maintaining the secret message in all the cases tested compared to Facebook post or profile photo. Furthermore, from our previous persistency experiment, we found that Facebook cover photo has a higher persistency in preserving the stego message than other categories; therefore, selecting Facebook cover photo for this experiment is the perfect choice, especially if the selected photo is preprocessed in advance by Facebook image processing. Hence, based on our second steganography experiment (steganography with Facebook image processing), the outcomes achieved are better than applying steganography without Facebook processing. The purpose of this experiment was to investigate the secret message identity if we were to embed two secret messages respectively in the same photo to indicate which message of these two messages is going to be retrieved and based on which tool is selected. The size of the secret message selected was 134 bytes; hence this is the size that matches with all three of the tools especially for Secretbook, as it is the only maximum size accepted for the original photo designed for Secretbook. We prepared two different secret messages of the same size (134 bytes) to perform the experiment. We started this experiment by uploading the original photo to Facebook as a cover photo. Downloading the previous uploaded photo, we embedded the first secret message using the three tools and uploaded them to Facebook. As we downloaded the first stego photos from Facebook, we confirmed the existence of the first message embedded using the three tools and we attempted to embed the second message on the download tested stego photos using the same tools. Uploading these stego photos, which hold two stego messages on them, to Facebook as a cover photo and downloading them, we used the three steganography tools to check which message of the two embedded ones is retrieved. We found that all three of the tools retrieved only the second embedded message; hence, the first stego message got overwritten by the second one. We concluded from this experiment that performing nested steganography always leads to retrieving the last embedded message. Table XVII demonstrates our findings of this experiment. The following section covers the comparison of the three selected steganography tools.

TABLE XVII Nested Steganography on Facebook Cover Photo

Nested Steganography		
Facebook Cover Photo		
Tool	***Retrieve 1ST Message***	***Retrieve 2ND Message***
SilentEye	×	✓
JPHS	×	✓
Secretbook	×	✓

VII. STEGANOGRAPHY TOOLS' COMPARISONS

In this section, we attempt to compare the steganography tools employed in our experiments. Through this research, we have conducted experiments on several scenarios or methods for applying steganography on Facebook. We have used three tools throughout these experiments, which are SilentEye, JPHide and JPSeek (JPHS), and Secretbook. We have tested the ability of SilentEye and JPHide and JPSeek tools to recover the embedded secret messages prior to uploading to Facebook and then we proceeded with steganography experiments through Facebook. Our observations during the embedding process using these tools are illustrated in Table XVI below. We noticed that using SilentEye tool to embed secret messages of different sizes (77 Bytes, 134 Bytes, 1 KB, 10 KB, and 20 KB) will somehow alter the visible features of the original photo. This is especially the case when the image white background showed some dots that did not exist previously on the original photo; and as we increase the size of the embedded secret message, the dots in the background increase too and the photo becomes more visually suspicious for steganography.

For the case of JPHide and JPSeek (JPHS), when embedding the secret message of different sizes on the original photo, we have noticed that the visible features of the stego image remained approximately the same as the original photo and the stego image was not suspicious for steganography to the naked eye. For Secretbook, the visual features of the stego image remained approximately the same as the original photo. Therefore, based on our observation and the experiment done on [2] and [33], we concluded that SilentEye generates significant artifacts on the stego image that disclose the steganography use with all different message sizes, whereas JPHide and JPSeek avoid such a disclosure. Out of the three tools, we concluded that the message retrievals using SilentEye tool provided better results than JPHS tool. Moreover, based on our observations, we found that the quality of stego photos in terms of image similarity to the original photo was high with Secretbook and JPHide and JPSeek (JPHS) and was low with SilentEye. To provide accurate results about our observations for the stego images' quality, we calculated peak signal to noise ratio (PSNR) since it is used as a quality measure for stego images. PSNR measures the similarity between two images and it is measured in decibels (db). A large PSNR value reflects a high-quality image, which indicates that both the original photo and the stego photo are very similar to each other [41][42]. The mathematical representation of PSNR is as follows:

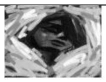
$$\text{PSNR} = 10 \log \frac{(255)^2}{MSE} \quad (1)$$

Where the MSE (Mean Squared Error)

$$\text{MSE} = \frac{1}{MN} \sum_{I=1}^{M} \sum_{J=1}^{N} (X_{i,j} - X'_{i,j})^2 \quad (2)$$

Where X represents the matrix data of the original photo, X' represents the matrix data of the stego image, M represents the numbers of rows of pixels of the image and i represents the index of that row, N represents the number of columns of pixels of the image and j represents the index of that column. We used Matlab (version: R2010a) as a tool to calculate PSNR for the photos used in our steganography experiments. We found that PSNR value between the original photo and the stego images from the three steganography tools was high in JPHS, medium in Secretbook, and low in SilentEye, which supports our observations. We calculated PSNR for all stego photos of the three tools in all the sizes tested. Table XVIII illustrates the findings of the steganography tools comparison displaying the stego photos, the size, the dimension and the PSNR values. Furthermore, we repeated the same imperceptibility test through calculating PSNR using a different photo to confirm our findings. We selected a standard test image for Lenna of size 51,403 bytes and with a dimension of 480 × 480. We embedded a secret message of size 27 bytes using the three tools as this size is accepted with Secretbook. The results of this experiment confirmed our previous findings. Table XIX illustrates the PSNR findings of Lenna image. Furthermore, the stego photos are analyzed by the histogram analysis. We found that the stego images that contain low payload capacity show minimum changes in the histograms compared to the original photo histogram which makes it difficult to infer the existence of the secret messages. Fig. 18 illustrates Lenna stego photos that contain a secret message of 27 bytes using the three tools and Fig. 19 illustrates the histograms of those photos.

TABLE XVIII Stego Photos Using Different Steganography Tools and PSNR values

Tool	*Original*	134 *bytes*	1 KB	10 KB	20 KB
SilentEye	49,607 Bytes 705x856 Pixels	53,698 Bytes 705x856 Pixels 34.0236 db	53,741 Bytes 705x856 Pixels 34.0660 db	55,627 Bytes 705x856 Pixels 33.4426 db	56,271 Bytes 705x856 Pixels 33.2330 db
JPHS	49,607 Bytes 705x856 Pixels	68,267 Bytes 705x856 Pixels 63.8162 db	68,318 Bytes 705x856 Pixels 64.0423 db	68,307 Bytes 705x856 Pixels 63.9493 db	68,298 Bytes 705x856 Pixels 63.9279 db
Secretbook	248,446 Bytes 960x720 Pixels	 103,087 Bytes 960x720 Pixels 35.0515 db	N/A	N/A	N/A

N/A: Not Applicable

Figure 18. Lenna test image

TABLE XIX PSNR Values for Lenna Stego Photo with Payload Capacity of 27 B

Tool	PSNR
SilentEye	34.6515 db
JPHS	54.893 db
Secretbook	39.23 db

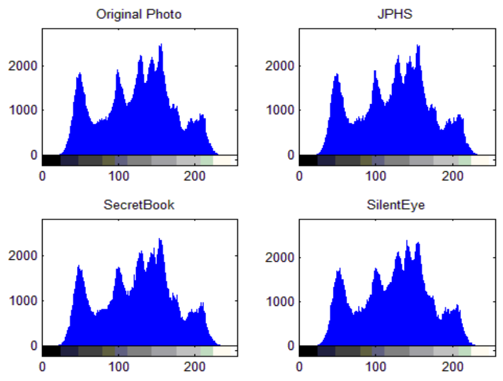


Figure 19. Histograms of Lenna photo and the stego photos of 27 bytes using the 3 tools

VIII. DISCUSSION

In this section, we discuss and highlight the major conclusions of our paper. Analyzing our findings from all the experiments that we have implemented on Facebook with all the methods or scenarios that we have employed, we conclude that Facebook image processing performed on Facebook cover photo, Facebook post photo, and Facebook profile photo are different from each other; this is clearly demonstrated by our findings in all the methods followed in which the findings of any of the same experiment look different for each Facebook photo category, even though we used the same original photo in all the experiments, the same tools, and the same secret message sizes. Also, based on our experiments in all of the scenarios, we conclude that applying steganography to Facebook cover photos provides better results as it is supported by the work in [31]. In addition, throughout all our experiments, we proved that steganography in Facebook cover photos, in terms of successfully extracting the embedded messages of all the different sizes tested and maintaining the secret messages, performed successful outcomes more consistently than applying steganography on Facebook post or profile photos. More precisely, in our preliminary experiments, we demonstrated the binary comparison of Facebook cover photos indicating a consistent number of byte block changed, at the rate of exactly one change in every uploading try, which to our knowledge explains the causes of successful message retrieval and the higher rate of steganography persistency in Facebook cover photos than in Facebook post or profile photos. Also, from our preliminary experiments of uploading the original photo of size 49,607 bytes to Facebook for 50 times, we determined that Facebook processing for a cover photo increases the size of the uploaded photo to maintain a new size of 66,305 bytes for all of the 50 tries, while processing for the post and profile photo decreases the original image size to 46,891 bytes; thus, this allows for more capacity in the cover photo to hide and maintain the secret message than in post and profile photos. For the case of steganography on Facebook post photos and Facebook profile photos using SilentEye tool, we indicate that experiment 2, which consists of performing Facebook image processing on the original photo prior to uploading it to Facebook, provides better results for message retrieval than experiment 1, which consists of uploading the stego photo directly to Facebook without performing any prior processing. To elaborate further, in the first experiment we were able to extract the hidden messages from the stego images of Facebook post and profile photos only when the size of the image was up to 1 KB, whereas after conducting experiment 2 and applying Facebook image processing prior to uploading stego images to Facebook, we were able to extract embedded messages that have a size of up to 10 KB. However, JPSeek was unable to retrieve any messages from Facebook post photos and Facebook profile photos in experiment 1, but in experiment 2 it succeeded when the message size was 77 bytes or less than 1 KB. Moreover, with the use of the method of re-embedding the secret message on Facebook photos that failed to extract a secret message from them, we were able to enhance some of the failed results of the first steganography experiment; hence, previously we were not able to extract messages that hold a size of 10 KB from Facebook post photos. Furthermore, after re-embedding the 10-KB secret message again on the previous failed Facebook post photo and uploading it to Facebook, in the second round of the experiment, we succeeded in retrieving the 10-KB embedded message. Additionally, based on our experiment of checking the persistency of the embedded secret message regardless of the number of times the photo uploaded to Facebook, we conclude that stego Facebook cover photos have a higher persistency of preserving the secret messages on them regardless of the number of times the same photos are uploaded to Facebook as compared to Facebook profile and post photos. In addition, we calculated the PSNR values for all the stego photos and found that JPHS tool provided the best quality of stego photos compared to other tested tools; moreover, we illustrated the photos histograms for a test image that held 27 bytes secret message embedded using all the three tools.

IX. CONCLUSION

Steganography is the art and science of hiding secret messages on another object. Several methods for using Facebook for image steganography exist. In this research, we proved that we can apply image steganography on Facebook cover photos with secret messages up to 20 KB using SilentEye and JPHide and JPSeek tools, and with Facebook post photos and Facebook profile photos with secret messages up to 10 KB using SilentEye tool. Furthermore, we illustrated that Facebook image processing to the uploaded photos through the mobile application is more intense than processing through a personal computer (PC) in which major changes in the photos perceptibility, block of bytes, and sizes occurred when we upload photos using the Facebook mobile application. In addition, we demonstrated the Facebook processing schemes on the uploaded photos from different platforms, operating systems, and browsers, and we provided the binary comparisons of the subsequent uploaded photos. Through the research, we were able to enhance some of the failed cases of message retrievals for steganography by re-embedding messages again on them. We proved that we could use stego photos as new carriers for new secret messages. We also concluded that the persistency for secret messages is higher when it is applied to the findings of experiment 2, which is steganography with Facebook processing, than if it is applied to the findings of the first steganography experiment. Furthermore, we proved that applying nested steganography always leads to successful retrieval of the last embedded message. Moreover, we showed that the quality of stego photos in terms of image similarity with the original photo was high with JPHS tool, was also good with Secretbook, and was low with SilentEye; however, message retrievals and the payload capacity handled using SilentEye tool provides better results than JPHS and Secretbook tools.

REFERENCES

[1] N. Provos and P. Honeyman, "Hide and seek: An introduction to steganography," *Security & Privacy, IEEE,* vol. 1, pp. 32-44, 2003.

[2] A. Chee, "Steganographic techniques on social media: investigation guidelines," Thesis, School of Computing and Mathematical Science Auckland University of Technology, 2013.

[3] D. Boyd and N. Ellison, "Social network sites: definition, history, and scholarship," *IEEE Engineering Management Review,* vol. 3, pp. 16-31, 2010.

[4] Stastista. "Leading social networks worldwide as of August 2015, ranked by number of active users (in millions)". Available: http://www.statista.com/statistics/272014/global-social-networks-ranked-by-number-of-users/. Accessed October 1, 2015.

[5] *Facebook. "Facebook Mission"*. Available: https://www.facebook.com/facebook/info?tab=page_info. Accessed October 30, 2015.

[6] A. N. Joinson, "Looking at, looking up or keeping up with people?: motives and use of facebook," in *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, 2008, pp. 1027-1036.

[7] A. Kumar and K. Pooja, "Steganography-A data hiding technique," *International Journal of Computer Applications,* vol. 9, pp. 19-23, 2010.

[8] S. Sharma and U. Kumar, " Steganography: The Art of Covert Communication," *Multidisciplinary Journal of Research in Engineering and Technology,* vol. 2, pp. 669-675, 2015.

[9] P. Richer, "Steganalysis: Detecting hidden information with computer forensic analysis," *SANS/GIAC Practical Assignment for GSEC Certification, SANS Institute,* vol. 6, 2003.

[10] T. Morkel, J. H. Eloff, and M. S. Olivier, "An overview of image steganography," in *ISSA*, pp. 1-11, 2005.

[11] P. Khare, J. Singh, and M. Tiwari, "Digital Image Steganography," *Journal of Engineering Research and Studies,* vol. 2, pp. 101-104, 2011.

[12] U. Rizwan and H. F. Ahmed, "A New Approach in Steganography using different Algorithms and Applying Randomization Concept," *International Journal of Advanced Research in Computer and Communication Engineering,* vol. 1, 2012.

[13] N. F. Johnson and S. Jajodia, "Exploring steganography: Seeing the unseen," *Computer,* vol. 31, pp. 26-34, 1998.

[14] A. Cheddad, J. Condell, K. Curran, and P. Mc Kevitt, "Digital image steganography: Survey and analysis of current methods," *Signal processing,* vol. 90, pp. 727-752, 2010.

[15] M. Hussain and M. Hussain, "A survey of image steganography techniques," 2013.

[16] K. Rabah, "Steganography-the art of hiding data," *Information Technology Journal,* vol. 3, pp. 245-269, 2004.

[17] M. B. Pope, M. Warkentin, E. Bekkering, and M. B. Schmidt, "Digital Steganography—An Introduction to Techniques and Tools," *Communications of the Association for Information Systems,* vol. 30, p. 22, 2012.

[18] E. Walia, P. Jain, and N. Navdeep, "An analysis of LSB & DCT based steganography," *Global Journal of Computer Science and Technology,* vol. 10, 2010.

[19] D. Neeta, K. Snehal, and D. Jacobs, "Implementation of LSB steganography and its evaluation for various bits," in *Digital Information Management, 2006 1st International Conference on*, 2006, pp. 173-178.

[20] T. Morkel, J. H. Eloff, and M. S. Olivier, "An overview of image steganography," in *ISSA*, 2005, pp. 1-11.

[21] S. Goel, A. Rana, and M. Kaur, "A Review of Comparison Techniques of Image Steganography," *Global Journal of Computer Science and Technology,* vol. 13, 2013.

[22] K. M. Sullivan, "Image Steganalysis: Hunting & Escaping," University of California Santa Barbara, 2005.

[23] R. J. Anderson and F. A. Petitcolas, "On the limits of steganography," *Selected Areas in Communications, IEEE Journal on,* vol. 16, pp. 474-481, 1998.

[24] R. Chandramouli, M. Kharrazi, and N. Memon, "Image steganography and steganalysis: Concepts and practice," in *Digital Watermarking*, ed: Springer, 2004, pp. 35-49.

[25] G. C. Kessler, "An overview of steganography for the computer forensics examiner," *Forensic Science Communications,* vol. 6, pp. 1-27, 2004.

[26] A. Castiglione, G. Cattaneo, and A. De Santis, "A forensic analysis of images on online social networks," in *Intelligent Networking and Collaborative Systems (INCoS), 2011 Third International Conference on*, 2011, pp. 679-684.

[27] S. Nagaraja, A. Houmansadr, P. Piyawongwisal, V. Singh, P. Agarwal, and N. Borisov, "Stegobot: a covert social network botnet," *Information Hiding,* pp. 299-313, 2011.

[28] K. Solanki, A. Sarkar, and B. Manjunath, "YASS: Yet another steganographic scheme that resists blind steganalysis," in *Information Hiding*, 2007, pp. 16-31.

[29] O. Moore, "Secret Communication on Facebook implemented with Browser-Based Steganography," Thesis, Department of Computer Science, Oxford University, 2013.

[30] R. Beckhusen. (April 10, 2013, Secretbook lets you encode hidden messages in your Facebook pics. Available: http://www.wired.com/2013/04/secretbook/. Accessed October 10, 2015.

[31] N. D. Amsden, L. Chen, and X. Yuan, "Transmitting hidden information using steganography via Facebook," in *Computing, Communication and Networking Technologies (ICCCNT), 2014 International Conference on*, 2014, pp. 1-7.

[32] J. Hiney, T. Dakve, K. Szczypiorski, and K. Gaj, "Using Facebook for Image Steganography," *arXiv preprint arXiv:1506.02071,* 2015.

[33] B. Cusack and A. Chee, "Steganographic Checks In Digital Forensic Investigation: A Social Networking Case," 2013.

[34] A. Chorein, "SilentEye-Steganography is yours," 2008.

[35] A. Latham. *"Steganography"*. Available: http://linux01.gwdg.de/~alatham/stego.html. Accessed 10, Nov. 2015.

[36] O. Moore. *"An Error-Resistant Steganography Algorithm for Communicating Secretly on Facebook."* M.A. thesis, Oxford University, Britain, 2013.

[37] O. Moore. "Hide Secret Messages In Facebook Photos Using This New Chrome Extension". Available: http://www.owencampbellmoore.com/blog/2013/04/hide-secret-messages-in-facebook-photos-using-this-new-chrome-extension/. Accessed October 18, 2015.

[38] *Facebook, Inc. "How can I make sure that my photos display in the highest possible quality?"*. Available: https://www.facebook.com/help/266520536764594#How-can-I-make-sure-thatmy-photos-display-in-the-highest-possible-quality?. Accessed October 3, 2015.

[39] *Araxis, Ltd. "Araxis Merge"*. Available: http://www.araxis.com/about. Accessed October 4, 2015.

[40] *Chrome-Extension. "Secretbook Instructions"*. Available: chrome-extension://plglafijddgpenmohgiemalpcfgjjbph/installed.html. Accessed November 3, 2015.

[41] A. Al-Mohammad, "Steganography-based secret and reliable communications: Improving steganographic capacity and imperceptibility," Brunel University, School of Information Systems, Computing and Mathematics Theses, 2010.

[42] S. Hemalatha, U. D. Acharya, A. Renuka, and P. R. Kamath, "A secure and high capacity image steganography technique," *Signal & Image Processing,* vol. 4, p. 83, 2013.

AUTHORS PROFILE

Budoor Salem Edhah is a master student in Information Systems at Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia. She obtained her bachelor degree with first honor in Management Information Systems and her MBA degree from Dar Al Hekma University, Jeddah, Saudi Arabia. She held the position of Vice President of Students Government and MIS representative at Dar Al Hekma University. She completed the Certified Ethical Hacker Course with EC-Council. Her research interests include Information Security, Mobile Learning, and Business Process Reengineering.

Dr. Daniyal Alghazzawi obtained his Bachelor's degree with honor in Computer Science from King Abdulaziz University in 1999. He completed his master's degree and doctorate in the field of Computer Science at the University of Kansas at the United States in 2007. He also obtained another master's degree in Teaching and Leadership from University of Kansas in 2004 which helped him to develop his teaching and leadership skills. He also obtained the certificate of Management International Leadership (LMI) and has been the Head of the Information Systems department, Faculty of Computing and Information Technology for over five years during which he organized many workshops and international and domestic conferences. He is holding an honorary lecturer position at the University of Essex since 2010 and joined them in several researches in the field of smart environment. In 2012, he got promoted to associate professor as a consideration for his output in research field which exceeded currently 80 researches and books in the field of Intelligent Systems and Information Security. He also is the head of the Information Security Research Group at King Abdulaziz University and a reviewer for more than 20 international journals. His research interests include Smart e-Learning, Information Security, and Computational Intelligent.

Dr. Li Cheng graduated from Zhejiang University and obtained his Bachelor’s degrees in Industrial Automation and Computer Science. He obtained his master’s degree and doctorate in computer science from the University of Kansas. He has been holding positions as senior software developer and researcher at the University of Kansas for four years. He joined U.S. DoD Biotechnology High Performance Software Institute in 2009 and served as a senior software architect and a research scientist since then. He was selected as a “1000 talent plan” expert and joined Chinese Academy of Sciences (CAS) as a full professor in 2013. He has been serving as a committee member for the Chinese language processing society in Chinese Computer Federation. Currently, he is the head of a research group focusing on big data analytics at CAS. His research interests include data mining, machine learning, cloud computing, intelligent systems, artificial intelligence, natural language processing and bioinformatics.