# Integrating Smartphone Network Architecture and Data Security Techniques to Mitigate Sharp Practices in Non-Profit Organizations

Matimu Caswell Nkuna[1], Ebenezer Esenogho[1, 2], and Reolyn Heymann[1]

[1] Centre for Collaborative Digital Networks, Dept. of Electrical and Electronic Engineering Science, University of Johannesburg, Auckland Park, South Africa

[2] Center for Telecommunication, Dept. of Electrical and Electronic Engineering Science, University of Johannesburg, Auckland Park, South Africa

Email: ebenezere@uj.ac.za

***Abstract***—Contemporary research is geared towards integrating multidisciplinary domains and targeted to developing robust solutions. This implies that single-discipline solutions most times do not give the desired results. This study intends to apply this concept statement in mitigating the hydra-headed challenge facing the world today called sharp-practice (corruption) which some believe that it is worse than the current severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2). The sharp practice is a major challenge in the world today and a threat to the development of any country. Sharp practices range from higher offices of the state institutions to the small offices located in remote and rural local areas. Non-profit Organizations (NPOs) are also affected by sharp practices as the office-bearers of these institutions are constantly misappropriating donors' funds meant for these facilities for their selfish interests. State and private donors offer a large portion of their resources to NPOs to improve the standard of living of needy people in remote and rural areas. However, most NPOs sited in rural areas do not have systems that can be used to monitor and capture the management of resources. In this paper, we attempt to solve this challenge by proposing the integrating Smartphone Network Architecture (SMA) and Data Security Techniques (DST) to mitigate sharp practices in NPOs. In our approach, a smartphone-mobile app algorithm was developed using JavaScript. The app uses the Least Significant Bits (LSB) method to secretly embed a Date Stamp on a PNG image whenever an event is captured by a secret camera at the NPO's center. The results showed that the LSB method is the most suitable technique to embed a small-scale secret data on the pixels of a PNG image file. The LSB technique does not require the original cover image for decoding the secret data hidden in the image over the network. From the study, these techniques have shown to improve accountability and alleviate corrupt practices in NPOs.

***Index-Terms***—Sharp practices, Smartphone, Data Security, Steganography, Least Significant Bits (LSB), Internet of Things (IoT), Android, Red Green Blue (RGB) NPOs.

## I. Introduction

Sharp practice may involve, but is not restricted to making deceptive declarations, threats, ignoring agreements, improperly using process, manipulating laydown protocols and procedures, or using other tricky and/or dishonourable means barely within or outside the law. It could also be described as a way of behaving in business that is dishonest but not illegal, however not encouraged.

This act could be in the form of misappropriation or diversion of state or donor funds meant for development of the communities. Most of the times these funds are meant for providing social amenities like pipe borne water, electricity, schools, food for kids in primary school and upkeep for the aged in the village. This consistent pattern of sharp practice may lead to discipline by a court or the state prosecution agent.

Sharp practices and corruption is a major problem in the world. It threatens the developing state of the economy in the country. It extends from the higher officers of the government institutions to the local area and remote offices. Office bearers are engaged in misusing resources meant for supporting the communities and the NPO's offices and in turn make false claims that they are delivering services to the people. This is done successfully because the regulatory gadget and information recording/capturing tools that are currently in place are not secured. Hence, it can be manipulated to suit the needs of the offenders, which are the office bearers in this context. Corruption and sharp practices hurts people and communities in the sense that it delays economic development, more especially in rural and remote areas. The resources that are allocated to bring development are being misused by the people who are in the distribution value chain. Media houses and blogs are publishing this kind of mismanagement with no concrete evidence. This is because there are no secured systems put in place to help monitor the distribution of resources and services to the people which will enable accountability.

Techniques such as steganography in connection with IoT technologies may be set up to help monitor mismanagement by office-bearers and hence assist in fighting corruption and sharp practices in NPOs operating in rural and remote areas. This kind of integration is important because it works in the real-time and makes it

Manuscript received April 5, 2020; revised September 3, 2020.
doi:10.12720/jcm.15.10.755-767

relevant in the day to day monitoring of resources and services claimed by the fund distributors. Note its application is not restricted to NPOs alone. It can be applied to NGOs, or any situation that requires or warrants sending classified videos and audio for high-level investigation by the police, military, and state agencies like FBI, etc.

In this investigation, corruption and sharp practices happening in rural and remote NPOs are the main challenges being addressed. This becomes imperative because these facilities are not well regulated and monitored especially in remote locations. The government and other private donors are donating huge resources to these facility centres so that the well-being of the people (kids and aged) in need may be improved.

As such, solving this problem will be beneficial to the well-being of the people who are living in rural and remote areas. Hence, people will be able to have improved access to resources because of the improved regulatory measures that will be put in place for proper accountability. Even the state and private donors will be satisfied from solving this problem as it will mean that their resources will now be optimally utilized and distributed in a much better way and accessible by the needy people will be improved.

For emphasis, the aim of this study is to develop and implement a monitoring tool/system that will be used to counteract sharp practices and mismanagement of resources in NPOs located in rural communities. Most often office-bearers in rural NPOs makes false claims regarding the offering and accessibility of services that they render to orphan children and widows. Through this strategy, the reimbursement/claim forms that is filled and submitted to their donors (both state and private) are scrutinized through our developed system and hence regarded as fraudulent since there are evidence that they did not supply services to the children (orphans and widows). Investigation has revealed that they use the donated resources for their own selfish needs hence, the objective of this research is to design and implement an integrated solution. Our solution incorporates security features of steganography with an IoT platform such that the office bearer will be oblivious to the fact that she/he is been captured in real-time. Secondly, no router is installed in the premise but with the steganography algorithm app embedded in smartphone, events (Images and audio) are captured and encoded (encrypted) as text to the state or private donors. However, if the office bearer gets hold of the smartphone, all he/she will rather see is a text file with messages like "hello world" or any distracting message. By so doing that will help to counteract the claims by the fund's distributors at the NPO's. The key contributions of this work are:

- We propose a strategy that integrates image steganography techniques and IoT technologies that monitors and captures events that occurs in rural NPOs. These captured events at NPO's offices are coded like messages which when delivered to funds donor, it decode to the real captured images of what happen at the remote and rural centers of NPOs.
- We further develop an android mobile app using JavaScript to implement this strategy.
- We analyzed the use of Least Significant Bits (LSB) method to secretly embed a DateStamp on a PNG image whenever an event is captured at the NPO's center and the results showed that the LSB method is the most suitable technique to embed a small scale secret data on the pixels of a PNG image.
- We also found that the LSB technique do not require the original cover image for decoding the secret data hidden in the image. Hence reduce complexity of our system.
- From our test run and analysis, these techniques have shown to improve accountability and alleviate corrupt practices in NPOs and its allied institution to a large extent.

## II. Background

Steganography is derived from Greek words "stegos" which means "cover" and "grafia" meaning "writing" [1]. So, steganography means covered writing. It includes three elements which are the 'secret message', 'cover message' and the 'Stego key'. The secret message is the information that needs to be embedded in some media file (image, text or audio, etc.). The cover message is the carrier of the message such as text, video, image, or other media (digital). The Stego key is the one used to embed the message; this works according to the algorithm of hiding the message. The embedding algorithm is implemented to hide the secret message on the cover, the result is then called the Stego-object. The process of steganography is the encoding of a secret message at the sender and decoding at the receiver end. Fig. 1 below illustrates the basic processes of steganography [2]. At the sender end, the secret message is encrypted using an encryption key. The message is then embedded in the cover. Then the Stego-object is sent through a communication channel (an app via an IoT system) to the decoder on the receiver's end [3]. The receiver will be able to decode the secret message using the Stego key and the decryption key. For hiding secret information, there is a large variety of steganography techniques that can be used.

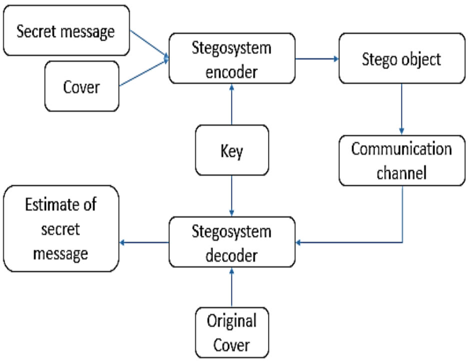


Fig. 1. A figure illustrating general Steganography processes

Some of these techniques are more complex than others, and they all have their strong and weak points. The techniques that are used depends on the application, where other applications may require absolute invisibility of the secret message and some may require a large secret message to be hidden. The different techniques are explained below [4]:

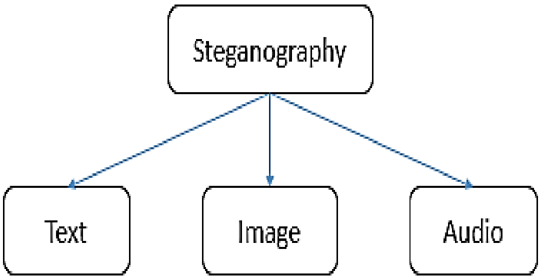


Fig. 2. Different types of steganography [8]

The text steganography can be accomplished by making alterations on a text file, by altering certain characters of a text element. These alterations should be reliably decodable even in a case where there is the presence of noise, but yet it should be indiscernible to the reader. So, the objective is to design coding methods that can achieve these objectives stated above [5]. This has challenges in the designing of the document marking techniques because the objectives (reliable decoding, minimum visible change etc.) are conflicting at some point. The file format of the document is a computer file of a type such as txt, TeX, PostScript2, @off etc. [6]. Then the image that the reader sees is generated from this file format. Three coding techniques are line-shift coding, feature coding and word-shift coding. These techniques can be jointly or separately implemented.

The image steganography technique is the most popular technique for hiding secret messages. It is divided into two main categories which are the Image Domain (ID) and the Transform Domain (TD). In the ID, the messages are embedded directly in the intensity of the image pixels, whereas in the TD, the image is first transformed to the frequency domain and then the secret messages are embedded in the image. Image steganography is the most frequently used technique because it is much easier to send an image file through the communication channel between the sender and the receiver end. There are three types of images: Red-Green-Blue (RGB), binary (Black White) and Grayscale images. The binary image uses one-bit value per pixel, and it is represented by 0 for black and 1 for a white pixel. A grayscale image has 8 bits value per pixel, and it is represented by 00,000,000 for a black pixel and 11,111,111 for a white pixel. Whereas the RGB images have 24 bits values per pixel and it is represented by (00000000, 00,000,000 and 00000000) for a black pixel and (11111111, 11111111, 11111111) for a white pixel [7]. RGB images are the most suitable image types to embed secret information because it contains a lot of information that helps in hiding the secret information. The bit changes in the image resolution do not affect the overall image quality and that helps in keeping the secret message more secured [8]. This method uses an image as a cover, and the cover is altered in noisy areas with a lot of colour variations. This is done to limit attention and avoid visible changes to the image file. Some most common algorithms are used to make these modifications in the image file, such as LSB (Least Significant Bit), masking, transformation, and the filtering methods. These algorithms are limited to different image types. Some of the techniques used to implement this method are LSB method, masking and filtering and the transform techniques. Audio steganography operates by embedding secret information into an audio signal that has been digitized [9]-[11]. The binary sequence of the audio file is slightly modified to embed the secret message. Some of the techniques of audio steganography are LSB coding and echo hiding. The Internet of Things is a system that is made up of different interrelated devices. These devices can be digital machines, objects (i.e. sensors), computing devices (i.e. Arduino UNO, Ethernet shields), mechanical machines and people, etc. [12]. The different devices have their respective Unique Identifiers (UIDs) and they can transfer data over some network protocol. The data transfer usually doesn't require any human-to-human or human-to-computer interaction.

In IoT, a thing can be anything ranging from just a simple sensor, a heart monitor implant, a biochip transponder or an advanced smart sensor to alert tire pressure, smoke, or any other natural factor. These devices are assigned different IP addresses to enable them to communicate over a given network protocol that has been set up [13]. The use of IoT is increasing and becoming more efficient in the recent days. The evolvement of IoT is from the convergence of the internet, micromechanical systems, wireless technologies, and micro services.

The methodology used in this paper follows: the engineering design methodology, which includes the design, implementation, and test phases. The design phase involves the use of IoT components and development of the app framework. The implementation stage comprises the developing app using java scripts, encrypting the messages (images/audios/videos) and integrating all the entire systems with the smartphone cameras and linking with Wi-Fi. The last stage is the testing of the developed system to show success in solving the problem. Note the scope of this work did not include rigorous mathematical analysis. However, will be critically investigated in our future work.

## III. Related Work

Securing data using image steganography is key and therefore, part of our concern in this paper is the security problem associated with IoT systems, especially when devices are exposed to the internet. Moreover, IoT devices with weak security levels and processing power, such as IP cameras lack confidentiality features and can be targets for hackers. Attackers can have a greater chance to figure out and intercept the on-going data transmission between the IoT devices [8]. To solve this problem of confidentiality in these IoT networks with devices that have weak security levels, a security scheme that involves connections to the internet and steganography is proposed. In this investigation, the IoT device that was considered was an IP camera. Sensitive information are transmitted between the IP camera and a home server, and to ensure secure data transmission, image steganography are used within the Local Area Network (LAN). This solution can provide more security in the sensitive data that are being transmitted such as the images of the user's faces in the LAN as the sensitive information will be hidden using image steganography. The face images of the users can be obtained by using the IP camera after which the images are sent to the server for authentication purposes. This scheme uses the inverted LSB image steganography. In the case of the home server sending any sensitive authentication information to the IoT device (IP camera in this case), the information will be encoded using the inverted LSB method to protect it from attackers. The figure below illustrates the proposed scheme [8]. The investigation in figure 3 as captured in [3] are similar to our research since it also uses IoT and image steganography to design a solution to the problem identified. However, the difference with this investigation is that we will not be using an IP camera since it can be seen installed in NPOs premises and this can be compromised. Instead, a smartphone app camera will be able to take pictures and embed certain secret information. This is a more convenient, reliable, and robust way of capturing sharp practices in NPOs. For quick understanding Fig. 3 shows the functional diagram of data security using steganography, IoT and simple flow.

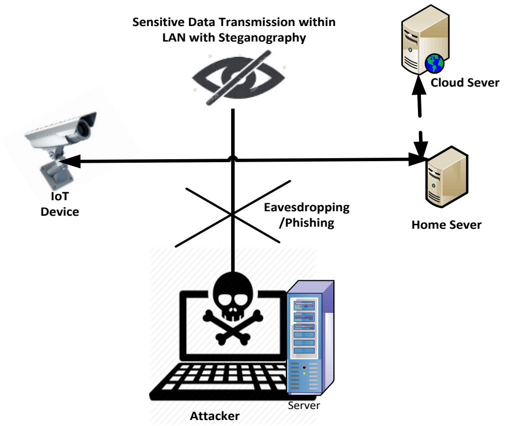


Fig. 3. Functional diagram of data security using steganography and IoT and Simple Flow.

## IV. The Generic Overview

This design uses a smartphone that connects the internet using a communication module. An app that capture pictures and embed some secret data in the picture without the knowledge of the user was developed using an android SDK development platform. The app takes a picture and embed the secret data on the pixels of the picture by using an encoding algorithm and then it will save the picture in the internal storage of the smart phone's picture directory. The picture will then be sent via a communication channel to the end-user. A decoding algorithm was implemented to decode and recover the original secret data for validation. The generic schematic diagram of the high-end design is shown in Fig. 4.

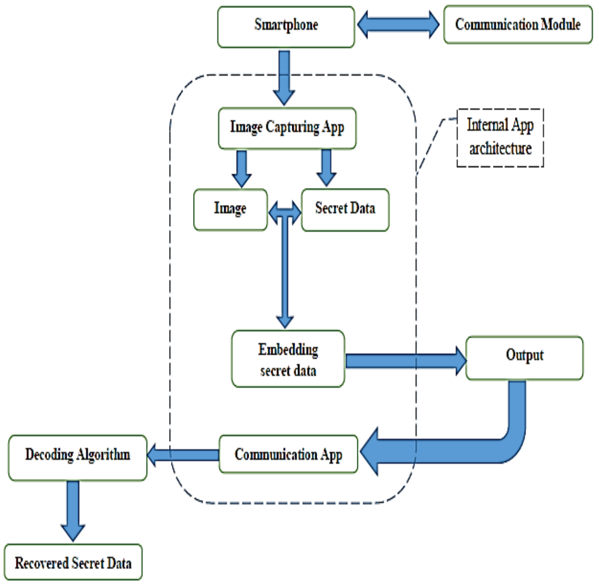


Fig. 4. Generic view of the design

### A. *Communication Module*

The communication module will enable the smartphone to connect to the internet. This module will maximize the internet speed rather than using a data connection of the smartphone. Mobile data connections are often slow especially in rural areas where there is limited network reception, hence the communication module will be selected such that it enables fast and reliable internet connectivity. Some of the application modules are ESP8266 Wi-Fi module, Mobile GSM module (3G/4G/LTE), GSM SIM900A Modem, USB to ESP8266 Wi-Fi module, Sigfox communication module, RF Module SI4463, Bluetooth Smart module, and IEEE802.15.4/ZigBee module. For this implementing this subsystem, it is preferred that the module to be used will cover a wide radius of operation so that even if the user is moving from one point to another, there will still be coverage to connect to the internet.

Hence, the ESP8266 Wi-Fi module is the more cost-effective communication module of all the modules listed above that are available at the local supplier with wider range of coverage as compared to the other communication modules [14], [15]. Considering all the factors described above, the best choice of the communication module to be used for the application of this project is the ESP8266 Wi-Fi module. This module works very similar to any other typical 802.11g Wi-Fi router.

### *B. Smartphone*

Some of the most common smartphone operating systems alternatives that are available include Android OS, Windows Phone, Blackberry OS, iPhone OS/iOS and Symbian OS. The Android mobile OS is an open and free software stack including an operating system compiled by Google. This Android OS is very common in mobile devices including smartphone. The Android OS has updates and different version names (SDKs) such as Android 4.0 (Ice Cream Sandwich), Android 4.4 (KitKat), Android 5.0 (Lollipop), Android 6.0 (Marshmallow) etc. The updates come with new enhancements and improvements which allows functionality of mobile applications. For any mobile application, there is a specific minimum SDK version required for it to work in an android device. The minimum SDK version is set up according to the needs of the application developer. According to statistics, in the periods of 2016Q1 until 2017Q1 more people use mobile devices that are operated by android OS. Android mobile phones are cost-friendly, and it can cost as low as $20 to purchase the mobile device. [14]. Windows phones use the Windows operating system and Microsoft is the company behind the production of these devices and the OS software. The mobile OS is compiled from the Windows CE 5.2 kernel. The Windows OS also has updates that come with improved features and functionality, but it is not open source and it is not free. Another alternative is the Blackberry OS mobile devices. This OS is also not free, and it is developed by Research in Motion for use in the Blackberry mobile devices. This alternative had a decline in customer use from the period of 2016Q1 until 2017Q1, hence this shows that fewer people use mobile devices powered by this OS. Platforms of developing apps supported by this OS are expensive and not widely available [15]. The Apple's iPhone OS was compiled for operation on its iPhone mobile devices such as the iPhone, iPad, iPod and iPad 2. IOS is only available on Apple's own manufactured devices. This OS is not licensed for third party hardware. The app developers for iOS are strictly employees of Apple. These mobile devices are typically expensive as compared to the other commonly used mobile devices brands. The Symbian OS is targeted for devices that has high-level PIM functionality. This OS combines middleware with wireless communications through the integration of Java and PIM functionality. The Symbian OS is not an open-source development project .

From the information stated below and the data in Table I, it can be concluded that the Android OS is the best choice of all the other smartphone alternatives. The Android OS is free which implies that it is more cost-friendly, and it is open source. The Android SDK development platforms are widely available and free open-source software. Hence the Android OS is chosen for this design.

TABLE I: SHOWS WORLDWIDE MOBILE SMARTPHONES SHIPMENT BASED ON THEIR OS ACCEPTABILITY

| Year | Android (%) | iOS (%) | Others (%) | Total (%) |
|---|---|---|---|---|
| 2017 | 85.1 | 14.7 | 0.2 | 100.0 |
| 2018 | 85.1 | 14.9 | 0.0 | 100.0 |
| 2019 | 86.7 | 13.3 | 0.0 | 100.0 |
| 2020 | 86.6 | 13.4 | 0.0 | 100.0 |
| 2021 | 86.9 | 13.1 | 0.0 | 100.0 |
| 2022 | 87.0 | 13.0 | 0.0 | 100.0 |
| 2023 | 87.1 | 12.9 | 0.0 | 100.0 |

### *C. Image Capturing App/ Operating System*

This subsection follows on the chosen smartphone Operating System (OS). For the development of the image capturing application, there are a few open-source software alternatives that can be used. Some of this software include Android Studio, Eclipse Java, Unity, and Microsoft Visual Studio, etc. All these software enables debugging and building android applications in the apk (android package) file format. Android Studio comes with all the basic libraries required for developing an android mobile application without the need to install any other extensions and platforms [16]. It is open-source software that can be installed on any operating system platform. Android Studio uses Java and Kotlin programming languages with the option of linking C++ projects, which makes it easier to use. The software automatically creates the Gradle files required for building mobile applications upon creating a new project. Another alternative is the Eclipse Java software. This software is used to debug and run java applications. It also has an option to merge an android mobile application development plugin to enable android applications development. The user has to download and install android plugins and extensions externally and link it with Eclipse Java software to allow building android applications. It is further required that the developer has to import android libraries to make an android project. This software also uses the Java programming language for application compilation [17]. Unity is another android application development alternative. It is free and open-source software that is used to build simple to complex android applications such as games. This software is easy to use and supports the development of applications using the C++ programming language. This software is a cross-platform game engine that enables a developer to build high-quality 2D and 3D games for mobile, console, desktop, and VR/AR [18]. The Microsoft Visual Studio is another application development software alternative. This software has a free community use version that can be used to develop applications. It uses the C#, C++ and Xamarin languages for compilation. A developer can build native or hybrid android applications using this software. Visual Studio also has a functionality to use the HTML/JavaScript by the Apache Cordova support.

Building android applications, it is required the developer should download and install the android libraries [19]. All these mobile application development platform alternatives have their advantages and disadvantages in terms of usability, complexity, and availability. Considering the factors of ease of access and android libraries required to build the mobile application, the Android Studio alternative is the best choice out of all the alternatives mentioned above for this work. The reasons is because Android Studio comes with all the libraries and Gradle files for app development, hence it is the one chosen for the design of the image capturing app.

### *D. Image*

For this design, the image capturing app will capture an image and save it in the local storage directory of the smartphone. The format of the image can be in the PNG, JPG, JPEG, BMP, PBM, PPM and PNM, etc. The captured image will be used as a cover image for the image steganography technique that will be implemented to embed the secret information on the pixels of the cover image. The image format that will be used for this subsection is the PNG format. PNG has the best quality than the other image formats alternatives because it uses a type of lossless compression whereas the other image formats use a lossy compression [17]. It is required that this cover image has a lossless compression such that when the secret information is embedded in the image pixels the Human Visual System cannot spot that the image pixels has been altered and tampered with. Hence the image will be saved in the PNG format [18].

### *E. Secret Data*

The secret data to be embedded in the image can be a DateStamp or Location Stamp. The DateStamp can be accessed from the smart phone's date information with ease by using the libraries and internal functions of the Android Studio [17]. The Location Stamp of the device is not easily accessible because of the permissions of the Android OS and it requires complex code algorithms to access it. Hence the secret data which will be embedded on the image is chosen to be the DateStamp.

### *F. Embedding Secret Data*

Since the project is based on image steganography, there are a few algorithms that may be used to embed the DateStamp on the image pixels. Some of the alternatives include the LSB method, Masking, and filtering method, Transform techniques, etc. The LSB method is suitable for PNG file formats since it uses a lossless compression to limit the changes/modifications to be observable. This method can also be applied to 24-bit colour images that use 9 bytes memory and to hide secret data it uses only half of the cover image bits [10]. The method of Masking and Filtering is more applicable to JPEG image files because it hides the secret data inside the visible parts of the image. This method changes the visible properties of the cover image, and the changes can be observed by the human visual system. Hence it is not suitable to apply it in this design [20]. Another alternative method is Transform techniques. This method is used strictly by JPEG compression algorithms which are lossy. This is the most complex method of all the other alternative algorithms. The output of the image with the embedded secret data is lossy and can be observable to the human visual system. Hence it is also not a suitable algorithm to use for this design [21]. Hence the chosen algorithm to be used for this design is the LSB method.

### *G. Output*

After embedding the secret data on the cover image, the mobile application will save the stegoimage as an output to the internal storage directory of the Android smart phone. The stegoimage will be in the format of PNG so that the alterations in the cover image cannot be visible to the human visual system.

### *H. Communication App*

The communication app is the platform that will be used to send the stegoimage from the NPO end user-side to the Donor end-user side. Some of the alternatives that are available for this functionality include Gmail, Yahoo, Hotmail, and other platforms. It is required from the specifications/requirements that the communication channel to be used for this function should be able to securely store and transmit the sensitive information of the images that will be captured in the NPO, to protect the identity of the NPO beneficiaries. Gmail is the best choice out of all the other alternatives because it has an end to end secure transmission of information. The Gmail account of the NPO will be used to send the images captured when an event occurs at the NPO. Yahoo is no longer widely used in recent times, hence it will not be used in this design, although it is another good alternative. Gmail will be used as the communication app between the NPO and the Donor.

### *I. Decoding Algorithm*

The decoding algorithm will be implemented to extract the original secret data (DateStamp) hidden in the stegoimage hence this algorithm can be implemented on a mobile device, a desktop computer or a laptop however we used a mobile device for convenience. This algorithm will be the same as the embedding algorithm (LSB method). To lower the complexity of this functionality, this algorithm will not be implemented in a mobile device because it will require another mobile application to be developed and it might get complex. The algorithm can then be implemented on a desktop or laptop computer. It might be required that the Donor end user will want to decode the stegoimage out of office, and hence the more suitable alternative is to use a laptop computer to implement this decoding algorithm.

### *J. Recovered Secret Data*

The recovered secret data is the DateStamp of the image captured at the NPO center during an event. The data can be saved in a spreadsheet, text document or a word document, etc. The data should be kept safe at all times for proof and validation of the images sent from the NPO. The data will be written in an external text document (.txt) file that will be saved in the laptop.

## V. The Proposed Scheme

The LSB method is a simple insertion algorithm to embed information in an image file. The algorithm embeds the bits of the secret message/information directly into the least significant bits plane of the cover image. It uses a deterministic sequence and it modulates the least significant bit, which does not result in a human perceptible difference because the change is too small for humans to observe. The embedding capacity of this algorithm depends on the number of least bits modulated. The higher the number of modulated least bits, the higher the risk of making the embedded message detectable and the image quality/fidelity degrades [21]. To minimize the risk of detecting the secret information, a variable size LSB embedding scheme is introduced. The number of the least significant bits used for embedding the secret message is dependent on the local characteristics of the pixel. The advantages of LSB is that it is easy to implement and produces high message payload. The disadvantage of using this method is that malicious people can easily retrieve the secret message because of the simplicity of the algorithm [22], [23]. To limit malicious people to retrieve the secret information, a system known as the Secure Information Hiding System is proposed. This method overcomes the problem of sequence-mapping by randomly embedding the message in random pixels of the image file. As a result, the secret message is located in random pixels that are scattered on the cover image [24], [25].

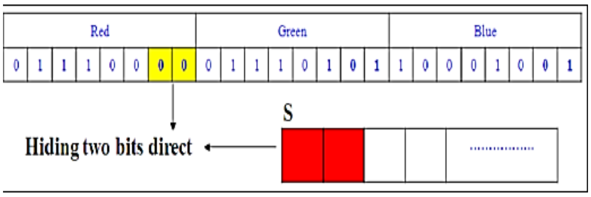


Fig. 5. Illustration of the LSB method [15].

A lossless compression format is necessary to use in this algorithm which in other words, is an attempt to abide with standard and best practices [26], [27]. This is to ensure that the secret information is not lost in the transformations when the algorithm is running. The Fig. 6 illustrates the process of the LSB hiding algorithm. It takes as an input the secret information and an RGB image file, and the password (known as the Stego-key) and it outputs the RGB image (with embedded secret information) known as the Stego-object [28]-[30]. The LSB algorithm scans the image row by row, and it encodes it in binary. It further encodes the secret information also in binary form. The algorithm additionally checks the size of the image and the size of the secret message [8]. It divides the image into three (3) colour groups (Green, Blue, and Red parts). The secret message bits (two by two bits) are hidden in each part of the pixel in the two least significant bits [4]. The image is set with new values and saved, and algorithm ends. In this technique, the secret information is embedded in the cover image by modulating coefficient in a transform domain. The transform domain can be a Discrete Fourier Transform (DFT) or a Wavelet Transform. This technique is more complex, and it applies the modifications of the Discrete Cosine Transformations (DCT). It can be applied over an entire image [31]. This method is used by JPEG compression algorithms. It transforms successive 8 × 8-pixel blocks of the cover image into 64 DCT coefficients each. The DCT coefficient of a single 8 × 8 block of an image pixel is given as in [8].

$$F_{(u,v)} = \frac{1}{4}C(u)C(v)\left[\sum_{x=0}^{7}\sum_{y=0}^{7} f(x,y) * \cos\frac{(2x+1)u\pi}{16} * \cos\frac{(2y+1)v\pi}{16}\right] \quad (1)$$

$$C(x) = \begin{cases} \frac{1}{\sqrt{2}} \quad 0 \quad , & x = 0\,, \\ 1\,, & otherwise \end{cases} \quad (2)$$

When the coefficients have been determined from the above formula, then a quantization operation is performed by the formula below:

$$F^{Q}{}_{(u,v)} = \left\lfloor \frac{F(u,v)}{Q(u,v)} \right\rceil \quad (3)$$

where $Q(u,v)$ is a quantization table with 64-elements. Below are the transformation technique and the corresponding pseudo code of the proposed algorithm is given as in [12]:

**DCT LSB pseudo code**

```
Input: secret information, cover image
Output: Stego image
while data left to embed do
    Get next DCT coefficient from cover image
    If DCT 6= 0 and DCT 6= 1 then
        get next LSB from secret information
        replace DCT LSB with secret information bit
    end if
insert DCT into stego image
end while
```

In a general space, on average, to hide a secret message using the maximum cover image size, only half of the bits in the image will need to be modified. The stego object (stego image) will look the same as the cover image, the changes/modification are too small for the human visual system to recognize. The least significant bit of the colour that is not modified is called a parity bit and can be used to check the correctness of the 8 bits which are embedded in the 3 pixels [14].

## VI. System Model and Assumptions

Our system model comprises of a Wi-Fi router which connect the android smartphone to the internet.

Furthermore, a camera-enabled app is developed using Android Studio with Java programming language. In addition, we assume the focus areas (the rural and remote areas in this case) have constant access to electricity and the shelters used in the NPO centres are safe with enough space to implement the designs. We also assume that these remote areas have signal reception for network and internet connections with a working Gmail account.

The app is developed such that it captures a picture and access the *DateStamp* of the smartphone. The app captures images by using the *android.hardware.camera2* API or the camera intent. This API is used to control the cameras of the android smartphone. The classes that are used include the camera, *SurfaceView* and Intent. The camera class is used to control the cameras and it's the older API. The SurfaceView class is used to make a camera preview of the captured image to the user. The Intent action type *mediastore.action_image_capture* is used to capture images without the use of the camera class object. Android Studio makes use of the AndroidManifest.xml,-MainActivity.java, Activity_main.xml and build Gradle app files to compile the android mobile application [26]. The AndroidManifest.xml file is used to make declarations to give access to the application to use the camera and other features related to the camera hardware. The mobile application should first request permission to utilize the device camera.

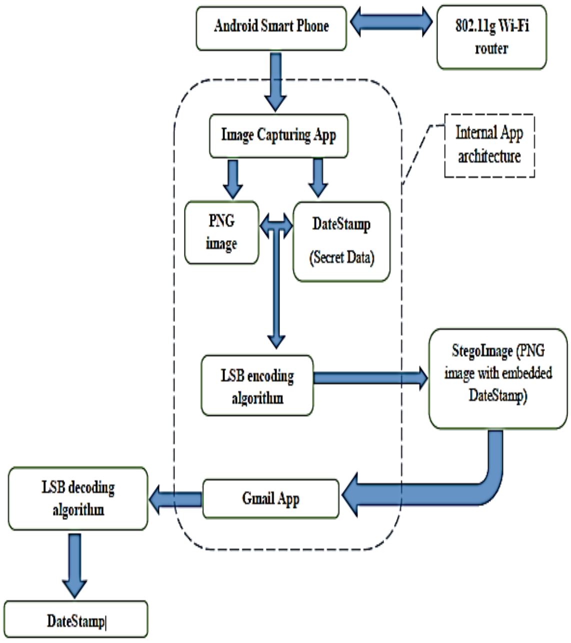


Fig. 6. A Functional block/Flow diagram of our detailed design.

The use of similar camera hardware is also declared in this manifest file. Since the app should be able to save the image in the smart phones' pictures directory, the *write_external_storage* permission is also declared in the manifest. The properties of the application such as the background theme, app icon and the app label are also declared in the manifest [17].

The *MainActivity.java* file contains all the main code that is used to control how the mobile application runs and what it does. All the code to capture images and perform the image steganography using the LSB method is declared in this file. And it uses the layout specified by the *activity_main.xml* file. The Main activity is the screen of the mobile application. The app first requests and implement the permissions from the manifest file and it launches the camera intent and allows the image to be captured in the Main Activity. All the libraries that are used to capture the image and perform the secret data embedding algorithm such as the *graphics. Bitmap*, *media.Image* and *media.ImageWriter* etc. are imported to the Main Activity file. The mobile application access the DateStamp in the form of a String by using the *SimpleDateFormat*() internal function. Then this application captures and save the image in the local storage directory of the android smartphone. To perform the LSB embedding algorithm, the captured image is used as the cover image, the DateStamp string is the secret data and the stego-image is written and saved in the internal storage of the phone by using the *ImageIO. Write()* internal function in the PNG format. The stego-image contains the faces of the NPO beneficiaries, and this information is sensitive and should always be kept safe as per the requirements of the research, hence it is stored directly into the android data files of the mobile application. The *activity_main.xml* file specifies how the application looks like. The app has a "Take Photo" button that is declared in this file. The button is located at the center of the application user interface. Once the button is clicked, it calls the MainActivity.java file and executes the image capturing intent and the LSB embedding algorithm and saves the Stego-image in the picture's directory of the smartphone. The layout orientation of the application is set to be vertical. The last file is the *build Gradle (Module: app)* file. This file contains all the settings of the android API level, application version, compatibility settings, dependencies and all the libraries for implementation and testing the mobile application. The minimum SDK version of this app is set at API 15 (Android 4.0.3) because some of the libraries used required an API level of 15 or above.

The application will work for android devices with Android 4.0.3 or above. The compiler SDK version is set at API 28 (Android 9) because the *ImageIOWrite ()* function only works for Android 9 compiler. The application version is set at 1.0 because it is the first app to be developed for this investigation scope. The compatibility of the source and target compilers is set at version 1.8 due to the *ImageIO* class that is imported to perform image processing. All the implementation and testing libraries required for the LSB algorithm are synchronised using the application's gradle files from the online sources group: *'com.jtransc'*, name: *'jtransc-rt'*, version: *'0.5.0-alpha4'*, *'com.android.support:appcompat-v7:28.0.0'* [17]. The person who captures the images is not aware that the least significant bits of the image has been replaced with the secret data (DateStamp). All the processes (embedding of the secret data) should happen inside the app without any knowledge of the user. The captured image is saved on the internal storage of the android device as a stego-object that carries the secret information. The stego-object is sent via a communication channel (in this case, Gmail is used as the communication channel between the NPO end-user and the Donor end-user). The donor end-user receives the stego-object and recover the secret embedded information using the same algorithm that was used to embed the secret data LSB method [31]-[34]. To recover the secretly embedded DateStamp, a reverse LSB algorithm is used.

The donor can check if the picture is valid by checking the DateStamp embedded if it corresponds to the date of the captured event at the NPO. This will ensure that all false claims practiced by the office bearers of the NPO's are discovered and will be dealt with accordingly.

## VII. Numerical Results and Discussions

The numerical results are presented from experiments conducted to determine functionality of the proposed scheme. Table I shows the download and upload speeds of a typical 802.11g Wi-Fi router while varying the distance between a mobile device and the router. This experiment was conducted to determine the most convenient position to place the Wi-Fi router for the best possible download and upload speed reception. This is indispensable for fast transmission of the pictures captured in the NPO center.

Table II: Showing Download and Upload Speed of an 802.11g Wi-Fi Router at Varying Distance Between Router and a Mobile Device.

| Distance between router and mobile device (m) | Download speed (Mbps) | Upload speed (Mbps) |
|---|---|---|
| 1 | 51.86 | 48.21 |
| 4 | 49.23 | 46.98 |
| 6 | 45.85 | 40.23 |
| 10 | 39.46 | 32.42 |
| 15 | 21.72 | 27.67 |
| 20 | 18.61 | 19.63 |
| 30 | 11.72 | 18.67 |
| 35 | 7.89 | 4.65 |
| 40 | 2.57 | 2.59 |

Fig. 7 and Fig. 8 shows the results for the download speed vs. distance.

From 1 m to 10 m, much of the signal strength is not lost because the mobile device is still in close range with the router. There is a sharp drop of the download speed between the distances of 10 m and 15 m, this can be due to the room temperature changes or obstacles between the mobile device and the router. From 15 m to 40 m, the trend line exhibits a linear relationship between the download speed and the distance.

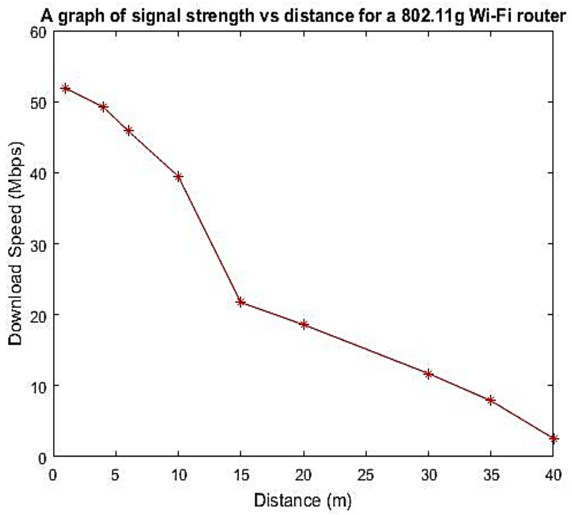


Fig. 7. A graph of download speed vs. Distance

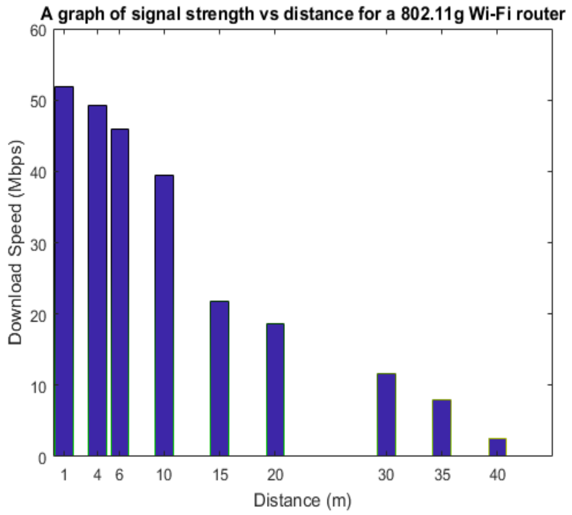


Fig. 8. A bar chat for download speed vs. distance

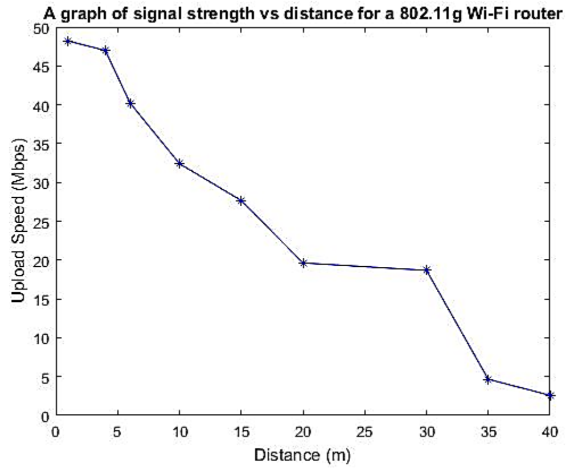


Fig. 9. A graph of upload speed vs. Distance

Fig. 9 and Fig. 10 shows the results for the upload speed and vs. distance. From 1 m to 20 m, there exists a linear relationship between the upload speed and the distance. Between 20 to 30 m, the signal strength remains constant. There is a sharp drop in signal strength between

30 m and 35 m, this can be as a result of far field signal losses

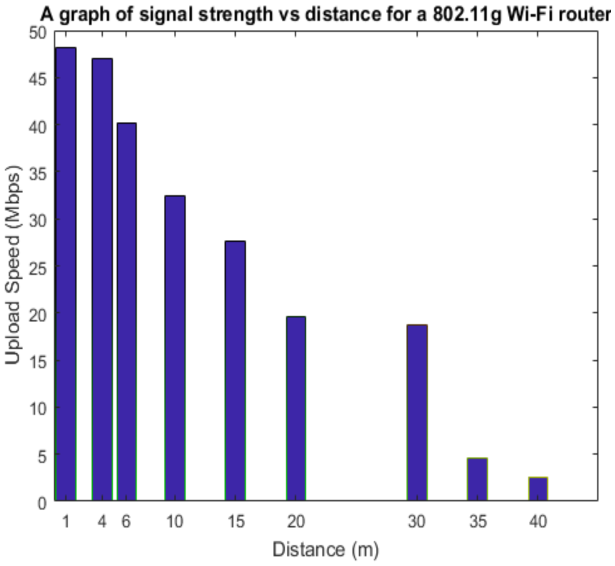


Fig. 10. A bar chat of upload speed vs. distance

The experiment was conducted to test the signal strength of an 802.11g Wi-Fi router for several distances away from the router module for a connected mobile device. The results obtained show that the upload and download speeds are inversely proportional to the distance between the router and the connected mobile device.

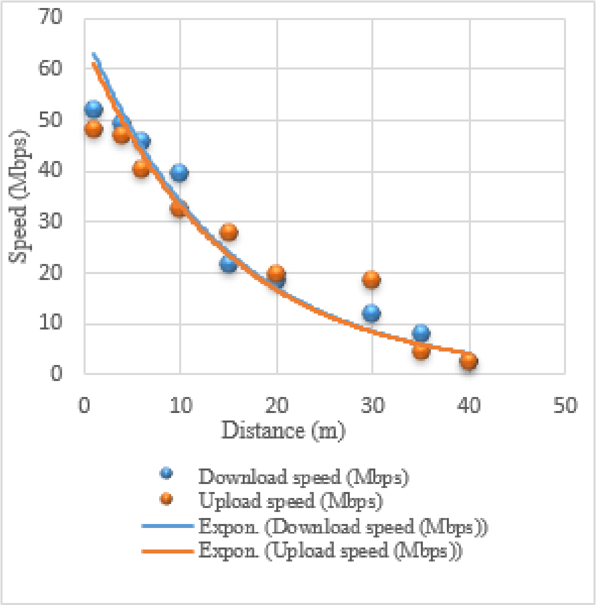


Fig. 11. Combined results of the download and upload speed for an 802.11g Wi-Fi router

The best fit line in Fig. 11 shows the exponential characteristics of the relationship between the upload/download speed and the distance. Between the distances 1m and 15 m, the download speed is greater than the upload speed. And from 15 m to 40 m, the upload speed tends to be higher than the download speed.

This illustrates that the further we move from the router, the more we lose upload signal strength than download. Hence download speed gets to be best at a closer range of the router than in the far field. Another experiment to test the functionality of the developed android app was conducted, and the results are presented in Fig. 12. It shows the interface of the mobile application. When the app is run, there is a 'TAKE PHOTO' button to capture pictures. Whenever a picture is captured, a DateStamp is secretly hidden in the LSB of the picture.

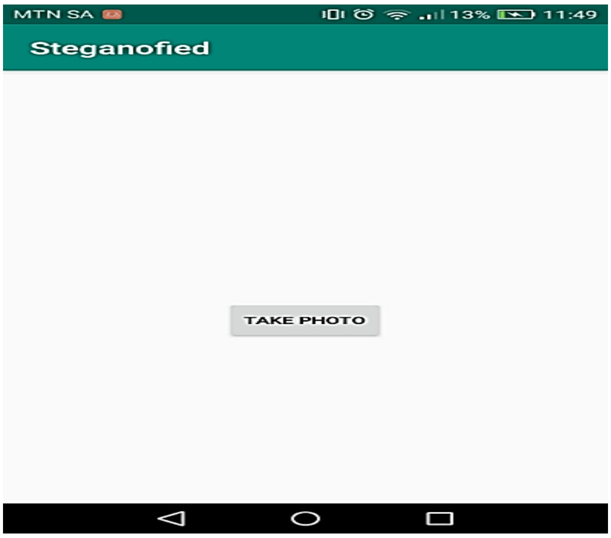
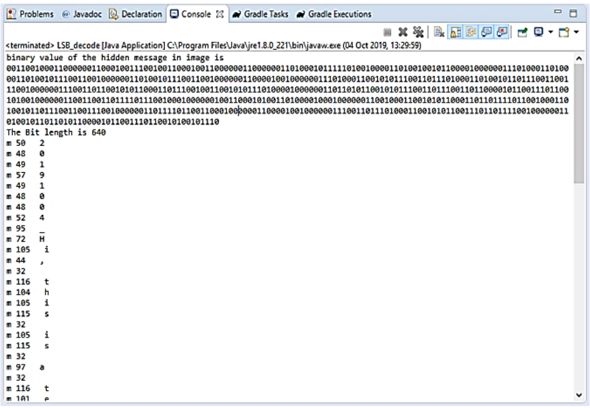


Fig. 12. Steganofied mobile app interface



```
<terminated> LSB_decode [Java Application] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (04 Oct 2019, 13:29:59)
binary value of the hidden message in image is
00110010001100000011000100111001001100010011000000110000001101000101111101001000011010010010110000100000011101000110100
00110100101110011001000000110100101110011001000000110000100100000011101000110010101110011011101000110100101101110011001
11001000000111001101100101011000110111001001100101011101000010000001101101011001010111001101110011011000010110011101100
10100100000001100110011011110111001000100000001001100010100110100001000100000001100100011001010110001101101111011001000110
10010110111001100111001000000110111101100110001000000110000100100000011100110111010001100101011001110110111100100000011
010010110110101100001011001110110010100101110
The Bit length is 640
m 50   2
m 48   0
m 49   1
m 57   9
m 49   1
m 48   0
m 48   0
m 52   4
m 95   _
m 72   H
m 105   i
m 44   ,
m 32
m 116   t
m 104   h
m 105   i
m 115   s
m 32
m 105   i
m 115   s
m 32
m 97   a
m 32
m 116   t
m 101   e
```

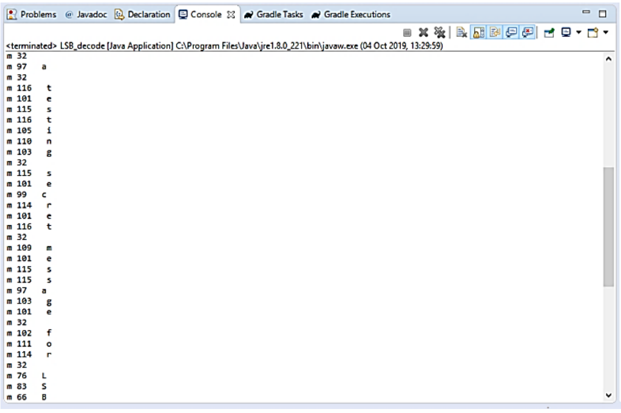
Fig. 13. Decoded secret data-1 hidden in Steg.png

The developed Steganofied mobile app was used to capture an image as shown in Fig. 12. The decoding algorithm was implemented on a desktop computer and the hidden secret message was decoded as in figures 14 below. The results in Fig. 13 and Fig. 14 are each of the secret message characters in each corresponding bitmap of the Stego-image. The Fig. 13 is the same results, and it were separated into two figures due to the space on the console platform.

Fig. 14 elucidates the bit length of the encoded secret message. And it maps each Least Significant Bit pixel to its analogous character hidden in the specific bit plane.

The results presented in Fig. 15 are the extension of the results illustrated in Fig. 14 since they were separated due to space constraint in the java console used for decoding.

The Steg.png stegoimage does not show any properties of image noise introduced by the encoding of the secret message in some of its bit pixels



```
<terminated> LSB_decode [Java Application] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (04 Oct 2019, 13:29:59)
m 32
m 97   a
m 32
m 116  t
m 101  e
m 115  s
m 116  t
m 105  i
m 110  n
m 103  g
m 32
m 115  s
m 101  e
m 99   c
m 114  r
m 101  e
m 116  t
m 32
m 109  m
m 101  e
m 115  s
m 115  s
m 97   a
m 103  g
m 101  e
m 32
m 102  f
m 111  o
m 114  r
m 32
m 76   L
m 83   S
m 66   B
```

Fig. 14. Decoded secret data-2 hidden in Steg.png

This shows that the LSB encoding method is suitable for png image files and hackers/intruders will not be able to detect that the picture contains a secret message. The encoded secret message in the Stego image was successfully decoded by using the LSB decoding algorithm. It shows the DateStamp and the message hidden in the Stego-image. Hence the decoding algorithm is highly efficient and works well.

## VIII. Conclusion

This paper integrates smartphone network architecture and data security techniques to mitigate sharp practices in NPOs and serves as a monitoring/regulatory and management tool for rural NPOs. From the analysis of the results obtained, the use of the variable size LSB for this research yielded the best results for hiding a small-scale data on a PNG image with minimum observable modifications. In our further work, the LSB image steganography technique will be applied and implemented in an android mobile app for flexible operation in the application of the study objective. The app can be further customized to also embed the Location-Stamp of a captured event. The location will be greatly efficient in validating that indeed an event (Sharp practice) has occurred in the rightful location of the NPO center. Our system will further be automated to dynamically detect and discard all the pictures that do not complement some given criteria. This will make it easier for the donors to cut the manual decoding of the pictures. Also, we will consider the use of Television Wide Space (TVWS) as a medium of communication instead of the conventional Wi-Fi/GSM network protocol as proposed in [33]-[35]. In addition, a comparison of some steganography techniques and the corresponding analysis will be studied. Apart from that, we will be looking at the Advanced Encryption Standard (AES) which has been extensively applied in several areas, such as Internet-of-Things (IoT) and wireless communication [36].

## Conflict of Intrest

The authors declare no conflict of interest.

## Authors Contribution

Conceptualization, M. C. N, E. E, R. H ; methodology, M. C. N., E. E, R. H.; software, M. C. N., E. E.; validation, M. C. N., E. E.; formal analysis, M. C. N., E. E, R. H.,: investigation, M. C. N., E. E, R. H..; resources, E. E, R. H.; data curation, E. E, R. H..; writing original draft preparation, M. C. N., E. E,.; writing review and editing, E. E, R. H.; visualization, E.E.; supervision, E. E, R. H..; project administration, E. E, R. H..; funding acquisition, E. E, R. H.

## Funding

This research received no external funding. However, the APC will be funded from the research center.

## Acknowledgement

This work is supported partially by the Centre for Collaborative Digital Networks and Prof T. G. Swart Center of Telecommunications, University of Johannesburg.

## References

[1] M. Chapman, G. Davida, and M. Reinhard, *A Practical and Effective Approach to Large-Scale Automated Linguistic Steganography*, Springer-Verlag Berlin Heidelberg, 2001, pp. 156–165.

[2] L. Eugene and E. Delph, "A review of data hiding in digital images," in *Proc. IS&T's PICS Conference*, Savannah, Georgia, April 1999, pp. 274-278.

[3] D. Bret, "A detailed look at Steganographic Techniques and their use in an open-systems environment," Sans Institute, 2002.

[4] K. Sara, A. D. Mashallah, and H. Y. Mohammad, "A new steganography method based HIOP (Higher Intensity of Pixel).

[5] V. K. Pachghare, *Cryptography and Information Security*, (Second Edition) Published by PHI Learning, 2015.

[6] P. Wayner, *Disappearing Cryptography Information Hiding: Steganography & Watermarking*, 2009.

[7] B. Souvik, B. Indradip, and S. Gautam, "Data hiding through Multi Level steganography and SSCE," *Journal of Global Research in Computer Science*, vol. 2, no. 2, 2011.

[8] K. Jagvinder and K. Sanjeev, "Study and analysis of various image steganography techniques," *International Journal of Computer Science and Technology*, vol. 2, no. 3, pp. 353-359, 2011.

[9] I. A. Aiad, "Hiding data using LSB," *J. Basrah Researches (Sciences)*, vol. 33, no.4. pp. 81-88, December 2007.

[10] S. Katzenbeisser and F. A. P. Petitcolas, "Information hiding techniques for steganography and digital watermarking," Artech House, Norwood, MA.

[11] D. Guillermito. Steganography: A Few Tools to Discover Hidden Data, (2004). [Online]. Available: http://guillermito2.net/stegano/tools/index.html

[12] A. F Mohammed, "Image steganography by mapping pixels to letters," *Journal of Computer Science*, vol. 5, no. 1, pp. 33-38, 2009.

[13] S. S. Al-Hussein, "Enhancing the (MSLDIP) Image Steganographic method (ESLDIP Method)," in *Proc. International Conference on Graphic and Image Processing*, 2011.

[14] D. R. Stinson, *Cryptography: Theory and Practice: CRC Press*, 2005.

[15] A. R. Ahmed, S. Ahmed, and S. S. Al-Hussein, " A high capacity SLDIP (Substitute Last Digit in Pixel)," in *Proc. Fifth International Conference on Intelligent Computing and Information Systems*, Cairo, Egypt, 2011.

[16] Takealot. Cell phones and Wearables, Cellular accessories. [Online]. Available: https://www.takealot.com

[17] M. Electronics. ManTech Electronics. [Online]. Available: https://www.mantech.co.za/default.aspx.-

[18] Google. Android Developers. Google, [Online]. Available: https://developer.android.com

[19] J. Fredrich and M. Long, "Steganalysis of LSB encoding in color images," in *Proc. IEEE International Conference on Multimedia and Expo*., NY, New York, 2000.

[20] Direct Sequence Spread Spectrum (DSSS), Frequency Hopping Spread Spectrum (FHSS) Wikipedia, The Free Encyclopedia, GNU Free.

[21] A. A. Abdelmged and S. S. Al-Hussein, "Image steganography technique by using braille method of blind people (LSBraille)," *International Journal of Image Processing*, vol. 7, no. 1, 2013.

[22] A. M. Mohamed, S. Ibrahim, M. Salleh, and M. R. Katmin, "Information hiding using steganography," in *Proc 4th National Conference of Telecommunication Technology*, Shah Alam, Malaysia, 2003.

[23] K. Atul, *Cryptography and Network Security*, Second Edition, Tata McGraw-Hill Education, 2009.

[24] A Guide to the Project Management Body of Knowledge, Newtown Square, Pennsylvania 19073-3299 USA: Project Management Institute, Inc., 2008.

[25] C. Hosmer, "Discovering hidden evidence," *Journal of Digital Forensic Practice*, vol. 1, no. 1, pp. 47-56, 2007.

[26] J. Nicholas, J. L. Hopper, and V. A. Luis, "Provably secure steganography," *IEEE Transactions on Computers*, vol. 58, no. 5, pp. 662–676, 2009.

[27] M. Natarajan and N. Lopamudra, "Steganalysis algorithms for detecting the hidden information in image, audio and video cover media," *International Journal of Network Security & Its Applications*, vol. 2, no. 1, pp. 43-55, 2010.

[28] C. Eric, *Hiding in Plain Text*, Wiley Publishing, Inc, 2003.

[29] F. J. Neil and J. Sushil, *Steganography: Seeing the Unseen IEEE Computer*, pp. 26-34, 1998.

[30] E. A. Akbas, "A new text steganography method by using non-printing Unicode characters," *Engineering and Technology Journal*, vol. 28, no. 1, pp. 72-83, 2010.

[31] S. Chutani and H. Goyal "LSB embedding in spatial domain - a review of improved techniques," *International Journal of Computers & Technology*, vol. 3, no. 1, 2012.

[32] Information and Computer Security Architecture (ICSA) Research Group, An Overview of Image Steganography, University of Pretoria, Pretoria, South Africa.

[33] E. Esenogho and T. Walingo, "Performance evaluation of channel assembling strategies with multi-class secondary users in cognitive radio networks," in *Proc. South Africa Telecommunication, Networking and Application Conference SATNAC'15*, Cape Town, South Africa, September 2015, pp. 81-86.

[34] E. Esenogho, E. N. Mambou, and H. Ferreira, "Integrating two queuing regime into cognitive radio channel aggregation policies: A performance evaluation," *International Journal of Electronic and Telecommunications*, vol. 64, no. 4, pp. 519–525, 2018.

[35] E. Esenogho and V. M Srivastava, "Channel assembling strategy in cognitive radio networks: A queuing-based approach," *International Journal on Communications Antennas and Propagation*, vol. 7, no. 1, pp. 31-47, 2017.

[36] X. Yang, W. Wen, and M. Fan, "Improving AES Co re performance via an advanced IBUS protocol," *ACM Journal on Emerging Technologies in Computing*, vol. 14, no. 1, pp. 61-63, Jan. 2018.

Matimu Caswell Nkuna received the B.Eng. (Second Class Upper-Div.) degree in Electrical/Electronic Engineering in 2019 and Currently pursuing his M.Eng degree in Electronic/Telecommunication Eng., at the University of Johannesburg, King square, Auckland Park Campus. His research interests include Mobile Traffic, IoTs Deployment and Big Data, Cyber-Security, Information Theory, Machine learning and 3D robotic gamming.

**Dr. Esenogho Ebenezer** received the B.Eng. degree in Computer Engineering in 2008, the M.Eng. in Electronic/Telecomm Engineering from the University of Benin in 2012. He previously lectured at the University of Benin before embarking on his Ph.D program in 2013. Prior to now, he was involved in research and teaching with the Centre for Radio Access and Rural Technology, University of KwaZulu-Natal, Centre of Excellence, where he rounded-up his Ph.D degree in Electronic/Telecom (5G Cognitive Network). Currently, he is hired into the prestigious Global Excellence Stature Post-Doctoral Research Fellowship at the University of

Johannesburg, Auckland Park under the Institute for Intelligent System (IIS), Center for Telecommunication Research (CfT) to pursue further research. He is a recipient of several grants/scholarship/fellowships including the CEPS/Eskom's HVDC 2013, CEPS/Eskom's HVDC 2014, J. W Nelson 2015 and GES 2017-2020. Presently he is serving as the first postdoctoral research fellowship representative in the University of Johannesburg Senate, 2018-date. He was a UJ/DST/NRF research delegate to the H2020-ESASTAP EU-South Africa STI Cooperation on Strengthing Technology Research and Innovation in Vienna, Austria. Dr. Esenogho has authored/co-authored several peer-reviewed journals and conference papers, chaireed session in conferences and, reviewed for some notable ISI/Scopus journals in his field. His research interests are in the Fifth Generation (5G) Wireless Networks, Cognitive Radio Networks, Smart Grid Networks, IoT/IoE, SDN/SDR, Wireless Sensor Networks, Artificial Intelligence, Mobile Computing and Visible light communication. He is a registered Engineer and member SAIEE/IEEE region 8.

**Dr Reolyn Heymann** is a researcher and a Senior Lecturer in the Faculty of Engineering and the Built Environment, Department of Electrical and Electronic Engineering Science, University of Johannesburg, Auckland park Campus. She doubles as the Director of the Liquid Telecoms Research Hub/ Centre for Collaborative Digital Networks at the same University. Her research interests include Telecommunications, Information Theory, Machine learning, 3D gamming just to mention a few.