Secure Data Hiding and Extraction Using RSA Algorithm

Aminu Aminu Muazu[1,2(✉)], Umar Danjuma Maiwada[1,2], Abubakar Rufa'i Garba[2], Muhammad Garzali Qabasiyu[3], and Kamaluddeen Usman Danyaro[1]

[1] Department of Computer and Information Science, Universiti Teknologi PETRONAS, Perak, Malaysia
{aminu.aminu,umar.danjuma}@umyu.edu.ng, kamaluddeen.usman@utp.edu.my

[2] Computer Science Department, Faculty of Natural and Applied Sciences, Umaru Musa Yar'adua University, Katsina, Nigeria
abubakar.rufai@umyu.edu.ng

[3] Department of Computer Studies, Hassan Usman Katsina Polytechnic, Katsina, Nigeria
garzaliqabasiyu@hukpoly.edu.ng

Abstract. Data hiding and extraction is an important aspect of information security because the data will be safer and more manageable. Data hiding technology that works well produces data that is efficient, secure, and simple to connect. The underlying communication network that enables the transport of sensitive data is insecure and unprotected. The fast rise of electronic methods of communication implies that information security has become a critical concern in the real world. As such, anyone with the necessary expertise and software may eavesdrop and intercept data transmissions, which can be extremely harmful and even life threatening in so many cases. Therefore, the research of this paper offers a software framework that allows users to create messages and hide them within image file documents that can be sent to a desired recipient. Moreover, we conducted an experiment with different data that guarantee the increase level of confidentiality of information exchange over the internet.

Keywords: Data hiding · RSA algorithm · Encryption · Decryption · Bit-plane

1 Introduction

In today's environment, information technology is the most important factor [1–3]. Based on this fact, computer applications are still being developed to better manage financial and personal data in a safe manner [4–7]. These data are incredibly valuable in every way, and we must protect them from illegal access. The process of preventing and detecting unwanted data, computer, or network usage is known as security [8]. Preventative measures assist us in preventing unauthorized users from gaining access to any component of the computer system [9]. Detection aids in determining whether someone attempted to break into the system, if they were successful, and what they did.

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022
V. Sugumaran et al. (Eds.): AIR 2022, CCIS 1738, pp. 14–28, 2022.
https://doi.org/10.1007/978-3-031-23724-9_2

We may employ a variety of encryption approaches to attain that security. However, data encryption is no longer sufficient, and we must additionally safeguard the existence of data if we want to achieve good security [10, 11].

The Rivest-Shamir-Adleman (RSA) Algorithm becomes necessary at this point. However, we must keep in mind that neither cryptography nor the RSA method can guarantee entire security for data or information, thus we must use both approaches to obtain necessary security. There are a variety of ways to accomplish this, but we'll focus on techniques for modifying information in such a manner that the receiver can reverse the change and recover the original content [10, 12]. Although the RSA method is sometimes mistaken with cryptography, there are several significant and clear distinctions between the two. Because the cypher text is a scrambled output of the plaintext in cryptography, the attacker can predict that encryption has been conducted and so use decryption techniques to obtain the secret data in some scenarios, the RSA Algorithm is generally preferable over cryptography [12]. Furthermore, cryptographic approaches frequently necessitate a lot of computer power to execute encryption, which can be a big problem for small devices that don't have enough computing power [8]. To disguise the data in this paper, a bitmap (bmp) picture will be employed. The pixels will be used to insert data into the picture. The RSA technique may then be used to recover the concealed data inside the picture [8].

The remainder of the paper is structured as follows: Sect. 2 will discuss some related works, Sect. 3 to present the Image Data Hiding Algorithm for Donathan Hutchings, Sect. 4 contains the System Implementation Using RSA Algorithm, Sect. 5 gives the system evaluation, and the final Sect. 6 to conclude the paperwork.

2 Related Works

Kuo et al. has proposed a reversible adaptive data concealing technique which is a novel strategy based on the histogram and slope method that improves data concealing capacity while simultaneously increasing efficiency and maintaining image quality [13]. Ma et al. has another strategy which was introduced, they used a method called reversible data hiding (RDH). The space for data embedding is discovered in this method by examining the redundancy in the provided picture. Because of the storage capacity of secret data and the visual quality of embedded images, this approach is the finest of all the techniques in literature survey. In addition, when compared to other methods, this has a low computing complexity [14]. Xioatian et al. proposed another technique for partitioning an image into many encrypted shares to safeguard it [15]. When sufficient shares were obtained, the image can be losslessly retrieved. The study implements an image encryption algorithm using Shamir's secret sharing methods. Ref. [16] introduced a unique reversible data hiding in encrypted images (RDHEI) technique that synchronizes embedding and re-encryption and is well implemented by a data hider that uses a permutation ordered binary (POB) number system. This approach uses two peak locations for embedding, but it completely disregards the previous idea of a shifting process. The proposed method's superiority was confirmed by the experimental results.

Another high-capacity RDHEI technique based on multi-MSB (most significant bit) prediction and Huffman coding was introduced in ref. [17]. Initially, the multi-MSB of each pixel was adaptively anticipated and marked in the original image using

Huffman coding. The image was then encrypted using the stream cypher approach. Finally, utilizing multi-MSB replacement, the empty space was utilized to embed more data. The proposed approach achieved a higher embedding capacity, according to the experimental data.

In [12] a new hybrid approach combining RSA and steganography was introduced. The RSA technique was employed to encrypt and decode the secret file for image compression in the study. Furthermore, the two algorithms were used to compress images for both lossy and lossless compression techniques. This approach has a larger capacity, allowing the total message bits to be reduced by up to 25% of the original message bits. Achieving an average PSNR score of above 40db and SSIM nearest to the unit also demonstrates good stego-image quality.

It is more broadly concerned with establishing and analyzing protocols that are resistant to adversarial impact and are linked to various aspects of information security, including data secrecy, data integrity, authentication, and non-repudiation. However, the modern cryptography brings together mathematics, computer science, and electrical engineering. Cryptography applications include ATM cards, computer passwords, and electronic commerce. The usage of a key file, full-volume encryption, and plausible deniability are only a few of the features. The broad availability of software that performs these activities has significantly disadvantaged the area of digital forensics. Most of the open-source encryption software allows users to build virtual encrypted drives that can only be accessed with a certain key. These programs make data practically impossible to read without the required key by employing advanced encryption algorithms and techniques.

3 Image Data Hiding Algorithm for Donathan Hutchings

This algorithm embeds the text inside a graphic file and creates a key file to be used to read the message later.

Input: Image File, Secret_Message.

a. Determine the size of the Image File.
b. Determine the length of the Secret_Message.
c. Determine the offset to use when embedding the Secret_Message. //This gives a reliable starting position based on the sizes of the Image_File and Secret_Message.
d. Loop through the Image_File data and begin placing one Secret_Message byte at a time.
e. Make sure we are not at the end of the Secret_Message. //This keeps the loop above to continue swapping the Image_File byte with the Secret_Message byte.
f. Expand our key data array. //This array stores the locations of each Secret_Message byte.
g. Store the location of the whole Secret_Message byte.
h. Create a new Image_File with the Secret_Message inside.
i. Create the key file.

Output: Stego_Image.
Input: Stego_Image _File.
This algorithm reads the message stored in a graphics file.

a. Determine the size of the Image_File.
b. Determine the length of the Secret_Message.
c. Read the Image_File byte data into a byte array.
d. Read our message location data from the key file.
e. Loop through the Image_File data to find our message and construct our actual Secret_Message.
f. Get the message byte from the Image_File data.
g. Convert the byte data back to our original Secret_Message.

Output Secret_Message.

3.1 Implementation

The use case diagram depicts the system's users, its components, and the interactions that exist between them (Fig. 1 and Fig. 2).

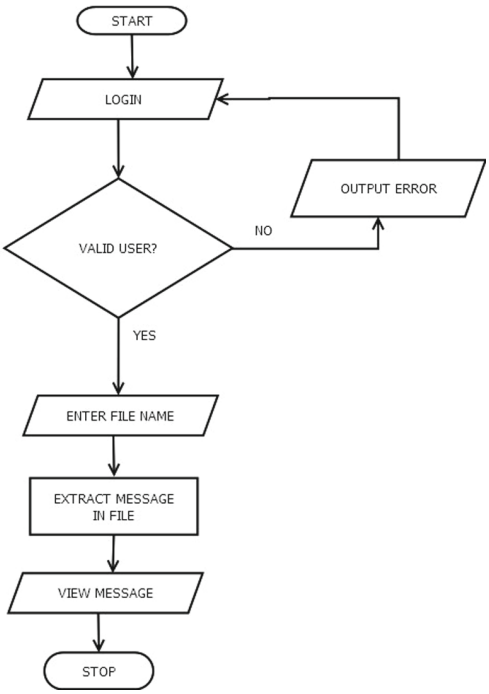

Fig. 1. Data retrieval process

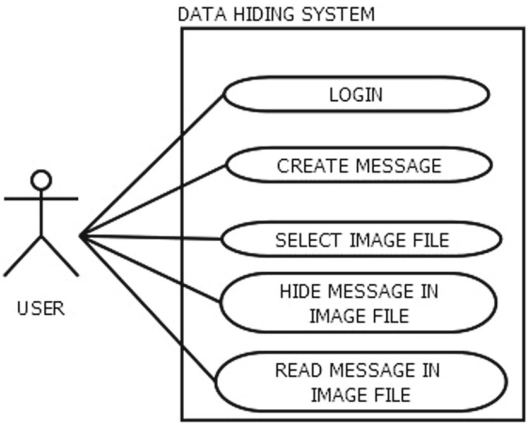

Fig. 2. Use case diagram

3.2 User Interface Design

When you launch the programme, the first interface that displays on the screen is illustrated is Fig. 3, Encrypt Image and Decrypt Image are the two-tab options for encryption and decryption, respectively. The right top panel displays image information such as size, height, and width.

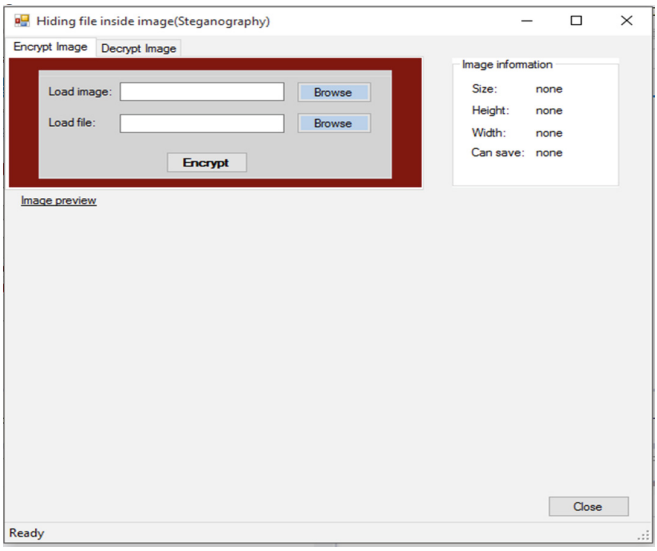

Fig. 3. Interface

3.3 Image Tab Encryption

The equation should be submitted in the equation editor and should be keyed in as below in editable text:

a. To encrypt an image, go to the encrypt image tab (Fig. 4).

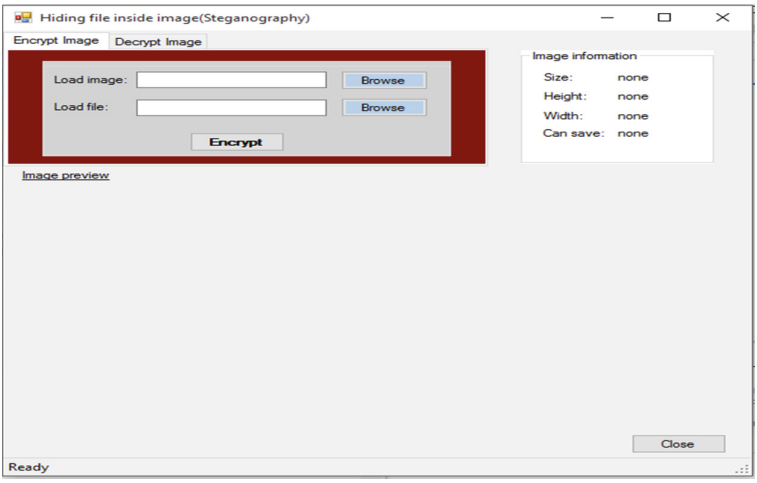

Fig. 4. Encryption tab

b. Click the "Browse" button next to the Load Image textbox to load an image. Select the Image file that you wish to use to hide your information and click the Open button in the file open dialogue box that appears (Fig. 5).

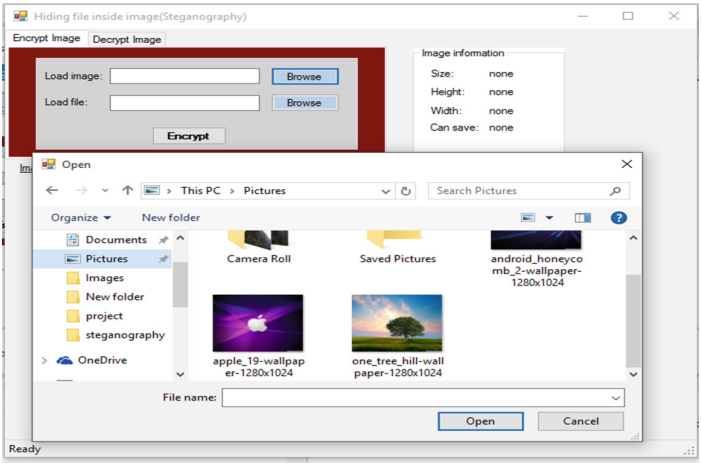

Fig. 5. Load Image interface

c. The image file will open and look like the Fig. 6 below. Then, next to the Load File textbox, select the "Browse" button.

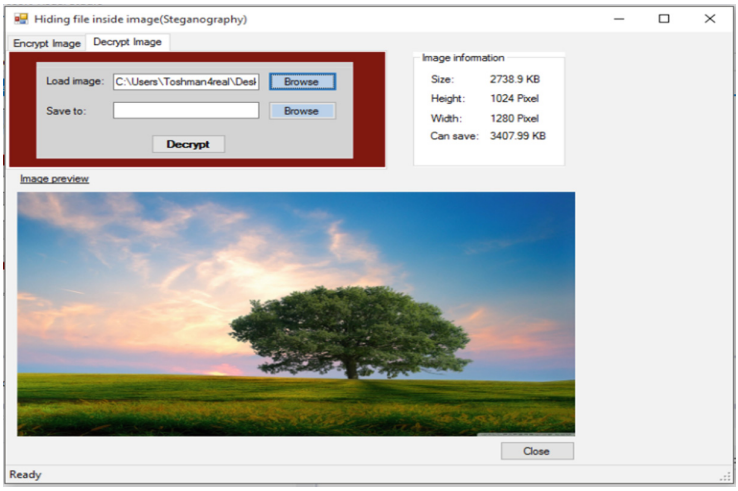

Fig. 6. Load image preview

d. The file open dialogue box will appear once more; select any file you want to hide within the image and click the open button (Fig. 7).

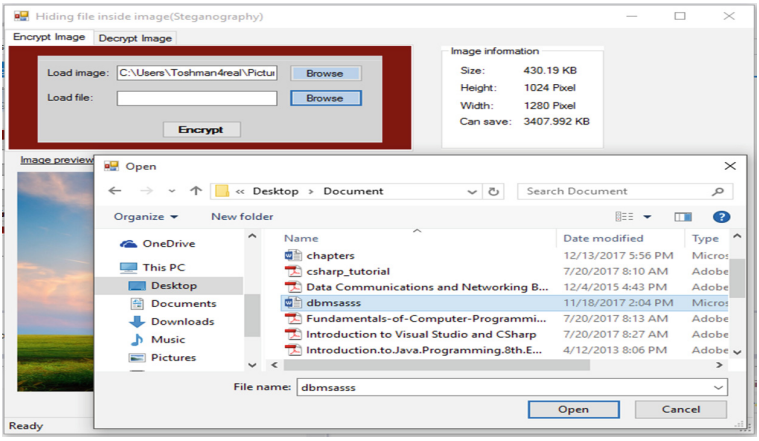

Fig. 7. Open file dialog for the encryption file

e. Encrypting the file is the next step. When you click the "Encrypt" button, a save dialogue box will appear, asking you to choose a location for the new image file and a name for it (Fig. 8 and Fig. 9).

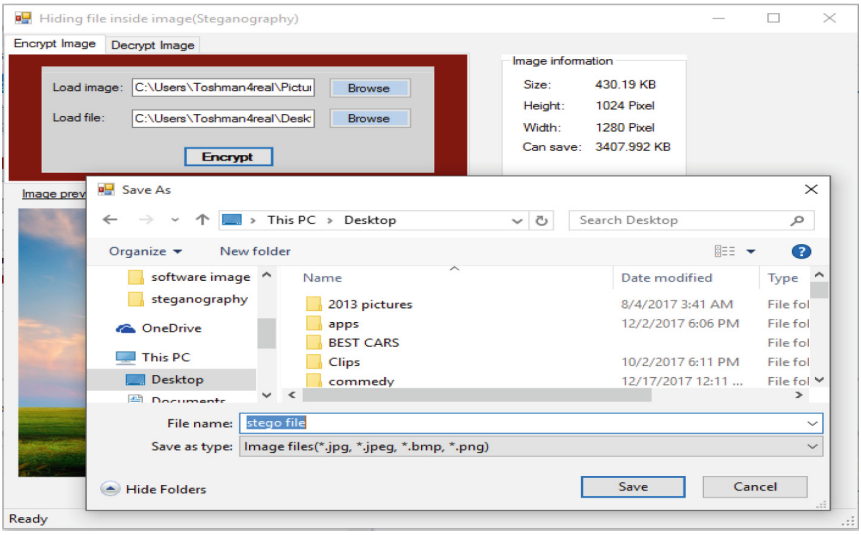

Fig. 8. Save file dialogue box

3.4 Image Tab Decryption

a. To decrypt an image, go to the decrypt image tab (Fig. 10).

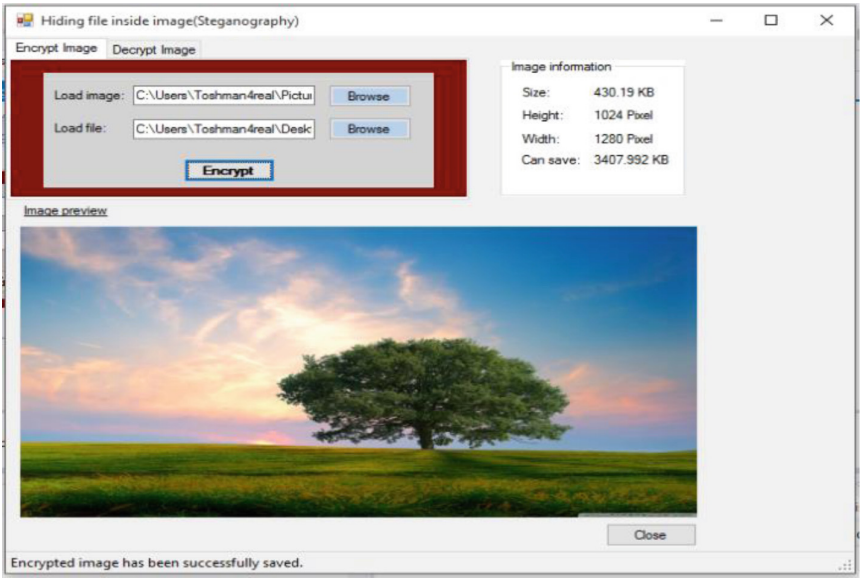

Fig. 9. Interface after successful encryption

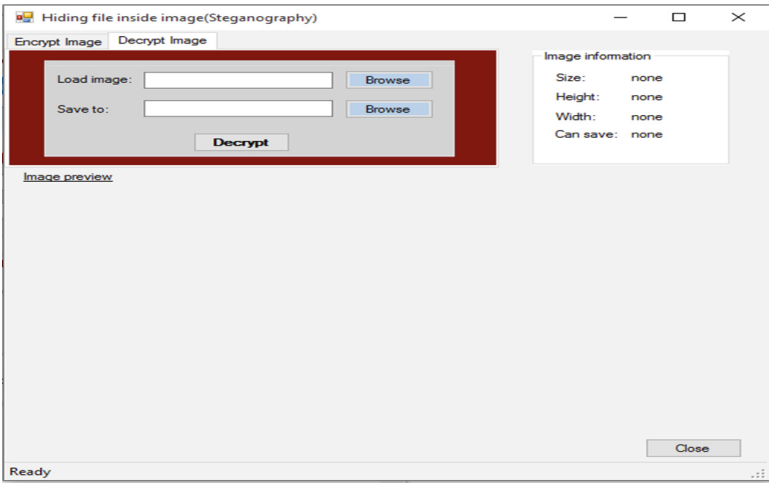

Fig. 10. Decrypt image tab

b. After that, click the "Browse" button to enter the Open file dialogue box, where you must select the image that is encrypted and contains concealed data. Click on the Open button after selecting the image file (Fig. 11).

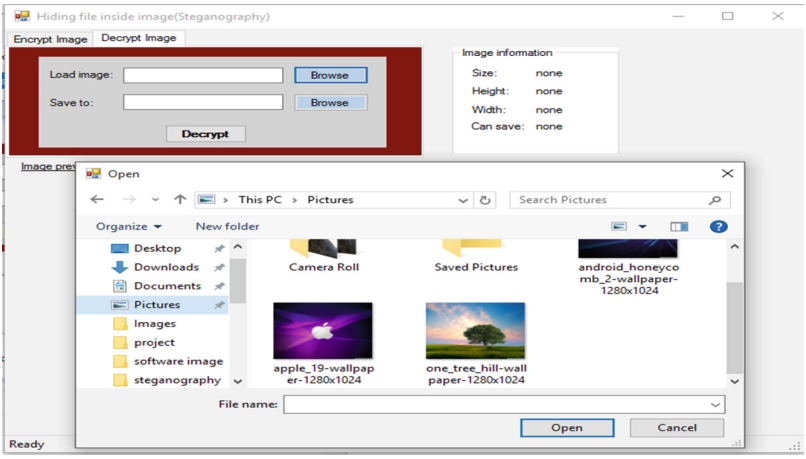

Fig. 11. Open encrypted file dialog box

c. The image file looked like the Fig. 12 below:

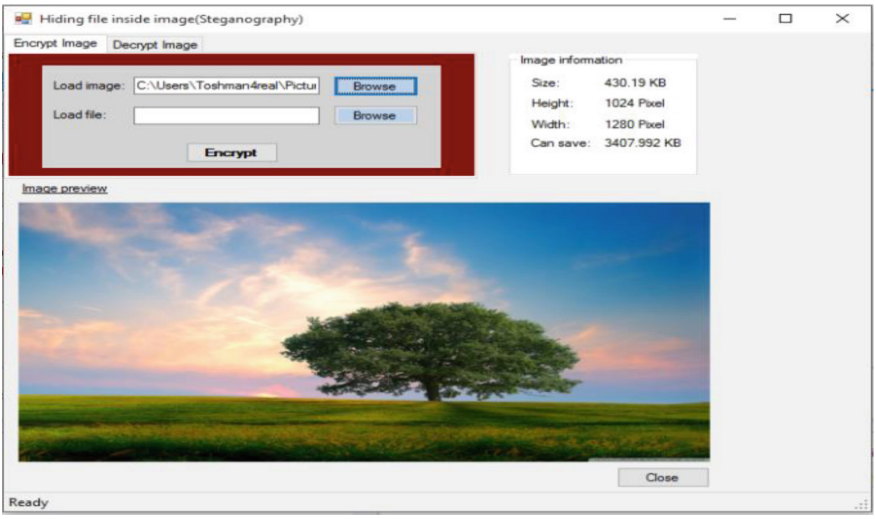

Fig. 12. Encrypted image file display

d. Now, next to the “Save file to” textbox, click “Browse” button. It will bring up a “Browse for Folder” dialogue box. It will prompt you to choose a path or folder to extract the hidden file from. Click the Ok button after selecting the folder (Fig. 13).

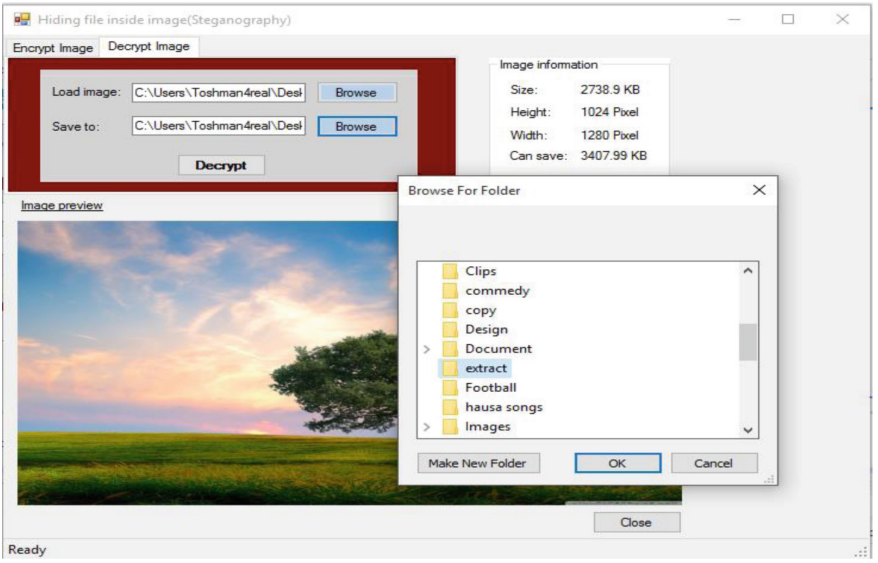

Fig. 13. Save file dialog box

e. Now, click the Decrypt button, which will decrypt the image and save the secret file to the specified folder. The status bar, which is located at the bottom of the screen, displays the message for successful decryption (Fig. 14).

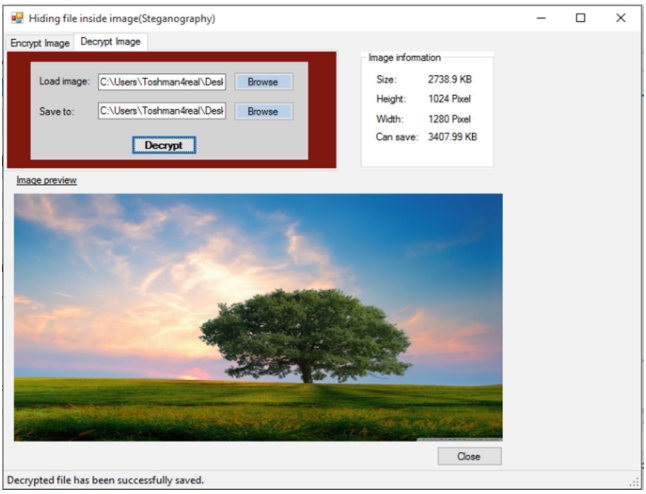

Fig. 14. Interface after successful decryption

4 System Implementation Using RSA Algorithm

This system was implemented using a parallel approach strategy. We construct a basic system that implements the RSA Algorithm based on the algorithm. The system is known as the Data Hiding System (DHS). According to the system's framework, the first layer is for login purposes, and the second layer is for concealing, retrieving, and finally displaying concealed message file information interface. Below are the several user interfaces that make up the Data Hiding System (Fig. 15):

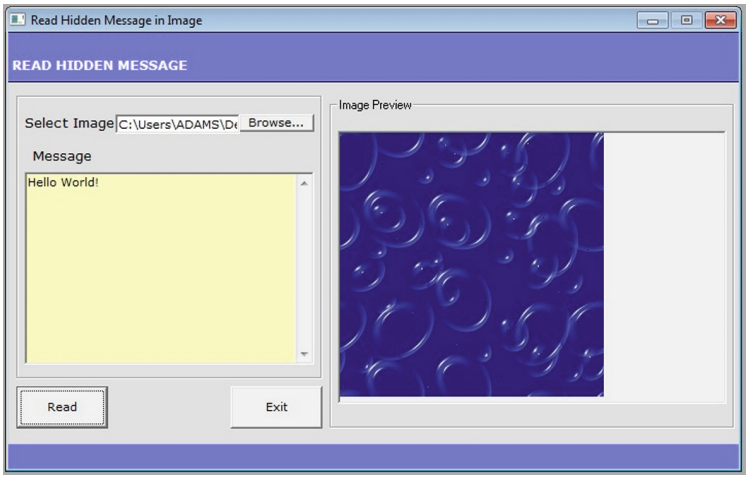

Fig. 15. Read hidden message interface

This form is used to decipher the secret message in the image. The user gets shown the hidden message in the picture file after selecting the image (Fig. 16).

This form is used to access the database of information about the hidden message pictures.

5 Evaluation

The equation should be submitted in the equation editor and should be keyed in as below in editable text:

The bitmap (BMP) format is the image file format utilized in this paper. Within the Microsoft Windows operating system, the BMP file format is used to store graphic files. BMP files are often uncompressed and hence big. The ease of use and widespread acceptance of BMP files in Windows programmes are two advantages of utilizing them. As a result, this picture type was chosen to be utilized in our paper. Because the BMP picture is quite huge in size, the pixels in the image are also relatively large. As a result, it has more room for binary codes to be encoded. In this new approach, we test various sizes of BMP pictures to observe the varied amounts of data being contained in the image, to improve the number of characters that may be hidden. Table 1 displays the

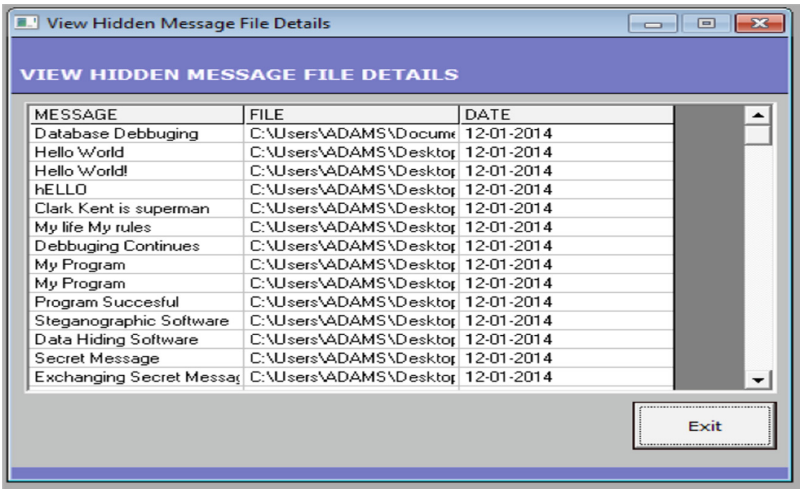

Fig. 16. View message file details interface

various testing findings as well as a comparison of different sizes in a BMP picture using steganography software. As cover pictures, these BMP images are utilized.

Table 1. Comparison of different sizes in bitmap images.

File sizes				
Cover image	Text file	Stego image	Hide message	Retrieve message
439 KB	4.01 KB	286 KB	Yes	Yes
439 KB	12.1 KB	286 KB	Yes	Yes
1.0 MB	10.4 KB	286 KB	Yes	Yes
1.0 MB	10.5 KB	286 KB	Yes	Yes
3.47 MB	12.1 KB	286 KB	Yes	Yes
3.47 MB	27.0 KB	286 KB	Yes	Yes
3.47 MB	54.1 KB	286 KB	Yes	Yes
6.74 MB	54.1 KB	286 KB	Yes	Yes
9.9 MB	334 KB	286 KB	Yes	Yes
9.9 MB	335 KB	286 KB	Yes	Yes

As a result of the new system's examination about various photos, we can infer that the method appears to downsize any size of bmp image from its original size to 286 KB, which greatly aids in data un-detection when sent over networks. The data concealment programme will hide the messages to be transmitted within an image file. This file may then be forwarded to the recipient by attaching it to the current message service. The

receiver uses his copy of the data hiding software to read the concealed message in the image file after getting it as a mail attachment.

6 Conclusion

In this paper, a new system was created by utilizing Microsoft Visual Basic Programming Language and the Microsoft Access database to handle the system's data. Moreover, a suggested Data Hiding System's design was thoroughly explored, and the Donathan Hutchings technique was employed in implementation. The successful implementation of this paper ensures that the results of this research can be used by other researchers in the field of data information security; the benefits of the proposed strategies are illustrated through experimental data which increases the level of confidentiality of information exchange over the internet; access to information by malicious individuals such as hackers will no longer be possible because they will be unable to determine which form of media was used to hide the information; and government monitoring will no longer be possible because malicious individuals such as hackers will be unable to determine which form of media was used to hide the information. Therefore, the documentations in this paper are strongly suggested for comparable steganography data concealment studies. Finally, there were several suggestions made such as efforts to construct a steganography data concealment system in the future.

References

1. Aminu Muazu, A., Ismaila Audi, U.: Network configuration by utilizing cisco technologies with proper segmentation of broadcast domain in FNAS-UMYUK Nigeria. J. Netw. Secur. Data Min. **4**(1), 1–13 (2021). https://doi.org/10.5281/zenodo.4776375
2. Maiwada, U.D., Muazu, A.A., Yakasai, I.K., Zakari, R.Y.: Identifying actual users in a web surfing session using tracing and tracking. JINAV: J. Inform. Vis. **1**, 36–43 (2020). https://doi.org/10.35877/454ri.jinav174
3. Maiwada, U.D., Muazu, A.A., Yakasai, I.K.: Using LTE-sim in new hanover decision algorithm for 2-tier macrocell-femtocell LTE network. Int. J. Comput. Inform. Technol. (2279-0764) **9**(4), 78–83 (2020). https://doi.org/10.24203/ijcit.v9i4.20
4. Muazu, A.A., Aminu, B., Kamal, A.: Design and implementation of android quiz app-in higher institution of learning: Umaru MUSA Yar'Auda University Katsina, Nigeria. In: AICTTRA 2018 Proceeding, pp. 67–71 (2018)
5. Alsewari, A.A., Mu'aza, A.A., Rassem, T.H., Tairan, N.M., Shah, H., Zamli, K.Z.: One-parameter-at-a-time combinatorial testing strategy based on harmony search algorithm OPAT-HS. Adv. Sci. Lett. **24**(10), 7273–7277 (2018). https://doi.org/10.1166/asl.2018.12927
6. Muazu, A.A., Maiwada, U.D.: PWiseHA: application of harmony search algorithm for test suites generation using pairwise techniques. Int. J. Comput. Inform. Technol.(2279-0764) **9**, 91–98 (2020). https://doi.org/10.24203/ijcit.v9i4.23
7. Muazu, A.A., Hashim, A.S., Sarlan, A.: Application and adjustment of "don't care" values in t-way testing techniques for generating an optimal test suite. J. Adv. Inform. Technol. **13**, 347–357 (2022). https://doi.org/10.12720/jait.13.4.347-357
8. Arora, H., Soni, G.K., Kushwaha, R.K., Prasoon, P.: Digital image security based on the hybrid model of image hiding and encryption. In: 2021 6th International Conference on Communication and Electronics Systems (ICCES) (2021). https://doi.org/10.1109/icces51350.2021.9488973

9. Bandyopadhyay, S.K., Roy, S.: Information security through data encryption and data hiding. Int. J. Comput. Appl.. **4**, 32–35 (2010). https://doi.org/10.5120/874-1235
10. Berinato, S.: The Rise of Anti-Forensics | CSO Online, https://www.csoonline.com/article/2122329/the-rise-of-anti-forensics.html. Accessed 1 2022
11. Umar, D.M., Aminu, A.M., Nadzira, N.: The security paradigm that strikes a balance between a holistic security mechanism and the wsn's resource constraints. East Asian J. Multi. Res. **1**, 343–352 (2022). https://doi.org/10.55927/eajmr.v1i3.102
12. Wahab, O.F.A., Khalaf, A.A.M., Hussein, A.I., Hamed, H.F.A.: Hiding data using efficient combination of rsa cryptography, and compression steganography techniques. IEEE Access **9**, 31805–31815 (2021). https://doi.org/10.1109/access.2021.3060317
13. Kuo, W.-C., Lai, P.-Y., Wuu, L.-C.: Adaptive reversible data hiding based on histogram. In: 2010 10th International Conference on Intelligent Systems Design and Applications (2010). https://doi.org/10.1109/isda.2010.5687102
14. Ma, K., Zhang, W., Zhao, X., Yu, N., Li, F.: Reversible data hiding in encrypted images by reserving room before encryption. IEEE Trans. Inf. Forensics Secur. **8**, 553–562 (2013). https://doi.org/10.1109/tifs.2013.2248725
15. Wu, X., Weng, J., Yan, W.: Adopting secret sharing for reversible data hiding in encrypted images. Signal Process. **143**, 269–281 (2018). https://doi.org/10.1016/j.sigpro.2017.09.017
16. Ren, H., Niu, S., Wang, X.: Reversible data hiding in encrypted images using POB number system. IEEE Access **7**, 149527–149541 (2019). https://doi.org/10.1109/access.2019.2946929
17. Yin, Z., Xiang, Y., Zhang, X.: Reversible data hiding in encrypted images based on multi-MSB prediction and huffman coding. IEEE Trans. Multimedia **22**, 874–884 (2020). https://doi.org/10.1109/tmm.2019.2936314