An Efficient Dual Text Steganographic Approach: Hiding Data in a List of Words

Monika Agarwal

Abstract In this paper, we present a novel approach in text steganography using a list of words to hide the secret message. The approach uses the ASCII value of characters and conceals secret message without altering the cover file. The hash value of a message is computed and appended at the end of the message. To provide security, both the message and the appended hash value are encrypted using the proposed encipher algorithm and then the resulting cipher text is embedded inside a cover file using the proposed hide algorithm. After embedding, the stego file, which consists of a list of words, is sent to the receiver. At the receiver side, message authentication and integrity can be verified by applying the same hash function to the deciphered message. The security of the approach is equivalent to the most secure one time pad cryptosystem. We also present an empirical comparison of the proposed approach with some of the existing approaches and show that our approach outperforms the existing approaches.

Keywords Cryptography · Information hiding · Steganography · Text steganography

1 Introduction

Steganography is the art and science of hiding a message inside another message without drawing any suspicion to others so that the message can only be detected by its intended recipient [1]. It is not enough to simply encipher the traffic, as criminals detect, and react to, the presence of encrypted communications [2]. But when steganography is used, even if an eavesdropper gets the stego object,

M. Agarwal (✉)
Department of Computer Science and Engineering,
PDPM-Indian Institute of Information Technology Design and Manufacturing,
Jabalpur, India
e-mail: peace1287@gmail.com

© Springer Science+Business Media New York 2013

he cannot suspect the communication since it is carried out in a concealed way. Steganography, is derived from a finding by Johannes Trithemus (1462–1516) entitled “Steganographia” and comes from the Greek (στεγανό-ς, γραφ − ειν) defined as “covered writing” [3–5]. Steganography gained importance because the US and British government banned the use of cryptography including international mailing of chess games, encrypted mails, clippings from newspapers, etc [6].
Modern steganography is generally understood to deal with electronic media rather than the physical objects and texts [7]. In steganography, the text which we want to conceal is called embedded data, and the text, image, audio, or video file which is used as a medium to hide the text is called cover. The key used (optional) is called stego-key. The resulting object after hiding the data in the cover medium is called stego-object which is transmitted along with the stego-key to its intended recipient. In text steganography, character based text is used to conceal the secret information [8]. Storing text files require less memory and its easier communication makes it preferable to other types of steganographic methods [9].
This paper presents a new approach in text steganography by hiding a message in a list of words. This method works on the ASCII value of a character rather than bits. Hash value of the message is computed and appended at the end of the message and then the message is encrypted using the proposed algorithm. Then for hiding each letter of the encrypted message, a word of certain length is used. At the receiver side, the steps can be performed in reverse order to get back the original message.
The rest of the paper is organized as follows: Sect. 2 describes some of the existing approaches of text steganography. In Sect. 3, the proposed approach is described. Section 4 shows the results of comparison of the proposed approach with the existing approaches. In Sect. 5, we discuss the merits and demerits of the proposed approach and other related issues. Section 6 draws the conclusions.

2 Existing Approaches

In this section, we present some of the popular approaches of text steganography.

2.1 Line Shift

In this method, the secret message is hidden by vertically shifting the text lines to some degree [10, 11]. A line marked has two unmarked control lines one on either side of it for detecting the direction of movement of the marked line [12]. To hide the bit 0, the line is shifted up and to hide the bit 1, the line is shifted down [13].

2.2 Word Shift

In this method, the secret message is hidden by shifting the words horizontally, i.e. left or right to represent the bit 0 or 1 respectively [13].

2.3 Syntactic Method

This technique uses punctuation marks such as full stop (.), comma (,), etc. at proper places to hide the bits 0 and 1 [10, 11, 14].

2.4 White Steg

This technique uses white spaces for hiding the secret message. For example, one space after a word represents the bit 0 and two spaces after a word represents the bit 1 [3, 5, 14].

2.5 Spam Texts

In XML and HTML files, the bit 0 is represented by a lack of space in the tag and the bit 1 is represented by inserting a space inside the tag [13].

2.6 SMS-Texting

In SMS-Texting, to hide the bit 0, full form of the word is used and the bit 1 is hidden by using the abbreviated form of that word [15].

2.7 Feature Coding

In feature coding, the secret message is hidden by altering one or more features of the text [13]. OCR program or re-typing can destroy the hidden information [6, 16].

2.8 SSCE (Secret Steganography Code for Embedding)

This technique first encrypts the secret message using SSCE table and then embed the resulting cipher text in a cover file by inserting articles a or an with the non specific nouns in English language using a certain mapping technique [8].

2.9 Word Mapping Method

This technique first encrypts the secret message using genetic operator crossover and then embed the resulting cipher text in a cover file by inserting blank spaces between words of even or odd length using a certain mapping technique [9].

2.10 MS Word Documents

This technique degenerates the text segments of a document and the secret message is embedded in the choice of degenerations which are then revised with the changes being tracked [17].

2.11 Cricket Match Scorecard

In this method, data is hidden in a cricket match scorecard by pre-appending a meaningless zero before a number to represent bit 1 and leaving the number as it is to represent bit 0 [18].

2.12 CSS (Cascading Style Sheet)

This technique first encrypts the data using RSA public key cryptosystem and then the resulting cipher text is embedded through a Cascading Style Sheet (CSS). A space after a semicolon embeds bit 0 and a tab after a semicolon embeds bit 1 [19].

3 The Proposed Approach

The proposed method hides a message in a list of words. Each letter is hidden in a word of certain length. First, a hash value of the message is computed and appended at the end of the message. The resulting file is then encrypted using a one-time secret key of nearly true random numbers. Then for hiding each character of the cipher text, a word of certain length is taken. The starting letter of a word is determined by masking the sum of digits of the ASCII value of the character to be hidden to an English alphabet. If the sum of digits is 1, then starting letter will be 'a'; if it is 2, then 'b' and so on. At the receiver side, the steps are performed in reverse order to get back the original message. Message authentication and integrity can be verified by calculating hash value of the deciphered message using the same hash function and matching its value against the appended hash value in the deciphered message. Figure 1 shows the proposed model of text steganography. The proposed model has five building blocks: Hash function, which generates a hash code of the message, Encipher function, which scrambles the message using a one-time secret key, Hide function, which hides the encrypted message using a stego key, Seek function, which extracts the hidden information from the stego file using the same stego key, and, Decipher function, which deciphers the extracted message using the same one-time secret key. In all algorithms, we have assumed integer division.

3.1 The Hash Algorithm

1. Get the binary value (a) of the first character of the message.
2. Initial hash value, h = CLS (a), where CLS is circular left shift by one bit.
3. Read next character of the message and convert it to its binary equivalent (b).
4. r = XOR(h, b), where XOR is exclusive OR operation.
5. The next hash value, h = CLS(r).
6. Repeat steps 3–5 till the end of the message.

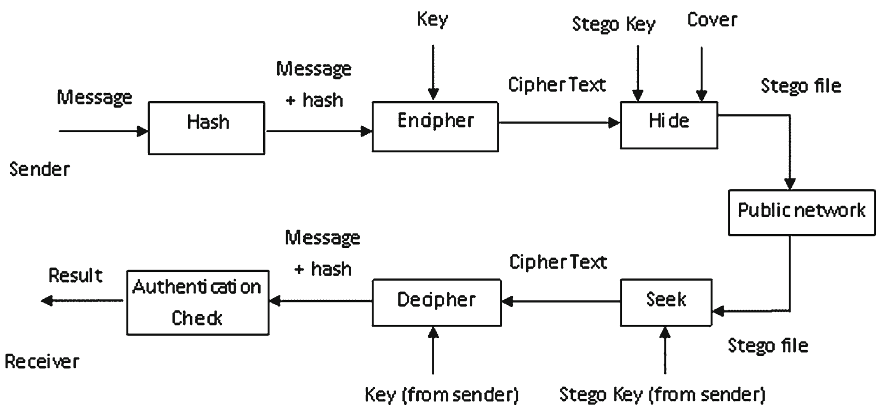

Fig. 1 The proposed model of text steganography

7. Set the most significant bit of h to 0 and then convert it to its decimal representation (n).
8. If $n < 30$, then $n = n + 100$.
9. Convert n to its character equivalent and append it at the end of the message.

3.2 The Encipher Algorithm

1. Fill an array A, of size 1000, with random numbers in the range 0–255.
2. Scan a character from the input file and get its ASCII value (n).
3. Generate a random index (i) to the array A and access the value at that index, $r = A[i]$. Write r in the secret key file.
4. Replace the value (A[i]) with a new random value in the same range.
5. Get s as the sum of square of digits of random number, r.
6. Compute x and y as follows,

$$x = s/10.$$

$$y = s(\mathrm{mod}10).$$

7. The scrambled value (e),

$$e = n - (x * y) + r.$$

8. Write character equivalent of e in cipher text file.
9. Execute steps 2–8 repeatedly till the end of the input file.
10. The secret key is sent to the receiver.

3.3 The Hide Algorithm

1. Calculate length (le) of the input file.
2. Read a letter from the cipher text and get its ASCII value (n).
3. If $n < 100$, then prefix it with zeroes to make it a three digit number.
4. Calculate k as the most significant digit of n and write it in the stego key file.
5. Length of the word, $l =$ the middle digit of n.
6. If $l < 6$, then $l = l + 10$.
7. Compute s as the sum of digits of n.
8. Take a word of length l starting from the sth letter in the English alphabet and write it in the stego file.
9. Repeat steps 2 to 8 till the end of the cipher text file.
10. If the length (le) of the cipher text <10, insert 10-le ten letter words in the stego file.

3.4 The Seek Algorithm

1. Read a value (k) from the stego key file.
2. Read a word from the stego file and get its length (l).
3. If $l > 9$, then $l = l - 10$.
4. Compute s by decoding the first letter of the word from the English alphabet.
5. $r = s - (l + k)$.
6. The ASCII value,

$$n = (k^{*}100) + (l^{*}10) + r.$$

7. Convert n to its character equivalent.
8. Repeat above steps till the end of the stego key file.

3.5 The Decipher Algorithm

1. Scan a character from the cipher text file and get its ASCII value, e.
2. Get a value from the secret key (r).
3. Get s as the sum of square of digits of r.
4. Compute x and y as follows,

$$x = s/10.$$

$$y = s(\bmod 10).$$

5. ASCII value of the character, n, is calculated as,

$$n = e + (x^{*}y) - r.$$

6. Convert n to its character equivalent.
7. Execute above steps repeatedly till the end of the cipher text file.

3.6 An Example

Consider the message “pascal” to be hidden in a cover file. The cipher text of the message and its appended hash value comes out to be “ídŔÚÇ H”. Figure 2 shows the stego file after hiding the cipher text.

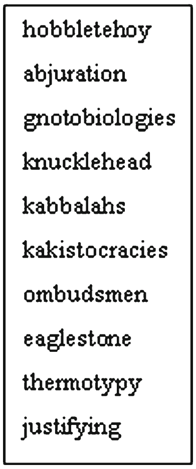
Fig. 2 The stego file

hobbletehoy
abjuration
gnotobiologies
knucklehead
kabbalahs
kakistocracies
ombudsmen
eaglestone
thermotypy
justifying

4 Experimental Analysis and Results

In this section, we present an experimental comparison of the various text steganographic approaches based on capacity ratio. The capacity ratio is computed by dividing the amount of hidden bytes over the size of the cover text in bytes [20].

Capacity ratio = (amount of hidden bytes)/(size of the cover text media in bytes).

Assuming that one character takes one byte in memory, we have calculated the percentage capacity which is capacity ratio multiplied by 100.
The samples of embedded data used are:

1. Ego (3 byte)
2. Minute (6 byte)
3. Hello World! (12 byte)
4. Failure is never final ! (24 byte)
5. Smile is an inexpensive way to improve your looks. (50 byte)
6. Its not the load that breaks you down, its the way you carry it. (63 byte)
7. Don't find hundred reasons why you can't do a thing, but just find one reason why you can and do it. (100 byte)
8. Tide recedes and leaves behind bright sea shells on sand
   Sun sets but its warmth lingers on land
   Music stops and its echoes on in sweet refrains
   For every joy that passes, something beautiful remains (202 byte)
9. Steganography is not a new area. It dates back to 5th century BC. Harpagus used hare to send his message by killing it and hiding the message inside its belly. A person disguised as hunter carried the hare to the destination. Another incident

was of King Darius of Susa. Histiaeus was assigned the duty of shaving the head of his most trusted slave (349 byte).

10. Steganography is not a new area. It dates back to 5th century BC. Harpagus used hare to send his message by killing it and hiding the message inside its belly. A person disguised as hunter carried the hare to the destination. Another incident was of King Darius of Susa. Histiaeus, prisoner of Darius, was assigned the duty of shaving the head of his most trusted slave and then the message was tattooed on his shaved scalp. After some time, when the hairs of the slave grew back, his head was shaved again (508 byte).

Table 1 shows the percentage capacity obtained when our method is applied to the ten experimental samples. We see that for messages having size less than 10 bytes, percentage capacity is low. After 10 bytes, percentage capacity increases and becomes somewhat constant. The rationale behind it is that, in the proposed approach, the stego file consists of at least 10 words regardless of the size of the message. One word can hide one character. If we want to hide the message "eng", which is of three bytes, three words are required. But, a stego file of three words will raise suspicion. Therefore, minimum size of the stego file is 10 words and hence the percentage capacity is low for messages having size less than 10 bytes. Also, the stego file depends on the ASCII value of characters to be hidden. Since we are concerned with concealing text messages with ASCII value in the range 30–129 and the range of random number is 0–255, therefore, the value of encrypted character falls in the range 1–380. Table 2 shows the average percentage capacity of the approaches when applied to the experimental samples. We observe that the average percentage capacity of our method is greater than the other approaches.

5 Discussion

There are three main issues to be considered when studying steganographic systems: capacity, security, and robustness. Capacity refers to the ability of cover media to store secret data. Security refers to the ability of an eavesdropper to suspect hidden

Table 1 Percentage capacity of the proposed approach over the ten experimental samples

I	II	III	IV	V	VI	VII	VIII	IX	X
3.8	6.7	9.0	9.1	8.9	8.3	8.6	8.8	9.0	8.8

Table 2 Comparison of the av. percentage capacity (>1) of approaches

List of words	White steg	SMS texting	Feature coding	Word mapping	Spam text	Word shift
8.162	1.874	1.71	1.479	1.464	1.164	1.03

information easily. Robustness refers to the ability of protecting the unseen data from modification or corruption [20].

As evident from the experimental results, the average percentage capacity of the proposed approach is much higher than the existing approaches. The stego file does not contain any unnecessary white space. So, if the file is opened with a word processor program, it does not have the possibility of raising suspicion regarding the presence of the concealed information. In case of very small messages (size less than ten bytes), a stego file of three or four words will raise suspicion. Therefore, at least ten words are there in a stego file regardless of the message size. To make it even more secure, it is used in conjunction with cryptography by encrypting a message before concealing it.

Generally, random numbers used by an algorithm are pseudo random numbers. For cryptographic purposes, we need more secure random numbers. True random numbers depend on the environmental conditions such as radioactive decay or CPU temperature, etc. and hence cannot be generated by any hardware or software. So, the encipher algorithm generates a key comprising of nearly true random numbers. Further, the length of the secret key equals the length of the message and the key is discarded after each encryption and decryption. For scrambling the same message again, a different key will be used and thereby resulting in a different cipher text for the same plain text. Thus, if an eavesdropper gets the cipher text or stego file, there is no fixed pattern which he can analyze to figure the hidden content. Hence, the security of the proposed encipher algorithm is equivalent to the one-time pad cryptosystem.

Since no font changing techniques have been applied, the stego file can withstand OCR techniques and retyping does not lead to the loss of the concealed content. Also, the words in a stego file are meaningful and there are no special characters, no extra spaces, and no misspellings. Therefore, the proposed approach is robust. A Limitation of the proposed approach is that it can successfully hide the secret message consisting of characters having ASCII value in the range 30–129, i.e., English alphabets, numbers, and some special characters but we have not considered the entire Unicode list.

6 Conclusion

Steganography although conceals the existence of a message but is not perfectly secure. It is not meant to supersede cryptography but to supplement it. This paper presents a novel approach in text steganography where each character of the embedded data is hidden in a word of cover file without involving any degradation of the cover file. The observed average percentage capacity of the proposed approach is found to be much higher than the existing approaches. The reason behind it is that the approach works on the ASCII value of characters rather than their binary value. The approach is secure because the stego file does not contain any special characters or white spaces and there are no misspelled words. Therefore, opening the stego file with a word processor program will not draw suspicion regarding the existence of concealed content. As no change is done to the cover file while embedding, the cover and stego file are exactly the same. The approach allows further security by scrambling a message, using the one-time pad scheme, before concealing it and thereby leading the message security equivalent to the most secure one-time pad cryptosystem. Apart from this, message integrity can be verified by the use of the hash function. As no font changing techniques have been used, the stego file can withstand OCR techniques and retyping does not lead to the loss of the hidden information. Hence, the proposed approach is robust. The proposed approach can be used to securely transmit sensitive information like PIN, user passwords.

References

1. Changder S, Ghosh D, Debnath NC (2010) Linguistic approach for text steganography through Indian text. In: 2010 2nd international conference on computer technology and development, pp 318–322
2. Anderson RJ, Petitcolas FAP (1998) On the limits of steganography. IEEE J Sel Areas Commun 16(4):474–481
3. Por LY, Delina B (2008) Information hiding—a new approach in text steganography. In: 7th WSEAS international conference on applied computer and applied computational science. Hangzhou China, pp 689–695
4. Petitcolas FAP, Anderson RJ, Kuhn MG (1999) Information hiding—a survey. In: Proc IEEE 87(7):1062–1078
5. Por LY, Ang TF, Delina B (2008) WhiteSteg-a new scheme in information hiding using text steganography. WSEAS Trans Comput 7(6):735–745
6. Rabah K (2004) Steganography-the art of hiding data. Inf Technol J 3(3):245–269
7. Benett K (2004) Linguistic steganography-survey, analysis and robustness concerns for hiding information in text. CERIAS technical report 2004-13
8. Banerjee I, Bhattacharyya S, Sanyal G (2011) Novel text steganography through special code generation. In: International conference on systemics, cybernetics and informatics, pp 298–303
9. Bhattacharyya S, Banerjee I, Sanyal G (2010) A novel approach of secure text based steganography model using word mapping method. Int J Comput Inf Eng 4(2):96–103
10. Shahreza MHS, Shahreza MS (2008) A new dynonym text steganography. In: International conference on intelligent information hiding and multimedia signal processing, pp 1524–1526
11. Shahreza MHS, Shahreza MS (2006) A new approach to persian/arabic text steganography. In: 5th IEEE/ACIS international conference on computer and information science and 1st IEEE/ACIS international workshop on component-based software engineering, software architecture and reuse, pp 310–315
12. Brassil JT, Low SH, Maxemchuk NF, O'Gorman L (1995) Document marking and identification using both line and word shifting. In: Proceedings of INFOCOM '95 proceedings of the fourteenth annual joint conference of the IEEE computer and communication societies, pp 853–860
13. Cummins J, Diskin P, Lau S, Parlett R (2004) Steganography and digital watermarking. School of Computer Science, pp 1–24
14. Bender W, Gruhl D, Morimoto N, Lu A (1996) Techniques for data hiding. IBM Syst J 3(3&4):313–336
15. Shahreza MS, Shahreza MHS (2007) Text steganography in SMS. In: 2007 international conference on convergence information technology, pp 2260–2265

16. Brassil JT, Low S, Maxemchuk NF, O'Gorman L (1995) Electronic marking and identification techniques to discourage document copying. IEEE J Sel Areas Commun 1(8):1495–1504
17. Liu T-Y, Tsai W-H (2007) A new steganographic method for data hiding in microsoft word documents by a change tracking technique. IEEE Trans Inf Forensics Secur 2(1):24–30
18. Khairullah M (2011) A novel text steganography system in cricket match scorecard. Int J Comput Appl 21(9):43–47
19. Kabetta H, Dwiandiyanta BY (2011) Suyoto: information hiding in CSS: a secure scheme text steganography using public key cryptosystem. Int J Cryptogr Inf Secur 1(1):13–22
20. Haidari FA, Gutub A, Kahsah KA, Hamodi J (2009) Improving security and capacity for arabic text steganography using "Kashida" extensions. In: 2009 IEEE/ACS international conference on computer systems and applications, pp 396–399