Summer 7-19-2017

Highly Imperceptible and Reversible Text Steganography Using Invisible Character based Codeword

Mohammad Saidur Rahman
RMIT University, Melbourne, mohammadsaidur.rahman@rmit.edu.au

Ibrahim Khalil
RMIT University, Melbourne, ibrahim.khalil@rmit.edu.au

Xun Yi
RMIT University, Melbourne, xun.yi@rmit.edu.au

Hai Dong
RMIT University, Melbourne, hai.dong@rmit.edu.au

Highly Imperceptible and Reversible Text Steganography Using Invisible Character based Codeword

Completed Research Paper

Mohammad Saidur Rahman
School of Science, RMIT University
Melbourne, Victoria, Australia
mohammadsaidur.rahman@rmit.edu.au

Ibrahim Khalil
School of Science, RMIT University
Melbourne, Victoria, Australia
ibrahim.khalil@rmit.edu.au

Xun Yi
School of Science, RMIT University
Melbourne, Victoria, Australia
xun.yi@rmit.edu.au

Hai Dong
School of Science, RMIT University
Melbourne, Victoria, Australia
hai.dong@rmit.edu.au

Abstract

Text steganography method can be applied for protecting privacy and authenticity of text-based documents. Text steganography is a challenging task as slight modification in text file can be identified. In general, imperceptibility of text steganography is very poor. Additionally, data hiding capacity of text steganography is very less. Current research works fail to solve both imperceptibility and capacity problems simultaneously. We propose a novel data compression based reversible text steganography scheme addressing both imperceptibility and capacity problems of text steganography. Our proposed method embeds secret message within text file as invisible character by following a set of embedding rules. We present experimental results that demonstrate higher imperceptibility and capacity of proposed method without increasing size of text file containing secret information. Most importantly, our proposed method successfully retrieves secret information and reconstructs original text file without any error.

Keywords: Text steganography, Huffman coding, security, privacy preservation

Introduction

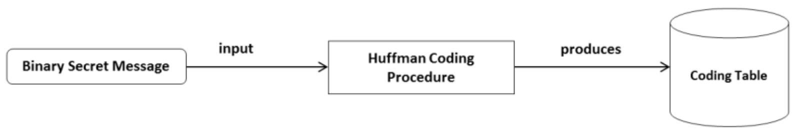
Recent improvement in Information and Communication Technology (ICT) produces large volume of text data related to customers, suppliers and operations owned by business organizations. Memos, business agreements, emails, reports and customer records are few examples of text-based business data. Managing large volume of text-based data by their own is expensive for business organizations. Cloud based management of large volume data minimizes the cost of data storage. Additionally, cloud offers scalable storage with efficient management of data produced by organizations. Hence, business organizations are outsourcing their big data to third-party cloud for storing and managing (Lu, 2013). Data stored in the cloud are sensitive. Outsourcing data to cloud arises privacy concerns for data owners (Hashem, 2015; Lu, 2014). For instance, stored data in the cloud can be compromised, and individual personal information can be disclosed (Lu, 2014). Several cryptography methods are being used for protecting privacy and authenticity of sensitive text-based data in cloud. However, operations over cryptographically encrypted data are usually complex and time consuming (Lu, 2014). Moreover, meaningless form of encrypted data may attract the attention of intruders (Chang, 2010). Alternatively, information hiding technique such as *steganography* can be used for provisioning privacy and authenticity of secret information.

Steganography is an art of information hiding that provides secret communication over public channel that allows avoidance of unauthorized users' attention (Artz, 2001). It is mainly used for sharing secret information. However, authenticity of a communication can be ensured as well by applying *steganography*. For example, digital signatures of two business parties can be hidden inside a business agreement through a *text steganography* method to produce a digitally *signed business agreement*. The signed business agreement can be stored in cloud. Later, the same text steganography method can be applied to check authenticity of the stored business agreement.

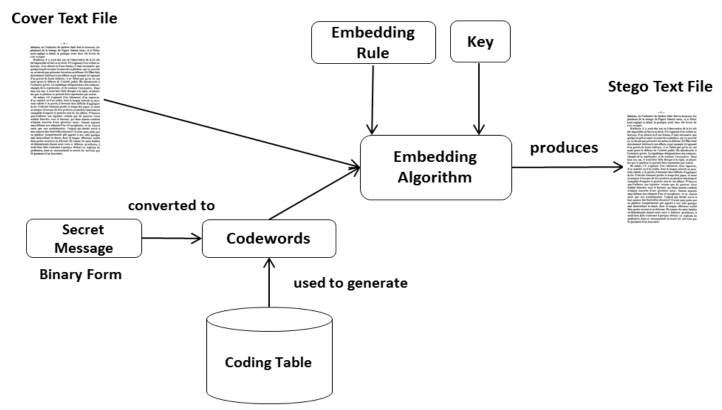
In general, steganography method masks a secret message in a *cover medium*. The cover medium can be any digital media such as text (Jin'An, 2008; Lee, 2013; Por, 2012), image (Subhedar, 2014), signal (Abuadbba, 2016), audio and video. A *stego medium* is produced as a result of masking secret message in cover medium. The function that is used to hide secret message in cover medium is called *embedding* procedure. *Extraction procedure* retrieves the hidden secret message from stego medium. Important requirements of a steganography method are *imperceptibility*, *robustness* and *capacity* (Zaker, 2012). Imperceptibility is a measure of security of steganography method that determines the level of perception of existence of a secret message in a cover file. Robustness denotes the level of ability to resist alteration of secret message. Capacity refers to amount of data that can be hidden in a cover medium. A *reversible* or *lossless* steganography approach can take out a secret message from stego medium and reconstruct cover medium without any alteration. Therefore, the primary goals of steganographic approaches are high embedding capacity and low distortion of the cover medium (Lee, 2013).

In this research work, we develop a text-based steganography approach. The text-based steganography is challenging since a slight modification in text content can be easily discovered (Lee, 2013). Hence, the imperceptibility of text cover medium is very poor. Several research works are done on text based steganography. Research works in (Jin'An, 2008; Yu, 2008) converts the text files into digital images and embed secret message by adjusting spacing of letters, words or lines. As a result, text file cannot be modified later. The work in (Por, 2012) hides secret message as invisible character in Microsoft Word file. The work is not applicable for other text format. Moreover, embedding secret message increases size of stego text file than cover text file. Few compression based techniques are proposed in (Chen, 2010; Lee, 2013; Satir, 2012; Satir, 2014) that compress cover text file before embedding. Imperceptibility of text cover file becomes very less if the cover text file is compressed.

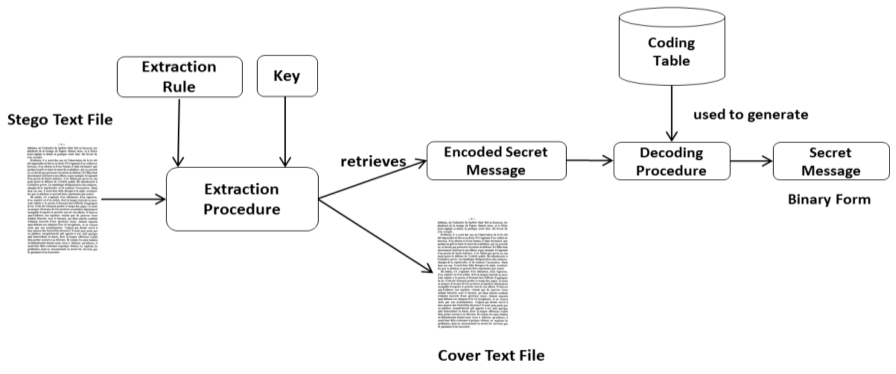
The objective of this research is to develop a highly imperceptible and reversible text steganography method with higher capacity. Our work presents a highly imperceptible and reversible text steganography method based on *Huffman Code* to encode secret message. We use binary format of secret message in our paper. A secret message can be data owner's identification number, digital signature of a business organization, etc. Encoded secret message is embedded into text cover file as invisible characters. Customer records, invoices, and business agreements can be named as text cover mediums. The proposed text steganography approach consists of three parts: (1) generating a *coding table* for a set of symbols, (2) embedding a secret message in a text cover file based on the coding table, and (3) extracting the hidden secret message from the stego text file and reconstruction of cover text file. A coding table is generated using *Huffman Code* that contains symbols, their corresponding frequency and codeword. Each codeword in the coding table has the property that no codeword in the dictionary is a prefix of any other codeword in the dictionary (Huffman, 1952). In our approach, a symbol is a binary block of length n . Therefore, we have 2^n possible binary symbols. If $n = 2$, then number of possible symbols is $2^2 = 4$. Hence, the symbols are $\{00, 01, 10, 11\}$. The coding table is used to embed the secret message. The length of a binary block n is a secret information that need to be decided prior to the embedding operation starts. The length of a block needs to be passed secretly to both embedding and extraction algorithm. The generation process of coding table is shown in Figure 1(a).



(a)



(b)



(c)

Figure 1. Overview of Proposed Text Steganography Approach. (a) Generation of Coding Table using Huffman Coding Technique, (b) Embedding Procedure, and (c) Extraction Procedure

The secret message embedding algorithm hides a secret message inside a text file using few steps (see Figure 1(b)). First, secret message is encoded using the coding table to obtain encoded secret message. Encoding secret message increases the security of the steganography (Satir, 2012;Satir, 2014). Second, embedding algorithm determines the locations of the space characters in the texts of cover text file. Let, total number of locations for SPACE characters is t . Assume that there are k number of binary bits in the secret message, where $k \le t$. Third, k number of locations of space characters is chosen. Location of the space characters are selected randomly based on a *pseudo-random sequence*. The pseudo-random sequence is generated using a secret key. The secret key is used as the *seed* for generating pseudo-random sequence. Same secret key is used to generate pseudo-random sequence during extraction process. Finally, binary bits of encoded secret message are replaced by invisible

Unicode characters based on a rule obtaining Stego text file. A *stego key* is constructed including necessary information that is required to extract secret message.

The extraction algorithm has multiple steps as well (see Figure 1(c)). First, the extraction algorithm identifies the i locations of invisible characters. Second, a pseudo-random sequence of k elements is generated using the same secret key (i.e. seed) that is used in the embedding process. Third, if an element of the sequence is invisible character then the character is replaced by corresponding binary bit and stored as encoded secret bit. The invisible characters are replaced by a SPACE character in the stego text file. As a result, the cover text file is reconstructed from the stego text file with any loss of information. Finally, the extracted encoded secret message is decoded using coding table and secret message is obtained.

Contributions of our work are as follows:

1) The Secret message is embedded as invisible characters. As a result, no alpha-numeric characters in stego text file are distorted. Therefore, imperceptibility of the stego text file becomes higher as the presence of secret message cannot be perceived.
2) Our proposed method successfully reconstructs the cover file after extraction of secret message. Hence, our work is fully reversible or lossless.
3) Sizes of cover text file and stego text file are same in our proposed method. Therefore, payload is zero in our method.
4) The secret message is disguised using Huffman Code. Hence, the length of secret message is reduced and embedding capacity is improved.

The rest of the paper is organized as follows. We briefly describe recent relevant work in *Related Work* section. In next Section, we discuss our proposed highly imperceptible and reversible text steganography method. We show the result of our experiments and discuss the performance of our proposed text steganography method in *Experimental Results and Performance Analysis* Section. Finally, Section *Conclusion* concludes the paper.

Related Work

Several text based steganography schemes have been proposed till date. We describe some of the recent research works related to text steganography approaches. Research works in (Jin'An, 2008;Yu, 2008) converts the text files into digital images first. Later, the secret message is embedded by moderately adjusting the spacing of letters, words, or lines. The disadvantage of converting text files into images is that the texts in cover file cannot be modified further once converted into images.

A degenerating document content based text steganography method is discussed in (Liu, 2007). Text segments are degenerated by inferior writing. Next, the document is revised using change tracking in Microsoft Word documents. The secret message is embedded in the change tracking and the stego file is sent to the receiver. The receiver extracts the secret information by using change tracking in Microsoft Word documents. The limitation of this approach is that large amount of editing rules needs to be stored. Moreover, the hiding capacity is very low. Another document based work in (Por, 2012) that is referred as *UniSpaCh*. The work embeds secret information in Microsoft Word document. UniSpaCh considers combination of inter-word, inter-sentence, end-of-line and inter-paragraph spacing to embed secret information in a document. In UniSpaCh, a two bits combination of secret information is replaced by a Unicode space characters and embedded in the cover document. The mapping of bit combinations and Unicode space characters are shared as a key prior to the communication. In order to improve the security of the steganography method, the secret information is encrypted before data embedding. However, this approach cannot be used for other text file format. Additionally, embedding secret message increases size of stego text file than cover text file.

Compression based text steganographic methods are addressed in (Chen, 2010; Lee, 2013;Satir, 2012;Satir, 2014). The work in (Chen, 2010) hides the secret message in the compression codes using Lempel-Ziv-Welch (LZW) method. This scheme embeds secret data in LZW compression codes by reducing symbol length. Another research work in (Lee, 2013) proposed a lossless text steganography method by Huffman Compression Coding. A modified Huffman Coding, named variable Huffman coding, is used in this work to generate codewords for symbols in the cover file. Secret data is then embedded in the codewords. Research works in (Satir, 2012) and (Satir, 2014) use LZW and Huffman code based compression techniques, respectively. Email is considered as cover medium. Secret message is embedded in texts of email body from the previously constructed text base. Later, email is compressed to produce stego text. The aforementioned compression based steganography methods compress the cover file to increase embedding capacity and reduce transmission cost. However, the disadvantage of compressing cover file is that the imperceptibility of stego file is very poor. As a result, stego file becomes more prone to attack. Moreover, codewords need to be recalculated if the content of the cover file is changed.

Proposed Reversible Text Steganography Approach

We discuss our proposed reversible text steganography approach in this section. We use text file as our cover medium. We assume that we have a text file H that has p characters. Formally, H can be denoted as $H = \{h_1, h_2, \dots, h_p\}$. A secret message M contains m binary bits and denoted by $M = \{b_1, b_2, \dots, b_m\}$. Our proposed scheme has three main steps: (1) encoding secret message, (2) embedding encoded message into cover text file, and (3) extracting encoded secret message and reconstruction of cover text file. The aforementioned steps of our proposed method are described in the following subsections.

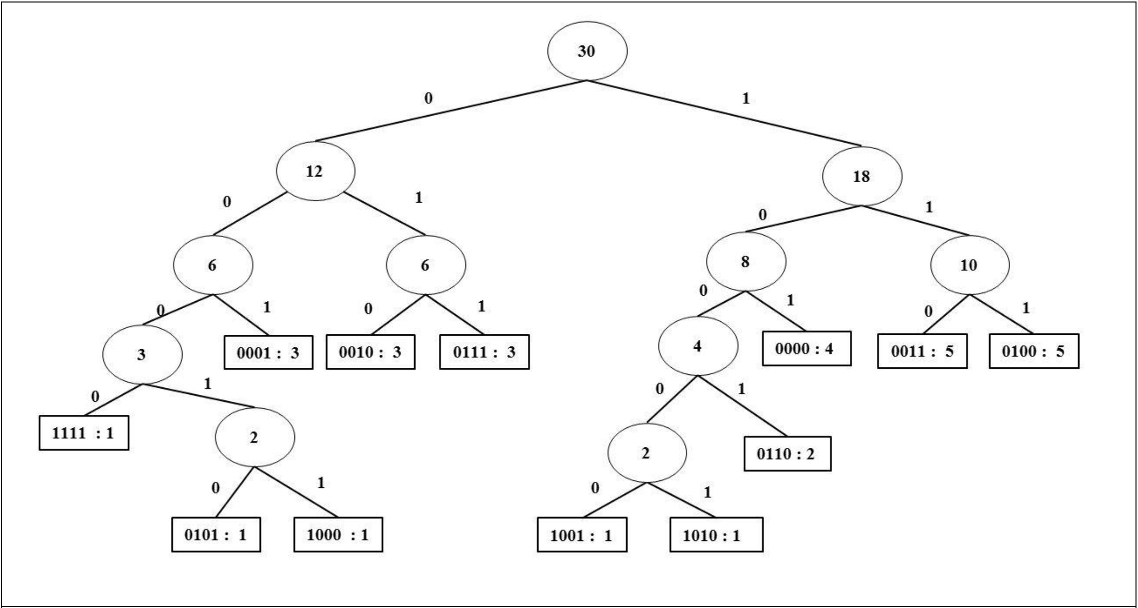


Figure 2. Generated Huffman Tree for Binary Secret Message

Encoding Secret Message

We use Huffman coding (Huffman, 1952) to encode secret message M . Steps of encoding M are described below:

Step-1: Identify set of unique symbols $S = \{s_1, s_2, \dots, s_n\}$ in M and corresponding frequencies $F = \{f_1, f_2, \dots, f_n\}$, where n is the number of unique symbols in M . In order to do this, choose a positive integer d that divides the number of bits m in M (i.e., $d \mid m$) and $2 \le d \le \frac{m}{2}$. Here, d determines the number of bits in a symbol. Therefore, maximum value of n is 2^d .

Step-2: Sort elements s_i of S in increasing order according to their occurrence frequencies f_i in F , where $1 \le i \le n$.

Step-3: Remove the two least frequent elements from S and make each element a *leaf node*. Next, create a new node as a *parent node* of the two leaf nodes with a frequency computed by summing the frequencies of its two child nodes. Two binary bits, 0 and 1 , are then assigned to the *left edge* and *right edge* of the new node, respectively.

Step-4: Insert the new created node and its frequency into set S and F , respectively.

Step-5: Repeat Steps 2-4 until the root node is obtained. Hence, the Huffman tree is generated.

Step-6: Each symbol s_i is assigned a unique binary string (codeword) by traversing the Huffman tree from the root node to each leaf node after constructing the Huffman tree. Formally, the set of codewords is denoted by $W = \{w_1, w_2, \dots, w_n\}$, where n is the number of symbols as well as codewords. A coding Table T is obtained that can be expressed as $T =< S, F, W >$.

Step-7: Substitute the original symbols by corresponding codewords according to T obtaining an encoded secret message $M_e = \{m_{e1}, m_{e2}, \dots, m_{ek}\}$, where m_{ei} is a bit of encoded secret message($1 \le i \le k$) and k is the number of bits of encoded secret message ($k < m$).

Table 1. The Coding Table, T

Symbol	Codeword	Frequency
0000	101	4
0001	001	3
0010	010	3
0011	110	5
0100	111	5
0101	00010	1
0110	1001	2
0111	011	3
1000	00011	1
1001	10000	1
1010	10001	1
1111	0000	1

Table 1. The Coding Table, T

Example 1: Encoding Secret Message

Assume that we have the following binary secret message for encoding using the aforementioned steps:

010000010111010101110100011010000110111101110010001000000100100101000100001110100010
0000001100010011000000011000000110001

The length of binary secret message is $m = 120$. Let, the number of bits in a symbol of the secret message is $d = 4$. Here, $d \mid m$ and $2 \le d \le \frac{m}{2}$. Hence, the maximum number of unique symbols is $2^4 = 16$. The set of unique symbols in binary secret message is $S = \{0000, 0001, 0010, 0011, 0100, 0101, 0110, 0111, 1000, 1001, 1010, 1111\}$. Here, the number of unique symbols in given binary secret message is $|S| = 12$. The set of frequencies of symbols in S are $F = \{4, 3, 3, 5, 5, 1, 2, 3, 1, 1, 1, 1\}$. Resulting Huffman tree for the binary secret message is shown in Figure 2. Next, codeword is obtained for each symbol in S from the Huffman tree and a coding table T is generated. The coding table T is presented in Table 1 containing symbols in binary secret message, their corresponding codewords that is obtained from the Huffman tree and frequencies. Now, each symbol of binary secret message is replaced by corresponding codeword stated in T. For example, symbol '0100' is replaced by '111', '0001' is replaced by '001', and so on. As a result, we get the following encoded secret message for the given secret message:

111001011000100111111001000111001000001101001010111110000111111110100010101011100011101
01110101110001

The encoded secret message has 101 bits, i.e. $k = 101$. Therefore, encoding the binary secret message using Huffman coding reduces secret message size.

Embedding Secret Message

The obtained encoded secret message M_e is embedded in cover text file H to generate stego text file $\overline{H}$. Formally, $\overline{H}$ is denoted as $\overline{H} = \{\overline{h_1}, \overline{h_2}, \dots, \overline{h_p}\}$, where p is the number of characters in $\overline{H}$. The steps of embedding secret message are stated below:

Step-1: Copy all the characters of H into $\overline{H}$.

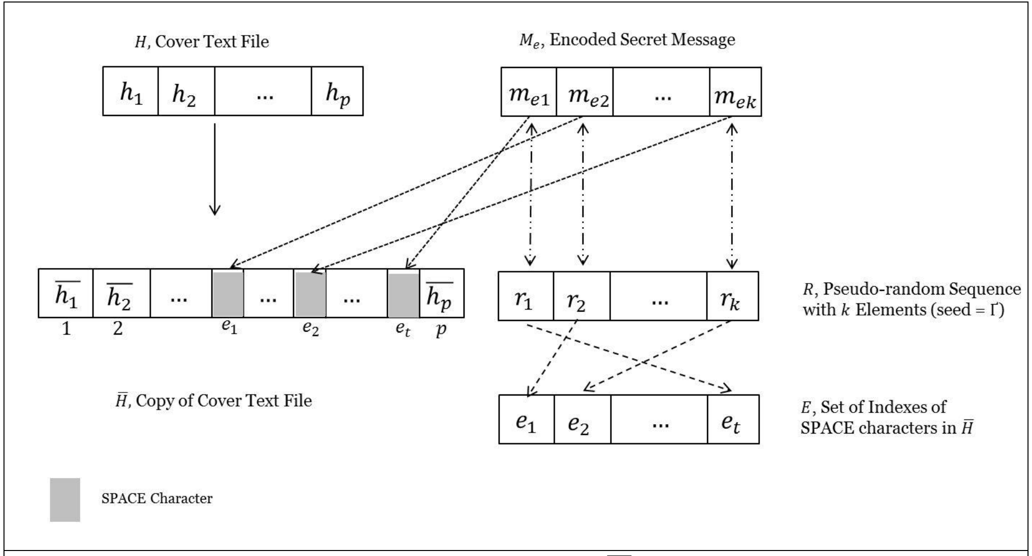
Step-2: Find a set E that contains indexes of SPACE characters in $\overline{H}$. Formally, E is denoted as $E = \{e_j \mid 1 \le j \le t, 1 \le e_j \le p, \overline{h_{e_j}} = SPACE\}$. Here, t is the total number of SPACE characters in $\overline{H}$.

Step-3: A *pseudo-random sequence* R with k elements is generated using a pseudo-random sequence generator seed τ . The value of an element of R must be between 1 and p . Formally, R is denoted as $R = \{r_i \mid 1 \le i \le k, 1 \le r_i \le p\}$. Here, r_i refers to an index of E (i.e., $r_i \in \{j \mid 1 \le j \le t\}$).

Step-4: Each m_{ei} is embedded at $e_j th$ index of $\overline{H}$ such that $r_i \in \{j \mid 1 \le j \le t\}$. Instead of embedding a secret bit directly, an invisible character is embedded. Assume that we have two invisible characters: α and β . In order to embed a secret bit m_{ei} , an invisible character replaces a SPACE character at $\overline{h_{e_j}}$ using the following rule:

$$\overline{h_{e_j}} = \begin{cases} \alpha & if \quad m_{ei} = 0 \\ \beta & if \quad m_{ei} = 1 \end{cases} \tag{1}$$

Finally, the stego text file $\overline{H}$ is obtained.



H, Cover Text File
h_1 h_2 ... h_p
M_e, Encoded Secret Message
m_{e1} m_{e2} ... m_{ek}
$\overline{h_1}$ $\overline{h_2}$ ... ... $\overline{h_p}$
1 2 e_1 e_2 e_t p
$\overline{H}$, Copy of Cover Text File
r_1 r_2 ... r_k
R, Pseudo-random Sequence with k Elements (seed = Γ)
e_1 e_2 ... e_t
E, Set of Indexes of SPACE characters in $\overline{H}$
SPACE Character

Figure 3. Relationship among H , $\overline{H}$, E , R and M_e

Generation of Stego Key

A stego key K is constructed at the end of embedding procedure in order to extract secret message and reconstruct cover text file. The stego key includes coding table (T), number of bits in encoded secret message (k), seed (τ) for generating pseudo-random sequence and values of two invisible characters α and β . Formally, K is defined as a tuple $K =< T, k, \tau, \alpha, \beta >$.

Example 2: Embedding Secret Message

Consider that we have a text file H. We plan to embed encoded secret message M_e within H. Let, number of bits in M_e is $k = 101$. The content of H is copied to another file $\overline{H}$. Next, a set E is generated containing the indexes of space characters in $\overline{H}$. Assume that number of spaces in $\overline{H}$ is 185 ($t = 185$). Hence, The number of elements E is 185 (i.e., $|E| = 185$). A pseudo random sequence R with 101 elements is generated using a seed $\tau = 119$. Here, elements of R range between 1 and 185. An element of R refers to an index of E. For example, the first element of R refers to the 19th index of E, the second element of R refers to the 7th index of E, and so on. Next, a bit m_{ei} of encoded secret message M_e is to be embedded in $\overline{H}$. The first bit '1' of encoded secret message is to be embedded at the index denoted by the 19th element of E. Consider, 19th element of E contains a value 761. Therefore, the first encoded secret bit 1 should be embedded at index 761 of $\overline{H}$. Relationship among H, $\overline{H}$, E, R and M_e is illustrated in Figure 3. Instead of embedding a binary bit, we replace a binary bit by an invisible character according to a rule specified in Equation 1. Consider that we select SPACE and NULL as two invisible characters, i.e., $\alpha = SPACE$ and $\beta = NULL$. If a bit of the encoded secret message is '0', then SPACE character of $\overline{H}$ at a location according to pseudo random sequence is replaced by SPACE. On the other hand, SPACE character of $\overline{H}$ at a location according to pseudo random sequence is replaced by NULL If a bit of the encoded secret message is '1'. Performing the operation for all of the encoded secret bits, $\overline{H}$ becomes the stego text file. After generating the stego file, a stego key $K =< T, k, \tau, \alpha, \beta >$ is constructed as $K =< T, 101, 119, SPACE, NULL >$.

Extracting Secret Message and Reconstruction of Cover text file

The objective of this procedure is to extract the secret message from stego text file $\overline{H} = \{\overline{h_1}, \overline{h_2}, \dots, \overline{h_p}\}$, where p is the number of characters in $\overline{H}$. Additionally, cover text file H is reconstructed in this procedure. Extraction of secret message and reconstruction of H requires stego key K. The steps are discussed below:

Step-1: Find a set $\overline{E}$ that contains the locations of invisible characters in $\overline{H}$. Formally, $\overline{E}$ is defined as $\overline{E} = \{\overline{e_j} \mid 1 \le j \le \overline{t}, 1 \le \overline{e_j} \le q, \overline{h_{e_j}} = \alpha \vee \overline{h_{e_j}} = \beta\}$. Here, $\overline{t}$ is the total number of α and β.

Step-2: A *pseudo-random sequence* $\overline{R}$ with k elements is generated using *seed* τ. $\overline{R}$ is denoted as $\overline{R} = \{\overline{r_i} \mid 1 \le i \le k, 1 \le \overline{r_i} \le q\}$. Here, $\overline{r_i}$ refers to an index of $\overline{E}$ (i.e., $\overline{r_i} \in \{j \mid 1 \le j \le \overline{t}\}$).

Step-3: Create a set $\overline{M_e} = \{\overline{m_{e1}}, \overline{m_{e2}}, \dots, \overline{m_{ek}}\}$ of length k to hold the extracted encoded bits of secret message.

Step-4: Check $\overline{e_j}th$ character of $\overline{H}$ (i.e., $\overline{h_{e_j}}$) where $j = \overline{r_i}$, and insert '0' or '1' at $\overline{m_{ei}}$ based on the following rule:

$$\overline{m_{ei}} = \begin{cases} 0 & if \quad \overline{h_{e_j}} = \alpha \\ 1 & if \quad \overline{h_{e_j}} = \beta \end{cases} \tag{2}$$

Step-5: Substitute $\overline{e_j}th$ character of $\overline{H}$ by SAPCE character if $\overline{h_{e_j}} = \beta$.

Step-6: Repeat Step 4-5 for each $\overline{r_i} \in \overline{R}$ to obtain encoded secret message $\overline{M_e}$ and reconstruct cover text file. $\overline{H}$ is the reconstructed cover text file.

Step-7: Create an empty set $\overline{M}$ to hold decoded secret message bits.

Step-8: Each codeword is identified in $\overline{M_e}$ and decoded using the coding table T. Each decoded symbol is added to $\overline{M}$. Finally, binary secret message $\overline{M}$ is obtained.

Example 3: Extracting Secret Message and Reconstruction of Cover text file

Assume that we have stego key $K =< T, k, \tau, \alpha, \beta >$ as $K =< T, 101, 119, SPACE, NULL >$. A set $\overline{E}$ is created that contains indexes of invisible characters $\alpha = SPACE$ and $\beta = NULL$. The number of elements of $\overline{E}$ is $|\overline{E}| = 185$. Subsequently, a pseudo-random sequence $\overline{R}$ with $k = 101$ elements is created using seed $\tau = 119$. Here, $\overline{R}$ is similar to the pseudo-random sequence R that is generated during embedding procedure due to same seed $\tau = 119$. Elements of $\overline{R}$ range between 1 and 185. An element of $\overline{R}$ refers to an index of $\overline{E}$. Next, a set $\overline{M_e}$ with $k = 101$ elements is created for holding encoded secret bits. Assume that first element of $\overline{R}$ is 19. Hence, character at a location of $\overline{H}$ that is pointed by the 19th element of $\overline{E}$ is checked and embedded bit is retrieved using Equation 2. If the character is SPACE then add '0' to the first index of $\overline{M_e}$. Add '1' to the first index of $\overline{M_e}$ if the character is NULL, and replace NULL character by SPACE character. Same procedure needs to be followed for every element of $\overline{R}$. As a result, we get a set of encoded secret message $\overline{M_e}$ and reconstructed cover text file $\overline{H}$, and $\overline{H} = H$. Afterwards, an empty set $\overline{M}$ is created. Next, each codeword in $\overline{M_e}$ is identified and decoded using coding table T (see Table 1). Finally, we get $\overline{M}$ as binary secret message.

Experimental Results and Performance Analysis

A set of experiments are carried out to evaluate the performance of our proposed text steganography approach in terms of three parameters: *embedding capacity (EC)*, *bit error rate (BER)* and *imperceptibility*. In particular, EC measures *bit rate* that is the size of secret message relative to size of cover text file (Desoky, 2009). EC is calculated as follows:

$$EC = \frac{bits\ of\ secret\ message}{bits\ of\ cover\ text\ file} \tag{3}$$

BER of extracted secret message is a measure that determines the correction capabilities of a steganography approach. BER is the ratio (in percentage) between number of bits that is extracted correctly and number of bits of original secret message (Abuadbba, 2016). BER can be defined as follows:

$$BER = \frac{bits\ retrieved\ with\ error}{Total\ bits} \times 100 \tag{4}$$

Imperceptibility of a steganography method determines the level of perception of the presence of secret message inside a file (Subhedar, 2014). The imperceptibility of a text based steganography is higher if relative changes of characters between cover and stego text file are lower. Experiments are performed on Intel(R) Core(TM) i7-4790 CPU @ 3.60GHz personal computer with 8 GB RAM. Software for the proposed method is developed in Java programming language. A large text file is chosen as cover text file from URL: *http://www.textfiles.com*. Different sizes (in bits) of secret message are examined to evaluate the performance of our proposed approach.

In Table 2, embedding capacity (EC) before and after encoding the secret message has been demonstrated. The size of our sample cover file is 148416 bits. We examine the embedding capacity of our proposed method for multiple secret messages with different length. It is found that embedding capacity is improved after encoding the secret message.

Table 3 illustrates values of bit error rate (BER) of our proposed method for different secret messages. It is found that BER is 0% for each secret message.

In Figure 4, it is demonstrated that the presence of secret message in stego file cannot be perceived. Figure 4(a) and 4(b) represent cover file and stego file respectively. Hence, the imperceptibility of our proposed method is very high.

Table 2. Embedding Capacity of Proposed Method

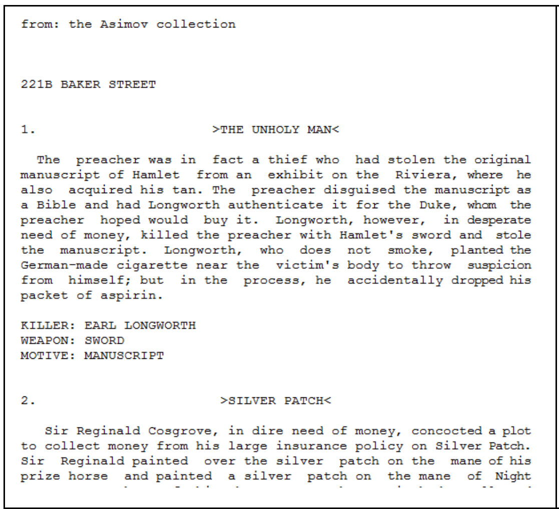
Cover File Size (in bits)	Secret Message Size (in bits)	Encoded Secret Message Size (in Bits)	EC (%) Before Encoding	EC (%) After Encoding
148416	184	154	0.12	0.10
	1048	861	0.70	0.58
	2296	1874	1.54	1.26
	3168	2581	2.13	1.73
	5152	4160	3.47	2.80
	6152	4946	4.14	3.33

Table 2. Embedding Capacity of Proposed Method

Table 3. Bit Error Rate (BER) of Proposed Method

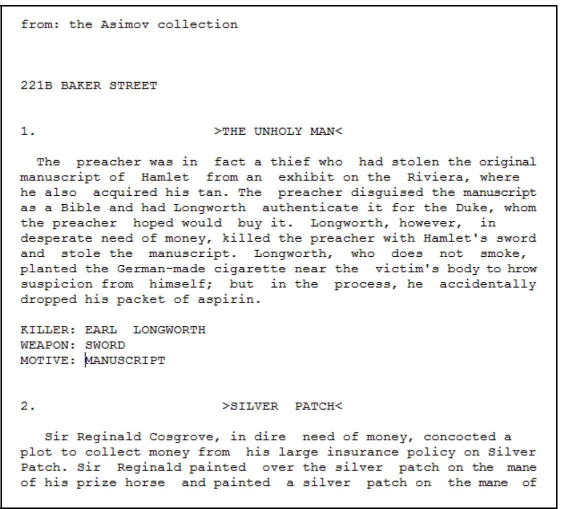
Secret Message Size (in bits)	Bits Retrieved Correctly	BER (%)
184	184	0
1048	1048	0
2296	2296	0
3168	3168	0
5152	5152	0
6152	6152	0

Table 3. Bit Error Rate (BER) of Proposed Method



```
from: the Asimov collection


221B BAKER STREET


1.                    >THE UNHOLY MAN<

  The  preacher was in  fact a thief who  had stolen the original
manuscript of Hamlet  from an  exhibit on the  Riviera, where  he
also  acquired his tan. The  preacher disguised the manuscript as
a Bible and had Longworth authenticate it for the Duke, whom  the
preacher  hoped would  buy it.  Longworth, however,  in desperate
need of money, killed the preacher with Hamlet's sword and  stole
the  manuscript.  Longworth,  who  does  not  smoke,  planted the
German-made cigarette near the  victim's body to throw  suspicion
from  himself; but  in the  process, he  accidentally dropped his
packet of aspirin.

KILLER: EARL LONGWORTH
WEAPON: SWORD
MOTIVE: MANUSCRIPT


2.                     >SILVER PATCH<

   Sir Reginald Cosgrove, in dire need of money, concocted a plot
to collect money from his large insurance policy on Silver Patch.
Sir  Reginald painted  over the silver  patch on the  mane of his
prize horse  and painted  a silver  patch on  the mane  of  Night
```

(a)



```
from: the Asimov collection


221B BAKER STREET


1.                     >THE UNHOLY MAN<

  The  preacher was in  fact a thief who  had stolen the original
manuscript of  Hamlet  from an  exhibit on the  Riviera, where
he also  acquired his tan. The  preacher disguised the manuscript
as a Bible and had Longworth  authenticate it for the Duke, whom
the preacher  hoped would  buy it.  Longworth, however,  in
desperate need of money, killed the preacher with Hamlet's sword
and  stole the  manuscript.  Longworth,  who  does  not  smoke,
planted the German-made cigarette near the  victim's body to hrow
suspicion from  himself;  but  in the  process, he  accidentally
dropped his packet of aspirin.

KILLER: EARL  LONGWORTH
WEAPON: SWORD
MOTIVE: MANUSCRIPT


2.                      >SILVER  PATCH<

   Sir Reginald Cosgrove, in dire  need of money, concocted a
plot to collect money from  his large insurance policy on Silver
Patch. Sir  Reginald painted  over the silver  patch on the  mane
of his prize horse  and painted  a silver  patch on  the mane  of
```

(b)

Figure 4. Demonstration of Imperceptibility of Proposed method. (a) Cover Text File and (b) Stego Text File

It is worthwhile mentioning that size of cover file and stego file in our proposed steganography method are same. Therefore, our steganography method has no payload. Additionally, the percentage of characters of cover file that are successfully reconstructed after extraction of secret messages and the number of characters in cover file is always same (see Figure 5). Hence, our proposed text steganography method is fully reversible.

In Table 4, our proposed method is compared with other mostly relevant research works. Comparison is performed in terms of imperceptibility and payload incurred. Research works in (Chen, 2010;Lee, 2013;Satir, 2012;Satir, 2014) compress cover file to produce stego file. Hence, imperceptibility becomes very poor. On the other hand, our proposed method used compression coding technique to reduce the length of secret message. Therefore, imperceptibility of our proposed method is very high. The work in (Por, 2012) hides secret message as invisible character within words, sentences and paragraphs. The size of stego file is higher than cover file that results in payload. In contrast, our proposed method has 0% payload.

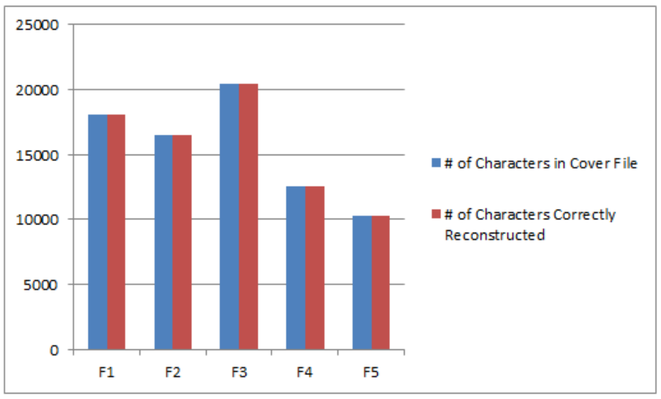


Figure 5. Reversibility of Proposed Method

Table 4. Comparison of Imperceptibility and Payload

Criteria	Method	Discussion
Imperceptibility	(Chen, 2010) (Lee, 2013) (Satir, 2012) (Satir, 2014)	Stego text file is produced by compressing cover file. Therefore, imperceptibility is very poor and prone to attack.
	Our proposed method	Secret message is encoded using compression coding technique and embedded in cover file to produce stego key. Therefore, imperceptibility is very high.
Payload	(Por, 2012)	A block of secret message is embedded as a block of invisible Unicode characters within words, sentences and paragraphs. The size of stego file is higher than that of cover file.
	Our proposed method	A bit of encoded secret message is embedded as an invisible character between words by replacing space character. Payload is 0% as the file sizes of stego file and cover file are same.

Table 4. Comparison of Imperceptibility and Payload

Conclusion

Outsourcing text-based business documents to cloud arises privacy concerns for business organizations. Several cryptography methods are being used for protecting privacy and authenticity of sensitive text-based data in cloud. However, operations over cryptographically encrypted data are usually complex and time consuming. Moreover, meaningless form of encrypted data may attract the attention of intruders. Alternatively, steganography methods can be used to provide privacy and authenticity of outsourced text-based business data. In this paper, we have proposed a novel text steganography technique. We use Huffman coding technique to encode secret message before embedding. As a result, size of secret message is reduced. The secret message is embedded in the cover text file as invisible character without affecting the texts in cover file. Therefore, our proposed method is highly imperceptible. From the experimental results, it is shown that proposed method improves hiding capacity. Additionally, our proposed method incurs no payload to stego file. In other words, sizes of the cover file and stego file are same. Our proposed text steganography scheme successfully retrieves secret information and reconstructs cover text file without any error. As a future work, we plan to improve the hiding capacity while keeping the imperceptibility higher.

References

Artz, D. 2001. "Digital steganography: hiding data within data." IEEE Internet computing **5**(3): 75-80.

Abuadbba, A., et al. 2016. "Resilient to shared spectrum noise scheme for protecting cognitive radio smart grid readings– BCH based steganographic approach." Ad Hoc Networks **41**: 30-46.

Chen, C.-C. and C.-C. Chang. 2010. "High-capacity reversible data-hiding for LZW codes". Computer Modeling and Simulation, 2010. ICCMS'10. Second International Conference on, IEEE.

Chang, C.-C. and T. D. Kieu. 2010. "A reversible data hiding scheme using complementary embedding strategy". Information Sciences **180**(16): 3045-3058.

Desoky, A. 2009. "Listega: list-based steganography methodology." International Journal of Information Security **8**(4): 247-261.

Huffman, D. A. 1952. "A method for the construction of minimum-redundancy codes". Proceedings of the IRE **40**(9): 1098-1101.

Hashem, I. A. T., et al. 2015. "The rise of "big data" on cloud computing: Review and open research issues". Information Systems **47**: 98-115.

Jin'An, L., et al. 2008. "A text digital watermarking for Chinese word document". Computer Science and Computational Technology, 2008. ISCSCT'08. International Symposium on, IEEE.

Lee, C.-F. and H.-L. Chen. 2013. "Lossless text steganography in compression coding". Recent Advances in Information Hiding and Applications, Springer**:** 155-179.

Liu, T.-Y. and W.-H. Tsai. 2007. "A new steganographic method for data hiding in microsoft word documents by a change tracking technique". IEEE Transactions on Information Forensics and Security **2**(1): 24-30.

Lu, C.-W., et al. 2013. "An improvement to data service in cloud computing with content sensitive transaction, analysis and adaptation". Computer Software and Applications Conference Workshops (COMPSACW), 2013 IEEE 37th Annual, IEEE.

Lu, R., et al. 2014. "Toward efficient and privacy-preserving computing in big data era." IEEE Network **28**(4): 46-50.

Lin, H.-Y. and W.-G. Tzeng. 2012. "A secure erasure code-based cloud storage system with secure data forwarding". IEEE transactions on parallel and distributed systems **23**(6): 995-1003.

Por, L. Y., et al. 2012. "UniSpaCh: A text-based data hiding method using Unicode space characters". Journal of Systems and Software **85**(5): 1075-1082.

Subhedar, M. S. and V. H. Mankar. 2014. "Current status and key issues in image steganography: A survey". Computer science review **13**: 95-113.

Satir, E. and H. Isik. 2012. "A compression-based text steganography method." Journal of Systems and Software **85**(10): 2385-2394.

Satir, E. and H. Isik. 2014. "A Huffman compression based text steganography method". Multimedia Tools and Applications **70**(3): 2085-2110.

Yu, J., et al. 2008. "A fragile document watermarking technique based on wet paper code". Intelligent Information Hiding and Multimedia Signal Processing, 2008. IIHMSP'08 International Conference on, IEEE.

Zaker, N. and A. Hamzeh. 2012. "A novel steganalysis for TPVD steganographic method based on differences of pixel difference histogram". Multimedia Tools and Applications **58**(1): 147-166.