Security Secret Information Hiding Based on Hash Function and Invisible ASCII Characters Replacement

Feng Liu[1], Pengpeng Luo[1], Zhujuan Ma[2], Cheng Zhang[1], Yiwen Zhang[1], Erzhou Zhu[1,*]
[1]School of Computer Science and Technology, Anhui University, Hefei, China
[2]School of Economic and Technical, Anhui Agricultural University, Hefei, China
[*]corresponding Author (ezzhu@ahu.edu.cn)

***Abstract*—The widely and frequently usage of the text document in the communication make the text-based information hiding technology still an important topic in computer security. Currently, the popularly used techniques in this area are facing many problems, such as poor robustness, lower embedding rate and semantic clutter. As a result, the secret information can be easily detected or even extracted by the interceptors. In order to cope with these problems, this paper proposes a novel secret information hiding algorithm based on the integration of the hash function and the invisible ASCII character replacement technology. Firstly, the binary secret information is encoded with even number of "1" in each group. Secondly, the space characters in each carrier segment are replaced with "SOH" by the replacement algorithm. Thirdly, the replaced segment is processed by the hash function. Finally, the algorithm is completed by comparing the generated hash values with the encoded secret information. Furthermore, by utilizing the hash collisions of the previous segment, the algorithm is improved to enhance the security and the embedding rate. The experimental results demonstrated that the proposed algorithm is feasible, effective and reliable.**

Keywords*—*Text-based information hiding; Invisible character replacement; Hash function; Information security

I. INTRODUCTION

With the rapid development of the networking technology, Internet becomes the main and efficient channel to communicate for commons. However, there is a big challenge when the end users transmit their private information via public network, since this information is likely to be intercepted by the eavesdropper. So, the safety transmission of private information on the internet attracts much attention from both industry and research communities.

In order to realize secure communication, many encryption techniques have been proposed to protect the private data, such as MACs [1], SHA-3[2], AES [3], RSA [4] and so on. But the data, which is readable by human, would become a pile of messy code after encrypt process. As it may imply the existence of secret information, interceptors are likely to pay attentions on such frequently appeared encrypted code. In order to cope with this shortage, the replacement technology, called information hiding [5], is proposed to ensure the security information transmission. In general, there are two main kinds of the information hiding technology, the digital watermarking [6] and the steganography [7]. In this paper, we focus only on steganography.

Steganography is the practice of concealing a file, message, image, or video within another file, message, image, or video. In this technology, many kinds of messages, like video, audio, email, digital image and text can be used as carriers for secret information transmission. According to the carriers, the steganogrephy technology can be classified into two categories: the multimedia-based (such as video and digital image) steganogrephy and the text-based (such as .txt, .doc and .pdf files) steganogrephy. Meriting from the relatively lager redundant space in multimedia carriers, it is relatively easier for the former one to embed secret information. Furthermore, the multimedia-based steganogrephy not only provides high rate of secret information embedding but also makes the embedded secret information hard to detect. On the contrary, due to smaller redundant space for secret information, relatively lower embedding rate and easier to figure out the change on the carrier documents, little work has been performed on the text-based steganogrephy technology. However, due to the widely and frequently usage of text document in the communication, the text-based steganogrephy technology has still attracted much attentions from the information hiding research communities. So the way utilizing of text as the carrier to embed the secret information is still has it merits.

According to the secret information embedding styles, the text-based information hiding mechanisms can be divided into two types: the format-based information hiding mechanism and the content-based information hiding mechanism. Specifically, the format-based information hiding mechanism can embed the secret information to the carrier document by adjusting its font, line space, word space, words count in one line, adding blank characters and so on. However, this method is relatively poor robustness and been rarely used in practice since the change of the format of the carrier document will directly lead to the disappearance of the secret information.

The content-based information hiding mechanism, also called natural language based information hiding, is realized by processing the syntax (such as TEXTO algorithm, NICETEXT algorithm and synonymous replacement algorithm [8]), semantic (such as machine translation based algorithm [9]) and statistical properties (such as Markov Chain based algorithm [10]) of natural languages. Since the underlying natural language processing technology is far from mature by now, there are still some obstacles to be resolved in this technology. On one hand, by the present language processing algorithms, there is obvious distinction between the generated carrier document and the original nature language. This difference can be easily distinguished by unaided eyes. So, the documents that generated

2324-9013/16 $31.00 © 2016 IEEE
DOI 10.1109/TrustCom.2016.298

by the present language processing algorithms cannot meet the practical application requirements from the point views of syntax, semantic and statistical properties respectively. On the other hand, the complex features of natural language make it is a difficult work for constructing a reasonable and effective substitution table for the replacement-based hiding algorithm. Even though an ideal substitution table is available, this algorithm is still not security enough, since the substitution table can be hacked by interceptors with little effort. In addition, most of the replacement-based hiding algorithms are accomplished by a simple way, i.e. "0" represents replacement and "1" represents unchanged or reverse. It is hard for this simple mode to resist attacks from detecting algorithms based on statistic analysis.

In order to cope with issues existing in the current context-based information hiding algorithms, this paper proposes a new secret information hiding algorithm by integrating the hash function and the invisible characters replacement technology. In this algorithm, the secret binary information is firstly encoded for parity which used to distinguish secret-carrying sections with non-secret-carrying ones and meanwhile enhanced the security of algorithm. Then space characters in the English text segmentation are replaced with an invisible character called "SOH", one at a time by corresponding replacement algorithm. After that hash function is employed to compare the hashed result and secret information to complete the embedding procedure. The receiver just needs a reverse process to extract secret information.

The remainder of this paper is organized as follows. Section II briefly analyzes the invisible ASCII characters. Section III discuses the implementation of the proposed algorithm. Section IV gives its improvement. Section V evaluates the experimental results. Finally, Section VI briefly concludes the paper and outlines our future work.

II. ANALYSIS OF ASCII CHARACTERS

ASCII is a computer coding system based on the Latin alphabet, mainly for the display of modern English and other western European languages. It is now the most common single-byte encoding system, equivalent to the international standard ISO/IEC 646.

At present, information hiding algorithms based on the substitution of invisible characters are mainly focusing on adding spaces or line breaks to some specific locations [11] or replacing blank space by null character (code as 0000000) according to secret information. For example, the famous information hiding system WbStego in the market is built based on these methods. But these methods are rarely used because of the poor robustness and relatively lower embedding rate.

After performing many different tests, we found that SOH (start of head, coded as 0000001) and *SP* (space character, coded as 0100000) have similar effects in most of documents. Simultaneously, there are a lot of candidate *SPs* in English document for replacing. So, in this paper, we use *SP* and SOH substitution method in English text to implement our information hiding algorithm.

III. IMPLEMENTATION

Fig. 1 outlines the overall secret information transmission workflow by applying our hiding algorithm. Generally, the workflow can be divided into 3 stages, the information hiding stage, the information transporting stage and the secret information extracting stage.

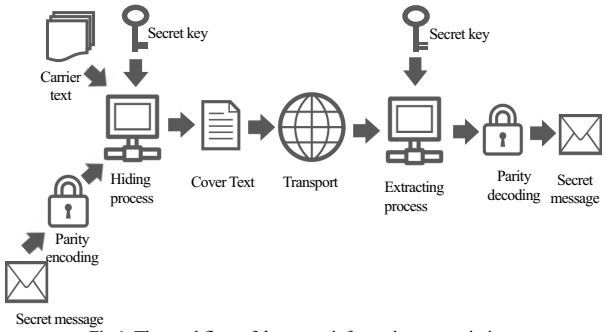

Fig 1. The workflow of the secret information transmission.

A. Sharing Setting

Before the secret message has been transmitted, some shared information, such as the division of carrier document and the division of the secret information, between the sender and receiver should be set. Furthermore, the sender and receiver also need to reach an agreement on the form of hash function. Actually, some famous hash functions, such as MD4, MD5, and SHA-1, and even user defined hash functions are all feasible. In this paper, MD5 is selected as our hash function.

1) Carrier document division

In the secret information embedding procedure, one of the crucial work, called segmentation, has to be performed to break down the carrier document into smaller segments. On available of each smaller segment, the information hidden process is performed. However, in order to properly process the embedded secret information, the two sides have to reach an agreement on the division of the carrier document.

Actually, the simplest implementation of this division is to set a full English sentence as one segment. This method looks feasible and reasonable and it is used in the MT-based methods. However, different division means different number of spaces in a single segment and different number of hashed values that can be compared with the present binary information. It is obviously that more spaces in a segment means a high probability of embedding success. Meanwhile, the way of division of the original carrier document directly influences the embedding rate (*ER*) as Equation (1) defines.

$$\mathrm{ER} = \frac{\text{Number of bits of secret information}}{\text{Number of bits of effective carrier document}} \quad (1)$$

In order to reach high secret information embedding rate, it is necessary for our algorithm to choose a reasonable division of the carrier document.

2) Secret information division

Before being embedded into the segments of the carrier document, the secret information has to be split into groups with h bits and a process called parity coding need to be performed. By parity coding, the *h* bits of secret information will generate an *h*+1 bits parity code. Consequently, there are even "1" in the *h*+1 bits of parity code. As a result, there are *h*+1 bits of secret information needed to embed into one segment of the carrier document. During the process of the secret information transmission, the sender as well as the receiver needs to be in agreement on the size of *h*.

Actually, the value of *h* is generally small but it has a significant impact on the *ER*. On one hand, smaller *h* will result in low *ER* even if the number of hash values for a given segment is

high. On the other hand, larger h will result in frequently occurs of errors (i.e., frequently fails to find a proper hash value same as the given secret bits). Supposes there are k *SP*s in a given segment of the carrier document, the failure probability (*PR*) with a given number h can be calculated as the formula below:

$$\mathrm{PR} = \left(1 - \left(1/2\right)^{h+1}\right)^{k} \tag{2}$$

In order to acquire high *ER*, relationship between the value of h and k needs to be carefully considered. By analyzing the experimental results (as described in Section), we set $h \in [1, 7]$ and $k \in [2^3, 2^6]$ in our algorithm.

However, it is also possible that there is no hash values for a given carrier segment and a group of secret information, even if we get the reasonable value of h and k. The occurrence of this situation is defined as *PNH* (possibility of no hash value) as (3) describes.

$$\mathrm{PNH} = \begin{cases} \left(1/2\right)^{k}, & h+1 \text{ is odd} \\ \left(1 - \left(1/2\right)^{h+1} - 1/2\right)^{k}, & h+1 \text{ is even} \end{cases} \tag{3}$$

From Equation (3), we can see that the *PNH* is relatively small. In our improved method (in Section IV), this value can be make so smaller to be negligible.

3) Shared secret key

The sender and receiver have to set a shared secret key used for hash process. As the secret key and the secret information are transmitted as the same way, the interceptors do not aware the existence of the secret shared key. So they cannot detect the existence or even extract the secret message by the hash process. If the secret keyed alone is considered to be not strong enough, the hidden message itself can additionally be encrypted with a secret key prior to the steganographic encoding process.

B. Embedding procedure

For definiteness and without loss of generality, it is assume that the secret information is in the form of binary stream. On available of this kind of secret information, the following four steps are employed to embed the secret information into the carrier document.

1) Secret information encoding

Firstly, it is supposes that the length of the secret information is L bits and it been split into groups with h bits. Then, a parity bit is added to ensure that there is even number of "1" in each group. As a result, there are h+1 bits in each group for the secret information embedding procedure. However, if L is not an integer multiple of h, the last group will be automatic completed, as processed by Fig. 2.

Actually, the additional bits will be neglected during the secret information extraction procedure. So, it is not necessary for the sender and receiver to have an agreement on the added bits. They just need to ensure that there is even number "1" in the last group.

2) Information substitution

Firstly, it has to choose an English text document as the carrier document. Then, the segments in carrier document are fetched in turn. Each segment is processed by the substitution function to replace the *SP* with the SOH. For security reason, only one *SP* is selected from a segment for substitution. Although, replaces more than one spaces will bring more options. However, more options will results in more changes on the original document. Consequently, more changes will attract more attentions from the interceptors.

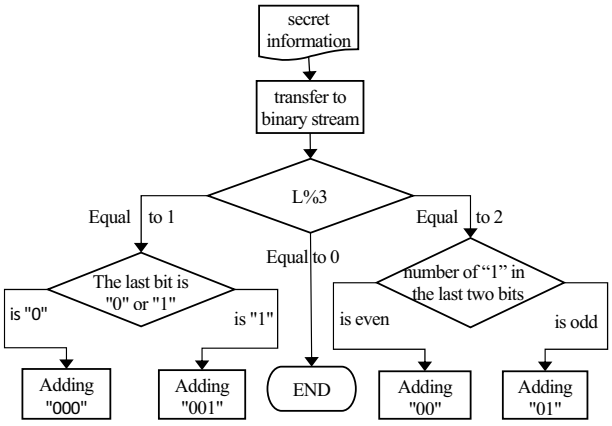

Fig 2. (An example of encoding auto-complete with h=3).

3) Replaced segment hashing

Each replaced segment is firstly hashed with a secret hash key. Then, the hash values will be compared with the encoded h+1 bits group of the secret information. During this process, if there is more than one hash values equal to the secret message, we can randomly select one. However, if there is no result same as the message, results that have odd number of "1" can be randomly selected. It has to notice that the selected odd "1" result has no secret information.

Repeat the above steps until all the secret information is successfully embedded into the carrier document. Latterly, copy the rest part of the original carrier document into the newly generated document to complete to embedding procedure.

C. Extraction procedure

Compared with the embedding procedure, the extracting procedure is relatively simple as long as the receiver has the same configuration with the sender. The workflow of the secret information extraction procedure can be described as: (1) orderly applies the same hash function and the same secret hash key on each segment of the received English document. (2) If the first h+1 bits of a derived hash value has even number of "1", then the first h bits of this value are the secret message. (3) Otherwise, if the derived hash value has odd number of "1", then this hash value can be neglected since it has no secret information. (4) Repeat reading the segment from the received document until a segment has no "SOH" character is met.

As described in Fig. 2, there might be some additional bits in the last segment of the document. Since these bits do not affect the secret information transmission, they don't need any special operations.

IV. ALGORITHM OPTIMIZATION

In our algorithm, the hash function is employed to make the information hiding method stronger to be detected. However, the phenomenon of collisions, namely different initial values may have the same hash result, is inevitable.

In our method, collision happens when the hash values derive from a single English segment happen to the same results in their highest h+1 bits. Specifically, for an original segment A, $X1$ is a segment generated by replacing the first *SP* of A; $X2$ is another segment generated by replacing the second *SP* of A. *Hash1* is the hash value comes from $X1$, and *hash2* comes from $X2$ processed by the same hash function as $X1$. Under this situation, if the

highest h+1 bits of *hash1* and *hash2* are identical, we can say that the collision is happened, and the *X1* and *X2* are called collision replacements. The existence of collision means less choice of candidate hash values.

According to the "birthday paradox" theory, when there are k *SPs* in a segment, the probability of not equal happens in the highest h+1 bits of any two hash values that are derived from this segment can be calculated as $\Pi_{i=0}^{k-1}(1\text{-}i \times 2^{-(h+1)})$, where $k < 2(h+1)$. However, if $k > 2(h+1)$, this probability is 0. In most cases, this probability (not equal) is pretty small even if we get the reasonable h and k. So, the collision frequently happens during the hashing process been described in Section .

In this section, we will propose an improved method which can settle the collisions effectively. In this improvement, the existence of hash collisions of a former segment can be used to help reducing the occurrence of collisions of its subsequent segments by providing them with a richer set of available hash values.

In order to better illustrate the improved method, we define that there are two cases of collision replacements, the even collision and the odd collision. The even collision happens when several replaced segment derived from a single original segment hash to values with the same part of their highest h+1 bits. For all the hash values of the replaced segments that are derived from a carrier segment, if they are different from the secret information needs to be hidden, then the hash values with odd number of "1" are called odd collision. The even collisions and odd collisions consist of all collisions of a carrier document. In our improved method, collisions of the former segment can be used to generate more hash values to its subsequent segments.

For the purpose of effectively utilize the existed collisions, a window W with C consecutive English segment is set. In this window, the numbers of the hash values come from the i^{th} segment are influenced by the collision replacements of all the segments prior to i. In the hash procedure, the present replaced segment will be hashed accompany with one of the former collisions. Namely, if p and q represent the present replaced segment and one of the former collision replacement respectively, then *hash(p+q)* is the hash value of p.

If the $(i\text{-}1)^{th}$ segment has N collision replacements, the number of available hash values of i^{th} segment will hike up N times. The growing number of hash values directly led to the probability of the existence of hash values same as the given secret bits for the i^{th} segment. Specifically, the amount of hash value for segment i (A) can be calculated as:

$$A = K_i \times (\Pi_{j=1}^{i-1} N_j) \quad (4)$$

In (4), K_i represents the amount of *SPs* in the $i^{th}(0<i<C)$ segment and N_j represents the amount of the collision replacements of the $j^{th}(0<j<i)$ segment. From the equation, we can see that A grows with C exponentially. A relatively big value is assigned to C, will make A an unacceptable huge number, so it is needed to set an upper bound t ($A \leq t$) for A. The reasonable value of t is affect by the hardware environment as well as the value of h and k. Here, h and k refer to the number of bits in a secret information group and number of *SPs* in a segment of carrier document respectively.

In our implementation, a window that accommodates two consecutive segments is set, namely $W=2$. From the above analysis, a larger W could bring more available hash values. But larger W also brings more cost to compute and more space to buffer hash values. Furthermore, the threshold of the number of hash values for a specific segment is set as t. The detailed embedding procedure of our improved method is described as follows:

Step 1. Encoding the initial secret information to ensure that there is even number of "1" in each group (same as the first step of Section Ⅲ.B).

Step 2. For each window W, utilize ASCII character "SOH" to replace the i^{th} ($0 < i \leq k$) *SP* of the first segment to get a k elements set X ($X=\{X_i \mid X_i$ is the replaced segment $\wedge\ 0<i\leq k\}$); Then, apply hash function on these replaced segments to get k hash values; At last, compare all these hash values with the given secret bits to get a collision set P ($P = \{P_i \mid P_i$ is an even or odd collision $\wedge\ 0<i\leq k<t\}$).

Step 3. Orderly select $P_i \in P$, $Y_j \notin Y$ ($Y=\{Y_j \mid Y_i$ is a replaced segment generated from the 2nd segment *of* $W \wedge\ 0 < j \leq k\}$), and utilize the hash function $hash(P_i+Y_j)$ to get a new group of hash values. Here, the number of new generated hash values is k times with the number of elements of P. Then, compare this group of hash values with the given secret bits to get a replaced collision set Q ($Q =\{Q_i \mid Q_i$ is an even or odd collision $\wedge\ 0<i\leq k\}$) of the second segment of W. Q is a subset of $Y_i \times P_i$. By now, the embedding procedure of the first window is finished.

Step 4. By the previous steps, two replaced collision sets, P and Q, may be generated for embedding the secret information. If P is generated, since it must contain some even collisions, we can randomly select an even collision as the carrier segment. Otherwise, only Q is generated, (1) if there is an even collision q in Q and q is derived from an even collision p in P (which means the result of *hash(p+q)* is equal to the given secret bits), we can embed $2h$ secret bits (not including the parity bit) in the present windows W. However, if q is derived from an odd collision of P, we can only embed h secret bits; (2) if all elements in Q are odd collisions, corresponding to the two cases of (1), only h bits and 0 bits are able to be embedded respectively. By considering all possible situations, proper segments of the carrier document are selected to embed the maximum number of secret bits.

Step 5. If the all the secret bits are embedded, terminate the embedding procedure, otherwise, continue processing the next window, turn to Step 2.

V. EXPERIMENTAL EVALUATION

The experiments in this section are carried out on the machine with Intel i7 4790 CPU (3.6 GHz), 8GB DDR3 1600 RAM and 64-bits Windows 7 OS. The carrier documents are .doc and .pdf files written in English.

A. Theoretical maximum embedding rate (MER)

MER (maximum embedding rate) is firstly mentioned in the machine translation based (MT-based) information hiding method [12]. In MT-based method, *MER* means the ratio of the theoretical maximum secret bits that could be embedded in one English sentence to the average length of an English sentence. The ratio is influenced by the quantity of the translations per sentence and more translations mean larger *MER*. The MT-based method uses English sentence as the basic embedding unit. However, diverse lengths of different sentences and different number of translations of a specific sentence directly lead to poor *MER*. In order to

facilitate processing, they statistically set 1168 bits as the average length of one sentence [13]. Some *MER* of MT-based method is shown in table 2 (*k'* represents the average number of translations per sentence; *AL* represents the average length of English sentence).

TABLE 1. *MER* OF MT-BASED METHOD.

k'	11.62	15.15	18.01
AL	1168	1168	1168
MER(%)	0.22	0.27	0.31

TABLE 2. *MER* OF OUR METHOD.

k	k=2	k=4	k=8	k=16	k=32	k=64
AL	11.6	23.6	47.4	95.0	190.6	381.2
MEB(%)	1.14	1.09	0.80	0.53	0.33	0.20

In this paper, we set the English segments correspond to the English sentences of the MT-based method, the number of *SPs* in a segment correspond to the number of translations per sentence of the MT-based method. There are two differences between our method and the MT-based method: (1) the length of our segment can be freely chosen and this length is relatively fixed by setting the same *SPs* in each segment, however, the length of sentence in the MT-based method is fixed; (2) the number of *SPs* in our segment is fixed and this number is under our control, however, the number of translations per sentence is undermined in the MT-based method.

According to the definition of *ER* described in Section Ⅲ.A, the *MER* of our method can be defined as: $MER = (\log k)/L_k$. In this definition, L_k represents the length (we use bit as unit) of the segment with k spaces. Table 3 shows the *MBR* under different value of k (*AL* represents the average length of our segment under different value of k).

In table 2, *MER* is increasing with the growth of *k'*. However, our results shown in Table 3 are the opposite: *MER* is decreasing with the growth of k. The value of *k'* in Table 2 is affected by underlying translation machine. In order to ensure the translation quality, too large *k'* is not acceptable. As a result, the MT-based method has a relatively small *MER* (the maximum *MER* is about 0.33% in practice). Since the value of k is changeable, our method can reach an ideal *ER* when a proper k is chosen. In Table 3, *MER* reaches large value when k is set as 4 or 8. However, too large *MER* may lead to the failure of the information embedding procedure. As a compromise, we choose 8 as the proper value of k.

B. The effect of h and k on ER

The above section analyzed the *MER* of our method theoretically. But in reality, the embedding error and the redundant bits introduced by parity code make the procedure cannot reach such high *ER*. As a matter of fact, we have a much lower *ER* in our original method. However, we are able to approach the *MER* when the reasonable values of h and k are chosen as our improved

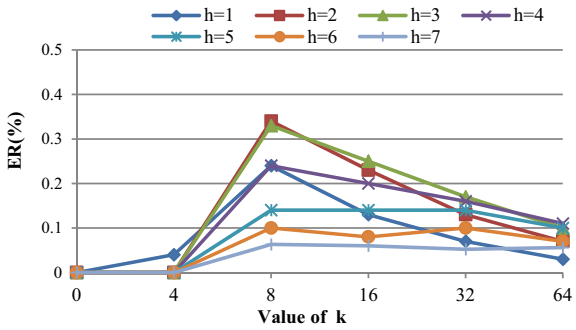

Fig. 3. The effect of h and k on *ER* of our original method.

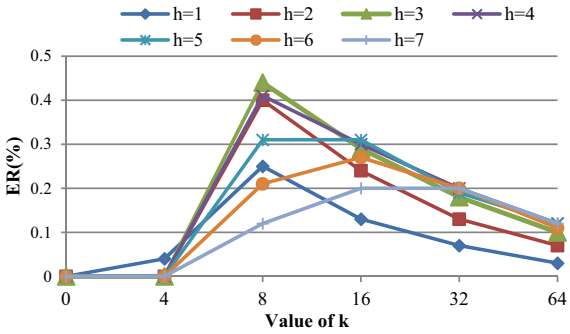

Fig. 4. The effect of h and k on ER of our improved method.

method does. In our methods, k represents the number of *SPs* in a carrier segment; h records the number of bits in a secret information group. Fig. 3 and Fig. 4 depict the effect of h and k on *ER* before and after the improvement. Since it is often not operable for the value of x-axis less than 4, the value of y-axis is basically "0" when the value of x-axis less than 4.

The figures also show that *ER* reaches a relatively large value when k is set as 8 or 16; h is set as 2, 3 and 4. Obviously, *ER* has a considerable increment after the improvement. In the improved method, the largest two values of *ER*, 0.44% and 0.41%, are achieved by setting k as 8 and 16 respectively. It is improved more than 30% compared with the original method.

However, there is still a big gap between the largest *ER* in Fig. 4 and the *MER*. Fig. 3 and Fig. 4 also show that *ER* is not regularly increasing or decreasing with the increment of h and k. As a matter of fact, the increment of k and h will result in larger capacity of the carrier segments; larger capacity of each carrier document will result in longer carrier segment; longer segment will result in the higher probability of embedding error since it will bring a lot of carrier segments with no embedded secret information.

C. The effect of h and k on embedding error

In the former subsection, we mentioned that *ER* is not regularly increasing or decreasing with the increment of h and k. In order to get the best *ER*, we need to evaluate the influences of h and k on the occurrence of embedding error. Fig. 5 and Fig. 6 give the results of embedding error with different h and k before and after the improvement.

In Fig. 5, *A1* and *A2* represent the number of carrier segments with no embedded secret information and the total number of segments in carrier document before improvement respectively. *B1* and *B2* represent the same results after the improvement. In

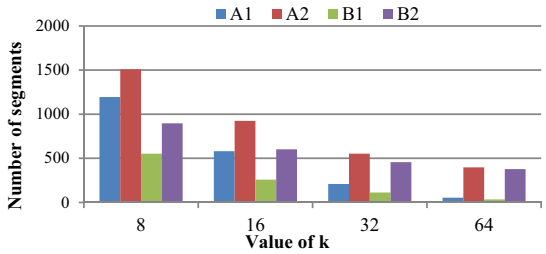

Fig. 5. "Empty" with different k(h=4).

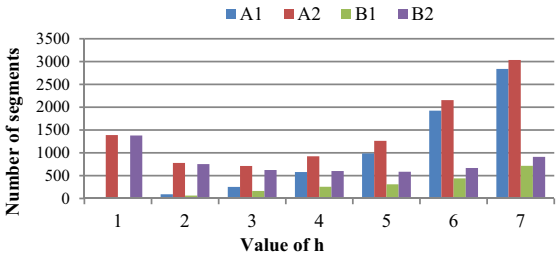

Fig. 6. "Empty" with different h (k=24).

this experiment, the length of secret information is 1376 bits (1720 bits after parity encoding). The secret information are divided into groups with 4 bits (i.e. *h=4*). From the figure, we can see that all the results, *A1*, *A2*, *B1* and *B2,* are decreasing with the growing number of *k*. *B1* and *B2* decrease faster than *A1* and *A2*.

In Fig. 6, *A1*, *A2*, *B1* and *B2* represent the same values as the ones in Fig. 5. In this experiment, the number of *SPs* in each carrier segment is set as 16 (i.e. *k=16*). From the figure, we can see that *A2* and *B2* are decreasing with the growth of *h* until it arrives at 3. This is because smaller *h* will get relatively lower rate of embedding error. As shown in Fig. 5, *A1* and *B1* are relatively small compared with *A2* and *B2*, so *A2* and *B2* are almost only affected by the capacity of the carrier segments. However, *A1* and *B1* are increasing rapidly with the growth of *h* when it exceeds 3. Under this circumstance, *A1* and *B1* take bigger and bigger proportions of *A2* and *B2* respectively. As a result, *A2* and *B2* also increase rapidly with *h.* Fig. 6 also shows that *B1* and *B2* are much smaller than *A1* and *A2* respectively. Which means our improved method gets better *ER* than the original method.

The results in the two figures also show that we get better *ER* than the original method. However, there is still a gap between our best *ER* and theoretically *MER*. By performing further experiments, we find that the first segment in the carrier windows is the biggest factor for the occurrence of the embedding error. As a matter of fact, our experiments only place two segments to each carrier windows (i.e. *w=2*) for simplicity. The improved method only increased the hash values to the second segment of the carrier window. But the first segment is processed as the original method. For this reason, we can place more segments to the carrier windows to get high *ER*.

D. Security analysis

There are two criteria for evaluating the security of the information hiding algorithms: the ability of covering the "noises (i.e. changes)" that the hiding algorithms bring into the original carrier documents cannot be easily detected by the interceptors and the ability of preventing the secret message being extracted by the interceptors when they are aware of the existence of this message.

1) Ability of Hiding

As mentioned in Section II, by performing many different tests, we found that the SOH has the similar effect in most of documents. Meanwhile, there are many candidate *SPs* in the document for replacing, our framework uses *SP* and SOH substitution method to implement the information hiding algorithm. However, similar do not means the same. In the document, the width of *SP* is shorter than the ones of SOH. This means there are many abnormal gaps with different widths among words of the document that carrying secret messages. This kind of "noise" may not be able to resist the test of detection algorithm based on word shift (e.g., algorithms in [14]). In order to cope with these detection algorithms, we introduce the word shift based hiding method to our algorithm to make SOH has the absolutely same width with *SP*.

Another kind of "noise" our framework brings into the original carrier document may be the probability of *SP* (denoted as r_1) and the probability of continuous *SP* (denoted as r_2). In the generated document that been embedded secret message, r_1 is defined as the number of all *SPs* divides by the number of all characters; r_2 is defined as the number of continuous *SPs* divides by the number of all *SPs*. Theoretically, the threshold for r1 is 0.3 and the threshold for r_2 is 0.2 [15]. Since we put the restriction on the value of *k*, the values of r_1 and r_2 in our method are much smaller than the theoretically ones. Table 4 lists the average results of our experiments by testing 100 different documents. In this table, the first line (*Original*) lists the results of the original carrier documents, while the rest lines are all the results of the generated documents that been embedded with secret message with different *h* and *k*.

TABLE 3. PROBABILITY OF SP AND PROBABILITY OF CONTINUOUS SP.

	r_1	r_2
Original	0.1893	0.0138
h=2, k=8	0.1630	0.0086
h=2, k=16	0.1767	0.0106
h=3, k=8	0.1628	0.0084
h=3, k=16	0.1464	0.0090
h=4, k=8	0.1643	0.0076
h=4,k=16	0.1764	0.0089

From the Table 4, we can see that results of our method are much better than the threshold ones in [15]. The generated documents of our method are totally met the demands of the normal document from perspectives of semantics, syntaxes and statistic characteristics respectively. As a result, detection algorithms, such as natural language processing based method, different form synonym replacement based method and MT-based method, cannot work well on our algorithm.

2) Ability of resisting attack

Unfortunately, if a smart interceptor happened to smell the existence of the secret message, he would find it is a significantly hard procedure to extract the secret message without known about the embedding algorithm and the values of relative parameters of our method. At present, there is no effective algorithm in the market has the ability of extracting the secret message from all the information hiding systems. Beside the values of *h*, *k*, *w*, *t* (the maximum number of hash values in a carrier document) and the location of the bits in hash values that used to compare with the given secret bits, we also employ a secret key in hash function. Brute force the hash function, like MD5, SHA-1, SHA-2, is proved harder. In addition, we can also construct a unique hash function and encrypt the secret messages beforehand to enhance the security of our system.

E. *Information hiding effect*

In our method, the inputs of the embedding procedure are the carrier document and the secret message. The output is the "normal" document that that been embedded with the secret message. Fig. 7 and Fig. 8 display the hiding effect in .doc document by a piece of the original carrier document and the corresponding document that been embedded with secret information ("Brussels followed Washington's lead in 1989 by a company of seven."). Fig. 9 and Fig. 10 show the similar effect in .pdf document. The document in Fig. 8 and Fig. 10 are further processed by the word shift algorithm in [14]. From the two figures we can see that the differences that brought by the embedding procedure brings can be neglected except for the format distinctions that built-in different types of files.

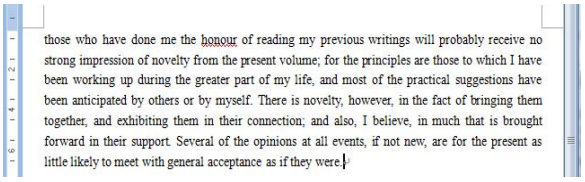
those who have done me the honour of reading my previous writings will probably receive no strong impression of novelty from the present volume; for the principles are those to which I have been working up during the greater part of my life, and most of the practical suggestions have been anticipated by others or by myself. There is novelty, however, in the fact of bringing them together, and exhibiting them in their connection; and also, I believe, in much that is brought forward in their support. Several of the opinions at all events, if not new, are for the present as little likely to meet with general acceptance as if they were.

Fig.7.Original carrier document.(.doc)

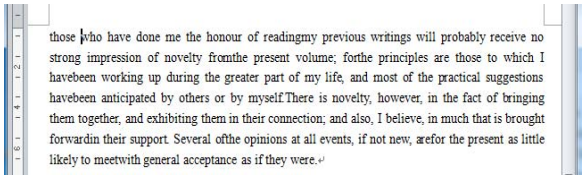
those who have done me the honour of readingmy previous writings will probably receive no strong impression of novelty fromthe present volume; forthe principles are those to which I havebeen working up during the greater part of my life, and most of the practical suggestions havebeen anticipated by others or by myself.There is novelty, however, in the fact of bringing them together, and exhibiting them in their connection; and also, I believe, in much that is brought forwardin their support. Several ofthe opinions at all events, if not new, arefor the present as little likely to meetwith general acceptance as if they were.

Fig. 8. Document embedded with secret message.(.doc)

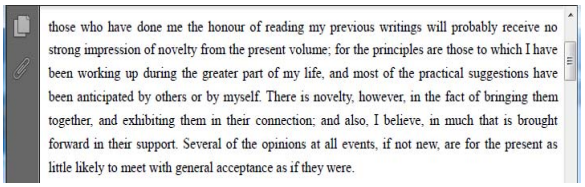
those who have done me the honour of reading my previous writings will probably receive no strong impression of novelty from the present volume; for the principles are those to which I have been working up during the greater part of my life, and most of the practical suggestions have been anticipated by others or by myself. There is novelty, however, in the fact of bringing them together, and exhibiting them in their connection; and also, I believe, in much that is brought forward in their support. Several of the opinions at all events, if not new, are for the present as little likely to meet with general acceptance as if they were.

Fig.9.Original carrier document.(.pdf)

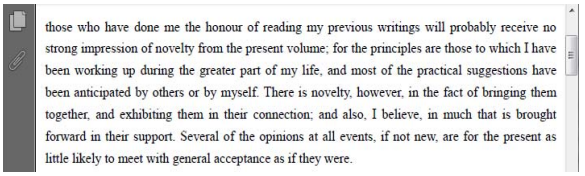
those who have done me the honour of reading my previous writings will probably receive no strong impression of novelty from the present volume; for the principles are those to which I have been working up during the greater part of my life, and most of the practical suggestions have been anticipated by others or by myself. There is novelty, however, in the fact of bringing them together, and exhibiting them in their connection; and also, I believe, in much that is brought forward in their support. Several of the opinions at all events, if not new, are for the present as little likely to meet with general acceptance as if they were.

Fig. 10. Document embedded with secret message.(.pdf)

V. CONCLUSION AND FUTURE WORK

Based on the analysis of the text-based information hiding technology state-of-art, this paper proposed a novel secret information hiding algorithm based on the integration of hash function and the invisible ASCII character replacement technology. By introducing a much more complex replace pattern, the generated document can easily resist the brute force secret information detecting methods. This algorithm needn't to construct a substitution table, which eliminates the safety risk brought by the table. By introducing the word shift based hiding method and setting the restriction on the number of space symbols in each carrier segment, the proposed algorithm brought much less "noise" to the carrier document and the generated document presented the same effect with the original ones. As a result, the interceptor cannot easily detect the existence of the secret information. By utilizing the collision replacements for the subsequent carrier segments, more available hash values are generated and then higher ER, embedding success and security are all achieved. The experimental results also show that the proposed algorithm is feasible, effective and reliable. However, only applicable for the English documents that have plenty of space symbols seriously restricts the applications of the proposed algorithm. So, in the future, it is urgently needed to apply this algorithm to a wider range of languages.

Acknowledgment

This work is partly supported by the academic and technical leader recruiting policy of Anhui University, the National Natural Science Foundation of China (Grant Nos.61300169), the Humanities and Social Sciences Project from Chinese Ministry of Education (Grant Nos.15YJAZH112), the Natural Science Foundation of Anhui Province (Grant Nos.1408085QF108), and the Natural Science Foundation of Education Department of Anhui province. (Grant Nos.KJ2016A257).

Reference

[1] Hugo Krawczyk, Mihir Bellare, Ran Canetti. HMAC: Keyed-Hashing for Message Authentication. RFC 2104, February 1997.

[2] SHA-3Standard: Permutation-Based Hash and Extendable-Output Functions, FIPS PUB 202. National Institute of Standards and Technology (NIST). August 2015.

[3] AES-The official Advanced Encryption Standard, FIPS PUB 197. Computer Security Resource Center. National Institute of Standards and Technology (NIST). Retrieved 26 March 2015.

[4] Ronald L.Rivest, Adi Shamir, Leonard M.Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. Communications of the ACM (Association for Computing Machinery) 26(1), 1983, pp. 96–99.

[5] Pierre Moulin, Joseph A. O'Sullivan. Information-theoretic analysis of information hiding. IEEE Transactions on Information Theory, 49(3), 2003 pp563-593.

[6] Mehran Andalibi, Damon M.Chandler. Digital Image Watermarking via Adaptive Logo Texturization. IEEE Transactions on Image Processing, 24(12), 2015, pp.1-15.

[7] Niels Provos, Peter Honeyman. Hide and seek: an introduction to steganography. IEEE Security & Privacy, 1(3), 2003, pp.32–44.

[8] Gongshen Liu, Xiaoyun Ding, Bo Su, Meng Kui. A Text Information Hiding Algorithm Based on Alternatives. Journal of Software, 8(8), 2013, pp.2072-2079.

[9] Christian Grothoff, Krista Grothoff, Ryan Stutsman, Ludmila Alkhutova, Mikhail J. Atallah Translation-based steganography. Journal of Computer Security, 17(3), 2009, pp.269-303.

[10] Shufeng Wu. A study of information hiding technology. M.S. dissertation, University of Science and Technology (China), 2003.

[11] AA Mohamed. An improved algorithm for information hiding based on features of Arabic text: A Unicode approach.Egyptian Informatics Journa,15(2), 2014, pp.79-87

[12] Peng Meng. Research on Linguistic Steganography and Steganalysis. Ph. D. dissertation, University of Science and Technology (China), 2012.

[13] Ryan Stutsman, Christian Grothoff, Mikhail Atallah. Lost in just the translation. In: Proceedings of the 2006 ACM Symposium on Applied Computing (SAC '06), Dijon, France, April 2006, pp 338-345.

[14] Chao Chen, Shuozhong Wang, and Xinpeng Zhang. Information hiding in text using typesetting tools with stego-encoding. In: Proceedings of the First International Conference on Innovative Computing, Information and Control (ICICIC '06), Washington, DC, USA, 2006, pp. 459–462.

[15] Xin-guang Sui, Hui Luo. A Steganalysis Method Based on the Distribution of Space Characters. In: Proceedings of 4th International Conference on Communications, Circuits and Systems. Guilin, China, 2006, pp 54-56.