Hindawi Publishing Corporation
Applied Computational Intelligence and Soft Computing
Volume 2015, Article ID 387985, 13 pages
http://dx.doi.org/10.1155/2015/387985

Research Article

Hiding Information in Reversible English Transforms for a Blind Receiver

Salma Banawan and Ibrahim Kamel

Department of Electrical and Computer Engineering, University of Sharjah, University City, Sharjah, UAE

Correspondence should be addressed to Ibrahim Kamel; kamel@sharjah.ac.ae

Received 29 October 2014; Revised 6 April 2015; Accepted 17 April 2015

Academic Editor: Ying-Tung Hsiao

This paper proposes a new technique for hiding secret messages in ordinary English text. The proposed technique exploits the redundancies existing in some English language constructs. Redundancies result from the flexibility in maneuvering certain statement constituents without altering the statement meaning or correctness. For example, one can say "she went to sleep, because she was tired" or "Because she was tired, she went to sleep." The paper provides a number of such transformations that can be applied concurrently, while keeping the overall meaning and grammar intact. The proposed data hiding technique is blind since the receiver does not keep a copy of the original uncoded text (cover). Moreover, it can hide more than three bits per statement, which is higher than that achieved in the prior work. A secret key that is a function of the various transformations used is proposed to protect the confidentiality of the hidden message. Our security analysis shows that even if the attacker knows how the transforms are employed, the secret key provides enough security to protect the confidentiality of the hidden message. Moreover, we show that the proposed transformations do not affect the inconspicuousness of the transformed statements, and thus unlikely to draw suspicion.

1. Introduction

Hiding information falls under the field of digital steganography or digital watermarking, which is the art of hiding data inside data in an inconspicuous manner. There are two main categories of digital watermarking: robust watermarking and fragile watermarking. The robust watermarking is used in the digital copyright protection, where the owner hides a watermark that can be used to proof ownership in front of the court. The fragile watermark, on the other hand, will be destroyed with the minimal change by the attacker. Fragile watermarking is used in data integrity and data hiding applications. Our proposed method falls under the fragile watermarking.

One of the most important goals of data hiding is to keep outsiders from even suspecting that hidden information may exist in the message. Textual data is the most widely used data type; therefore, it would seem to be the perfect candidate for information hiding. Hiding information in other data types, for example, images and video clips, utilized the redundancy in these data types. Hiding information inside natural language is bounded by challenging constraints arising from its well-defined structure and low redundancy. In general, the word order of a sentence in English is important, and altering it might change the meaning and/or the correctness of the grammar of the sentence. However, careful maneuvering of English sentence constituents is possible. In this paper we propose a new scheme that utilizes the redundancy in English text structure to pass secret messages.

The transitive English sentence, a sentence that contains all three parts, subject (S), verb (V), and object (O), generally follows the S-V-O order. However, O-S-V may be used to emphasize the object. For example, "Harry wants to play baseball" is S-V-O statement. It may be transformed to "Baseball is what Harry wants to play," which is O-S-V, to emphasize "baseball." In the following, swapping the order of the statement constituents, for example, changing a statement from S-V-O to O-S-V or vice versa, is called a transformation. This flexibility in constructing an English sentence is a type of redundancy that can be exploited in hiding secret messages. For example, to hide a 1-bit secret message, the sender and receiver can agree that S-V-O sentence indicates "0" while

O-S-V sentence indicates "1." Previous work on syntactic transformations, such as transforming from S-O-V to O-S-V, attained an embedding capacity of 0.5 bits per sentence [1]. In this work, we show that our transforms can embed at least 2 bits per sentence.

The goal of this paper is to utilize the maneuverability of certain statement constituents for hiding information inside an English document. The application scenario considered in this paper assumes two parties exchanging secret messages in the presence of a passive attacker. The sender and receiver, in addition to the passive attacker monitoring them, are all typical English speakers who are writing correct, and sometime casual, English expressions. The sender and receiver exchange secret messages inside English text documents (called cover). The purpose of using a cover document is to keep the message exchange inconspicuous. We consider an encoding of a cover document as "acceptable" if the grammar of the sentence remains correct and the meaning is preserved.

Secret messages can be hidden in the sentence by swapping words (called word-level transformation) or clauses (called clause-level transformation). Word-level transforms manipulate certain words in specific phrases, such as the verb phrase or object phrase of a sentence. They only affect one type of sentence constituent (either the subject or verb or object). For example, in the sentence: "Sally and Peter took a walk," if we transform the subject such that the sentence reads: "Peter and Sally took a walk," we have performed a word/phrase level transform. Transforms, at the clause-level, move entire clauses in order to embed bits. For example, the two clauses in the sentence "If you want to lose weight, you need to eat healthier" can be swapped to "You need to eat healthier, if you want to lose weight."

This paper proceeds as follows: Section 2 discusses prior works on hiding information inside text. Section 3 gives detailed description of the different transforms used by the proposed scheme in order to hide a secret message inside a cover text. A study on the frequency of the various transforms and their capacity in terms of the number of bits they can hide inside a typical cover text is detailed in Section 4. A step-by-step description of the encoding process is presented in Section 5. We provide a detailed security analysis of the secret key in Section 6. Finally, in Section 7 we draw some conclusions and suggest future directions to aid in embedding more bit using synonyms.

2. Related Work

There are different levels at which information can be hidden inside a cover document. The first method of hiding information that utilized the text medium was at the document level; text was hidden in characteristics of the document as a whole. Later on, methods for hiding text while utilizing the words of the cover document were developed. These methods are divided into two types: word/phrase substitution and linguistic transformations. Word/phrase substitutions deal with substituting words for high encodability without regarding the meaning. Linguistic transformations mainly include synonym substitution and syntactic modifications, which is found where the contribution of this paper lies.

2.1. Hiding Information at the Document Level. Traditionally, steganography dealing with text normally is comprised of using specific formatting styles and adjusting line spacing in order to hide a secret message. The work in [2] is based on interword spacing and interparagraph spacing. However, hiding information in this manner suffers a major drawback, in which simply retyping the document in a new document completely obliterates the hidden message. In addition, the processing tools available nowadays are capable of detecting such minute details yielding extraction of the hidden message to be especially simple.

Another form of hiding information at the document level employs features of Microsoft Word for producing a system that hides information inside a Microsoft Word document. In [3] text segments of a Word 2003 document are degenerated, and information is embedded in the revisions made by the cautious reviewer. The original cover, degenerated document, and the secret message are all contained in the document; allowing the recipient to decode the message based on their established dictionary. Quite recently, [4] a technique for embedding information in the records of revisions, revision identifiers (RIs), is applied made to a Word 2007 OOXML document. RIs have randomly generated unique values that are replaced by the algorithm for embedding information.

One of the latest steganography schemes that involves completely modifying an original cover document in two phases is Innocent-Cipher-Based Cryptography Paradigm (Innocipher) [5]. The work presented does provide a novel steganographic method; however, the hiding information scheme presented in this paper has the advantage of slightly and simply changing a cover document while hiding many bits of information.

2.2. Word/Phrase Replacement Schemes. NICETEXT [6] and Spammimic [7] were some of the early systems which attempt to generate encoded documents with an extremely high encoding capacity at the cost of producing remarkably suspicious text. More recently, proposals that attempt to keep function words, in order to preserve semantics, while only replacing content words, have emerged.

In [8], they develop a *LUNABEL*—a program based on replacing content-words of a sentence by their "syntactonym"—a different word that keeps the same syntactic structure. *LUNABEL* suffers a major drawback that the text encoded is not semantically related. For example, the content words in a sentence such as "The *dog* ate the *bone*" can be replaced by "The *cat* ate the *tree*." In [9] they have studied the conspicuousness of *LUNABEL* and found that they automation scheme they use is suspicious when observed by human readers as opposed to other steganography methods which are performed by humans. Our work produces much less conspicuous stego-texts as we have developed a new paradigm for ensuring inconspicuousness.

The work described in [10] provides a more sophisticated word replacement scheme that relies on a parts-of-speech (POS) tagger, for testing different scenarios to find which aspects of word replacement give better encoding capacity and which aspects produce more meaningful text. The POS

tagger tagged the n-grams to be encoded and looked for the most occurring n-grams with the same tags in order to substitute the most frequent n-grams for encoding. This paper aids in emphasizing the tradeoff between inconspicuous steganography and accurate steganography.

In [1] they describe a method for paraphrasing n-grams while attempting to maintain a high level of imperceptibility, as required by steganography. Once an n-gram is recognized, its respective paraphrase from the paraphrase dictionary is checked for in the Google corpus to ensure that the new paraphrase is widely used on the web. Also, the modified sentence is passed through the combinatory categorial grammar (CCG) parser before and after paraphrasing to check that the words surrounding the paraphrase have not been changed.

Synonym substitution is widely used for steganography, as it aids in keeping the semantics correct. In [11] a framework based on making the text as ambiguous as possible by synonym substitution is introduced. Homographs are selected from a weighted, undirected graph of (word, sense) pairs, and used for embedding information and guarding against an adversary with the same computational capabilities and knowledge of the system.

2.3. Embedding Information at the Sentence-Level. In [12] we presented some ideas with preliminary results for hiding text in the ordering of some simple English sentence parts. We showed how to hide bits in the maneuvering of phrasal verbs, Boolean operands, adverbs, and conditional clauses. In this paper, we present more encoding schemes and we give a complete picture of how the decoder can blindly decode a specific sentence.

We increase sentence encodability by exploiting the two widely found transforms even further: adverbs and conditionals. In this paper, we approach adverbs differently. Instead of only moving the adverbs that are already in cover text around the sentence, we have devised a new paradigm for selecting and inserting the most suitable adverb based on the Google Books Corpus. In this way, we ensure that every clause (which is approximately 5 words) will definitely be able to encode at least 2 bits, or as we like to describe it: "*Many adverbs a clause, make the receiver applause*." Previously, we looked at simple conditional statements and encoded them as 1 bit, while in this research we encode conditional statements as four bits. In addition, we have devised a way to encode even complex nested conditionals. In this paper, we present details of a complete sender-receiver system in which the receiver is blindly able to decode the message properly.

Linguistic transformations can be performed at the sentence-level in order to hide secret messages. The first work describing syntactic transformations for hiding information is in [13] by Atallah et al. In [13], there are three major reversible syntactic transformations: adjunct movement, clefting, and passive formation. They are applied to the syntactic tree representation in order to hide small portions of a watermark inside a text document. Later, in [14] sentences are selected based on a vocabulary that is mark-carrying: if the specific word is in the sentence, message bits may be inserted in that particular sentence. Each marked sentence is checked against a set of syntactical functions. If a sentence can handle a particular transformation, then it may be used for encoding. In [15] they apply syntactic transformations to Turkish language which is not that rigid when it comes to word order. Morphosyntactic features as well as functional dependences are used to produce the treebank representations of the sentences. A random selection based on a shared secret key chooses which type of tool will embed the watermark.

3. Manipulating Sentence Constituents

This paper studies the ability to embed a secret message in the text exchange between two typical users of English. The type of content they exchange may be a diverse range of natural language content, ranging from casual email communication and newspaper articles to technical reports. The goal is to insert a secret message, in the form of a bit stream, into a cover document, without disturbing the meaning or syntax of the original document.

For the most part, the words already in an English sentence are important and cannot be manipulated without affecting the sentence structure; however, due to the redundancy existing in the placement of certain keywords, some manipulation of the sentence constituents can give a similar meaning. For example, the sentence, "Because she was tired, she went to sleep," can be written in another form while still preserving the overall structure and meaning: "She went to sleep, because she was tired." In this example, we have switched the first clause beginning with the keyword "Because" with the second clause, in order to hide a bit of information. This redundancy can be exploited to hide secret messages. The sender and receiver agree on default encoding. For example, the "because" appears at the beginning of the sentence to indicate "0" bit value in the secret message and "1" if "because" appears in the middle.

This paper focuses on identifying specific transformation that can be manipulated for hiding secret message bits. These transformations can be classifies as word level and clause or phrase level transformations. With regard to the word level, in "everyday" English usage, some keywords appear more frequently than others. To maximize the capacity of the proposed technique, frequent keywords are chosen. For example, an adverb to describe the subject, verb, and object can be added without affecting the meaning of the sentence. If the adverb is inserted in the sentence, we embed "1" for each adverb. If the adverb is deleted we embed "0" for each of the sentence elements (subject, verb, and object). This alone ensures that any sentence with a subject, verb, and object can embed at least 3 bits and the value of these bits depends on whether or not the adverb is shown in the sentence.

Another way to hide secret bits at the word level is to exploit the abbreviation. Some keywords can be understood by convention whether or not they are abbreviated. Month of the year, such as doctor (Dr.) or event international conference (Intl.), are just few examples.

At the phrase level, we encode separable phrasal verbs as either "unseparated" to represent "0" or "separated" to represent "1." Also at the phrase level, we swap the operands around a Boolean operator such as "and." Conditional statement

constituents are swapped with regard to the clause level. In addition, conditional keywords are substituted to be able to embed more bits. There are two types of conditional statements, one that begins with the hypothesis (p →) and one that begins with the conclusion (→ q).

3.1. Splitting Separable Phrasal Verbs. Every English statement must have at least one subject and one verb. The verb can be one word or it can be a phrase (e.g., phrasal verb). When a verb (such as "*filled*") is combined with an adverb particle (such as "*up*"), the result is a phrasal verb ("*filled up*"). In English language there are *3,134 different phrasal verbs*, many of which are optionally separable [16]. If a phrasal verb has a direct object, the constituents of the verb can usually be separated. For example, the sentence "He *filled up* the gas tank" can also be written as "He *filled* the gas tank *up*." The phrasal verb parts are separated by placing the adverb particle after the object. The phrasal verbs can only be separated if the object is not a pronoun. For example, it is correct to say: "She tried the dress on" or "She tried on the dress" but we cannot say "She tried on it."

Encoding separable phrasal verbs will require a look-up table that includes all the optionally separable phrasal verbs. The program will check each sentence for the first word of the phrasal verb. If it is found, it will check whether or not the second word of the phrasal verb is in the sentence. Depending on whether the two parts of the phrasal verb are next to each other or not, the phrasal verb will be encoded as "0" or "1."

3.2. Swapping Operands of a Boolean Operator. Coordinating conjunctions are connecting words that explain a relationship between the equal-ranked words, phrases, clauses, or sentences that they are connecting. Coordinating conjunctions, such as "and," "but," "nor," and "or," are analogous to Boolean operators found in Boolean logic. We can exploit certain Boolean logic concepts to embed a secret message in a cover document. The three main Boolean operators that we will be concentrating on are AND, OR, and NOR. When the AND operator is applied to words in the English language, the components that are being added can be moved around the operator in the sentence to encode bits. The same is applicable to the operands being compared and contrasted when using the OR and NOR operators. This is due to the commutative property that these three operators share:

$$A \text{ operator } B = B \text{ operator } A. \quad (1)$$

Since the words "and" and "or" are two of the most common words in the English language [17], therefore they provide vast opportunities for hiding text. For example the two subject phrases "Sara and Alice" and "Alice and Sara" are equivalent in meaning and can be used to encode a bit at the phrase level. Another example where there are two interchangeable verb phrases is "We can read books or paint pictures," which can also be written as "We can paint pictures or read books."

In order to keep the semantics correct, certain precautions must be taken when attempting to simply swap operands. If the operands around "and" are describing events that take place in chronological order, we should not swap them in order to preserve the meaning. To illustrate, it would be incorrect to switch the two clauses in this sentence: "First, we ate lunch at home, and then we ate dinner at the restaurant." The clause discussing lunch should appear before the clause discussing dinner as lunch comes before dinner. The coordinating conjunctions "or" and "nor" also require special attention for acceptable automated encoding. When the words "or" and "nor" join operands that are listed in order of priority, the operands cannot be swapped.

Therefore, to ensure correct semantics, we will concentrate on encoding specific sentences with Boolean operators. If both operands are preceded by the specifying words "the" and "to," then we can swap the operands since the main verb has already been stated. In addition, we will automate the encoding of operands of the correlative conjunctions such as "both _ and _," "either _ or _" and "neither _ nor _." Also, we will automatically encode lists of items while being sure not to encode sentences with words that show chronological order or priority.

Table 1 shows different instances of the Boolean operators in sentences which can be encoded automatically and still yield an acceptable encoded cover. The encoding of the order of the operators can be done based on any reversible procedure decided by between the sender and receiver of the cover document. For example, alphabetical order of the operands or summations of the ASCII values of the operands (as performed in Table 1) are two simple ways for choosing the encoding of the operands.

3.3. Rearranging Conditional Sentence. The transformations discussed in the previous two subsections are classified as word-level transformations. In this section, we will look at clause-level transformations. In natural language text, many statements display cause-effect similar to "if-then" conditional statements. These English sentences are made up

Table 1: Boolean operators encoding.

	Sample sentence	Bit
AND	I would like to go to the park and to the zoo	0
	I would like to go to the zoo and to the park	1
NOR	Neither Arwa nor Jena wants to play	0
	Neither Jena nor Arwa wants to play	1
OR	We can eat cake, cookies, or ice cream	00
	We can eat cake, ice cream, or cookies	01
	We can eat cookies, cake, or ice cream	10
	We can eat ice cream, cookies, or cake	11

Table 2: Conditional structure format.

Format	Rearrangement of format
"if p, q"	"q, if p"
"for q, p"	"p, for q"
"q, when p"	"when p, q"
"q is necessary for p"	"for p, q is necessary"
"q unless ~p (not p)"	"unless ~p (not p), q"
"p if and only if q"	"q, if and only if p"

Table 3: Keywords for p → and → q type conditionals.

Keywords for two conditional statement types						
p → type	If	Despite	Because	When	Since	Unless
→ q type	For	In order for	To	The requirement for		

of an independent clause, which is a complete thought that can stand alone as a sentence and a dependent clause, which is a fragment that cannot stand alone as a statement. The dependent clause is normally a fragment because it begins with a subordinating conjunction, such as "if" or "for." Locating conditionals in natural language text and rearranging these conditionals can be used for hiding information. The basic idea is to look for the subordinating conjunction and the comma that symbols the end of a clause, check if the format of the sentence is suitable for automated rearrangement, and rearrange the conditional statement to represent the required bit of information.

If two propositions, p and q, appear in the form p → q, the statement is a conditional statement interpreted as "if p then q." The first proposition p is the hypothesis, and the conclusion is q. Two propositions, p and q, may also be in the form p↔q if the statement is implying "p if and only if q" [18]. Note that some forms begin with the hypothesis at the beginning of the statement (p →), while others begin with the conclusion (→ q). Utilizing the information in Table 2, we can easily transform conditional statements. For example, the sentence "If Alice is sick, she can go to the hospital" can be transformed to "She can go to the hospital, if Alice is sick." The remainder of this section will show how a conditional statement can be encoded by 4 bits. One bit is encoded for whether the subordinate clause is the first clause, one bit is encoded if the pronoun appears in the first clause, one bit is encoded if it is a p → or → q type conditional, and one bit is encoded for the synonym subordinating conjunction used.

3.3.1. Subordinate Clause Is the First Clause in the Statement. There are two ways to write a conditional statement. For example, when we have the subordinate clause first as in "Because Alice was sick, she went to the doctor," we can encode this as "0." While if we have the subordinate clause as the second clause, we can encode it as "1" and the sentence would read "Alice went to the doctor, because she was sick." Table 3 shows some of the keywords for p → type and → q type sentences.

3.3.2. Interchanging Subordinating Conjunctions. The scope of this work is to encode bits by syntactic manipulations, while avoiding changing the words of each statement so as to preserve the original meaning of the cover document. However, subordinating conjunctions can be considered as synonym pairs. Table 4 lists the synonym pairs considered in this study, for example, "because" and "since" may be used interchangeably without affecting the meaning of the sentence. If the sender and receiver have agreed that using "because" instead of "since" encodes a bit, then we have been able to encode one more bit. We do have to take care when interchanging the subordinating conjunctions, as some pronouns or articles might have to be exchanged as well.

Table 4: Synonym substitution for conditional keywords.

Although	0
Even though	1
If	0
Since	1
If not	0
Unless	1

3.3.3. Switching p → Type Statement into a → q Type Statement (and the Opposite). In general p → type statements include many of the subordinating conjunctions such as "if, since, because." The → q type statements utilize subordinating conjunctions such as "for, in order to, to…." We can simply identify the statement type depending on the subordinating conjunction employed. However, in order to transform p → type statement into a → q type statement (or the opposite), we need a special tool to generate the switching phrases from a sentence that begins with "if" to a sentence that begins with "for." A natural language text generator based on training sentences would be a likely candidate for such a tool. Automating the transform from p → type statement into a → q type statement allows us to encode another bit. The following table shows an example of how one sentence may be encoded by 2 bits depending on the type of statement and depending on whether the statement begins with the subordinating conjunction or not. Table 5 shows encoding two transforms to conditional statements: the interchanging of the conditional statement type and the swapping of the independent and dependent clause.

3.3.4. Pronouns and Their Identifiers. For correct encoding with regard to the pronoun, we must make sure that if there is a pronoun in the sentence, both the independent clause and dependent clause have the same pronoun. If one of the clauses has a pronoun and the other clause has the identifier of the pronoun, most likely we will not be able to simply interchange the clauses of the sentence and keep the semantics perfectly correct. For example, notice the sentence: "If Alice wants to see a movie, she must save up money." It may be slightly ambiguous as to who "must save up money" when we to transform it to "She must save up money, if Alice wants to see a movie."

For our purposes, this slight ambiguity is *acceptable* because our main objective is to provide a medium for steganography between two users who are not using completely proper English for their exchange cover documents.

Table 5: Encoding the interchanging of p → type and → q type conditionals and swapping the independent and Dependent clauses.

Conditional	Sample sentence	Type	Bit
If q, p	**If** you can see the stop sign, you are **not** blind	→ q	10
p, if q	You are **not** blind, **if** you can see the stop sign	→ q	11
Unless ~p, q	**Unless** you are blind, you can see the stop sign	p →	00
q, unless ~p	You can see the stop sign, **unless** you are blind	p →	01

Table 6: Encoding nested conditionals.

Nested conditional	Sample sentence	Type	Bit
Because p1, if p2, q	Because Hamza has four hundred dirhams, if he saves one hundred more, he can buy the new computer game	P—p-q	0
If p2, q, because p1	If he saves one hundred more, he can buy the new computer game, because Hamza has four hundred dirhams	P—q-p	1

However, in some sentences, not only is the subject unidentified in the first clause, but the object is also unidentified.

For example, if we swap the clauses of the conditional sentence, "Because the president promised to lower taxes, he must do so" then the sentence would read: "He must do so, because the president promised to lower taxes." "He must do so" is quite ambiguous as to who must do what. Therefore, it is better to ensure that at least the subject is identified in the first clause.

Identifying the subject in the first clause can be resolved with a speech tagger. When the tagger identifies an "NNP" or proper singular noun, such as "Alice" in the second clause and the tagger identifies a "PRP" or personal pronoun, such as "she" in the first clause, the main program can be instructed to swap these two words. The output of the speech tagger available at [19] when the input sentence was "Because she was sick, Alice went to the hospital" is as shown in Figure 1. This simple speech tagger's output shows that it is feasible to automate the identifier to be in the first clause. In addition, the interchanging of the pronoun and its identifier may be used to encode more bits. For example, if the pronoun tagged as "PRP" is in the first clause, then "0" can be encoded, and if the identifier tagged as "NNP" is in the first clause, then "1" can be encoded.

IN Because PRP she VBD was JJ sick , , NNP Alice VBD went T0 to DT the NN hospital .

Key:

(i) #: pound sign
(ii) $: dollar sign
(iii) ": close double quote
(iv) ' ': open double quote
(vi) ': close single quote
(vii) `: open single quote
(viii) NNP: proper single noun
(ix) NNPS: proper plural noun
(x) PDT: predeterminer
(xi) POS: possessive ending
(xii) PRP: personal pronoun

Figure 1: Output of speech tagger.

3.3.5. Nested Conditionals. In some statements there are more than one hypothesis, p leading to a conclusion q. For example, there are two hypotheses in the sentence "*When* you go to the supermarket, *if* you find ripe bananas, please buy some." The encoding scheme we have adopted, as shown in Table 6, encodes "0" if the first clause is the dependent clause (begins with a subordinating conjunction) and "1" if the first clause is the independent clause. Since we have two dependent clauses, the encoder will combine the innermost nested clauses and consider them as one clause.

Note that there might be other options for moving the clauses around; however, they might change the meaning of the statement. Table 6 shows also how the decoder will be able to decode the nested conditional. When there are two subordinate clauses (p) followed by the independent clause (q), the decoder automatically decodes a "0." When there is a dependent clause followed by an independent clause and ending with another dependent clause, the decoder outputs a "1." If we look closely at the sample sentence of Table 6 encoded as "1," it may seem a little conspicuous. Normally, a sentence with the same meaning would be written as "If Hamza saves one hundred more, he can buy the computer game, because he already has four hundred dirhams" (the pronoun is in the first clause). The next subsection shows how we deal with pronoun issues. Finally, the maximum possible encodability of a conditional statement utilizing (1) conditional, (2) statement type, (3) pronouns, and (4) synonym substitution is shown in Table 7.

3.4. Multiple Adverbs a Clause Make the Receiver Applause. One of the most manipulative word types is the adverb, as it can be placed nearly anywhere in the clause it belongs to. To maximize the encoding capacity, we can add adverbs at all possible location in the sentences. When an adverb is found before the subject it will be considered as hiding "1." When an adverb is not found before the subject, it will be considered as a "0." Similarly for verbs and objects, if an adverb is found

Table 7: Summary of conditional transforms. One statement is encoded in 16 different ways.

Conditional	Various forms of the sentence	S	B	P	T
Because p, q	Because Alice had an accident, she quickly rushed to the hospital	0	0	0	0
q, because p	Alice quickly rushed to the hospital, because she had an accident	1	0	0	0
Because p, q	Because she had an accident, Alice quickly rushed to the hospital	0	0	1	0
q, because p	She quickly rushed to the hospital, because Alice had an accident	1	0	1	0
For p, q	For Alice to quickly rush to the hospital, she must have had an accident	0	0	0	1
q, for p	Alice must have had an accident, for her to quickly rush to the hospital	1	0	0	1
For p, q	For her to quickly rush to the hospital, Alice must have had an accident	0	0	1	1
q, for p	She must have had an accident, for her to quickly rush to the hospital	1	0	1	1
Since p, q	Since Alice had an accident, she quickly rushed to the hospital	0	1	0	0
q, since p	Alice quickly rushed to the hospital, since she had an accident	1	1	0	0
Since p, q	Since she had an accident, Alice quickly rushed to the hospital	0	1	1	0
q, since p	She quickly rushed to the hospital, since Alice had an accident	1	1	1	0
In order p, q	In order for Alice to quickly rush to the hospital, she must have had an accident	0	1	0	1
q, in order p	Alice must have had an accident, in order for her to quickly rush to the hospital	1	1	0	1
In order p, q	In order for her to quickly rush to the hospital, Alice must have had an accident	0	1	1	1
q, in order p	She must have had an accident, in order for Alice to quickly rush to the hospital	1	1	1	1

S = subordinating conjunction is the first word of the sentence (e.g., If, For); B = Synonym substitution (e.g., Because = "0", Since = "1"); P = pronoun in first clause; T = type (p-type = 0, q-type = 1).

before the main verb it will encode "1," and if there is no adverb before the verb that encodes "0." For example, take the simple minimum sentence of only a subject and a verb: "He ran." This concise statement is also the main independent clause of the statement, as it is the only clause in the statement. "He ran" encodes the bits 00 because there is no adverb placed before the subject and no adverb placed before the verb. In order to let this simple 2-word statement encode 01, we add in the adverb before the main verb, making the sentence "He quickly ran." We base the choice of the newly inserted adverbs upon the *n*-gram's occurrence according to the Google Books Corpus. The Google Books Corpus shows us the *n*-grams appear in recent text and how frequent they are.

We choose the most frequently appearing adverbs to avoid altering the meaning of the original cover text. Take a look at the simple conditional statement: "If it rains, we will stay home." Since this is a conditional statement, we can write it also in the form: "We will stay home, if it rains." However, we were only able to encode one bit. We want to add in some descriptive words to each clause. Here we concentrate on verb-adverb pairs. We check the most commonly used adverb per verb and insert it. In Figure 2, we show the occurrence of "stay happily" as opposed to "stay sadly" and the occurrence of "rains heavily" as opposed to "rains lightly." We find that "stay sadly" and "rains lightly" never occurred in the text over about 40 years. These phrases probably never occurred because such phrases are not significant enough meaning-wise to publish in text. Therefore, we use the more commonly occurring verb-adverb phrases, such as "rains heavily" and discard nonoccurring *n*-grams such as "rains lightly." This technique of checking the Google Books Corpus for the most occurring *n*-grams over the past 40 years serves to ensure that only the most-fitting adverbs are inserted, and other adverbs are discarded. In this manner, sentence grammar and meaning is preserved, while every sentence can encode at least two bits.

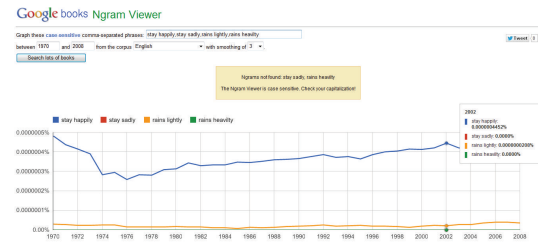

Figure 2: Google *n*-gram Viewer showing the most commonly found adverbs modifying their verbs.

3.5. Abbreviating Keywords. Through statistical analysis, we were able to discover that some of the most common words in the English language are words that can be abbreviated such as Street (St.). This is probably exactly why these keywords are abbreviated in the first place, because they are so common so there is no need to waste resources to always write them out. Certain casual texts may contain keywords that are often understood whether or not they are abbreviated or not. For example, in a sample text in which one party describes to another party the address of a place, the author may appear to not be consistent and sometimes abbreviate the word "street" or "boulevard." However, the actual truth is that when a proper address contains "St." it is hiding a "1," while if the word is written out as "Street" it is hiding a "0" bit. Likewise, if the text mentions a date, if the month is written out fully it may represent "1," while, for example, writing Jan. 31st may represent "0." We realize the inconsistency of abbreviating

and nonabbreviating common English statements such as "do not" and "do not," and therefore a secret key can be established between the sender and receiver to decide which keywords to abbreviate and which to ignore. For example, every 5th keyword can be encoded rather than every possible keyword in order not to arise suspicion.

3.6. Encoding Sentences with Multiple Transforms. These transforms ensure that any sentence, even the simplest 2-word sentence, such as "He ran" can at least hide two bits by inserting adverbs to describe the subject and verb. In this section we will show how one sentence can take on multiple transforms. We will study this sentence in particular: "The new groom, Khalid, can try on the plaid suit or the striped suit, if he likes." This sentence has a conditional, in which four bits can be hidden. The first clause has a subject "groom", and verb "can", and objects "plaid suit" and "striped suit", meaning it can also hide 4 bits by adding –ly adverbs. The second clause, "if he likes," has a subject and a verb, so two adverbs can be added in, hiding 2 more bits. Note that if the two adverbs are not inserted, the clause is encoded as "0 0." For instance, the sentence can be for coded as "1 1 1 1 0 0" by the following adverb insertions: "*Today*, the new groom, Khalid, can *definitely* try on the *slightly* plaid suit or the *darkly* striped suit, if he likes." The bits "1111" were encoded because adverbs could be found before the subject, verb and both objects, and the "0s" were encoded because no adverb could be found before the subject "he" and the verb "likes". The sentence also has a Boolean operator "or" where one bit is hidden and it has a separable phrasal verb "try on" where one bit is hidden. Finally, the commas around the name Khalid can be kept or taken out for hiding a bit of information. The total number of bits that this sentence, which is only comprised of 17 words, hides is 13 bits.

4. Transforms Evaluation

In the previous section, we have described various placement transforms and how they can be used to encode a cover document. This section will discuss the embedding capacity that these transforms can provide in a random page of text. To obtain the capacity for information hiding provided by our system, we have evaluated each transform for its frequency of occurrence according to the number of times a keyword for a transform appears in the Google *n*-gram Viewer of the Google Books Corpus [20]. This is the largest corpus, containing over 1 trillion word tokens of English text on publicly accessibly web pages collected in January 2006. It contains English *n*-grams and their observed frequency counts, so long as the token has been observed at least 40 times.

We have used the "English" genre of books and obtained results from the years of 1970–2008. This has given us a moving average of the occurrences of the keywords of a transform. We have chosen a fairly recent time frame because our scenario deals with everyday English, which can change vastly from generation to generation.

The transforms discussed in Section 3 have shown to be independent of each other; therefore, the capacity for encoding a random cover document will be considered as the total percentages of appearance of all the keywords per a Google Book statement. Table 8 gives an example of the statistics obtained for keywords of each transform type.

Table 8: Examples of keyword frequencies.

Transform type	Keywords	Frequency
Conditional	If	0.0525%
	for	0.64%
Adverb	He	0.13%
	he	0.36%
Boolean	and the	0.18%
Phrasal verb	put on	0.00145%

The keywords for adverb insertion are subjects, verbs and objects. Subjects and objects are generally nouns; therefore we count the keywords for adverb insertion to be the number of nouns and verbs. "The Second Edition of the 20-volume Oxford English Dictionary contains full entries for 171,476 words in current use... Over half of these words are nouns, about a quarter adjectives, and about a seventh verbs;..." [17]. We cannot search for all the 86,000 nouns and 24,000 verbs. Therefore, we assume that half the nouns are subjects and half of them are objects, or 43,000 words to represent subjects and 43,000 words to represent objects. In order to find the occurrence of subjects, we have found that the frequency of the occurrence of one noun to represent a subject, "he" (occurs 0.019%), and multiplied it by 43,000 to get 8.17% of words are subjects, and likewise using the object, "cat" (occurs 0.002%), shows that 0.86% of the words in a sentence are objects. To represent verbs, we used the word "walked" (occurs 0.004%) and multiplied it by one-seventh of the total number of English words to get the verb keyword occurrence to be 0.96%. Therefore adverb insertion occurs for about 10% of sentence words.

The keywords of a conditional statement are the subordinating conjunctions. Subordinating conjunctions make up about 3% of English language words according to the Google Books Corpus. Because we are only programming the correlative conjunctions of Boolean operators, the keywords of a correlative conjunction appeared as 0.3% of the words in the Google Books Corpus. Finally, separable phrasal verbs made up only approximately 0.1% of the words in the "English" genre of the Google Books Corpus. This number is quite small due to two reasons. Firstly, we can only search for the phrasal verb that is not separated as we cannot try out all the possible objects that can come between the verb and its adverb particle.

In addition, we hypothesize that phrasal verbs will appear more in the "Fiction" genre of the Google Books Corpus, as phrasal verbs are generally used in dialogue, and the "Fiction" genre is more likely to have more dialogue.

The total percentage of all the keywords of the transforms comes to be 13.4%, meaning that the keywords of the transforms comprise 13.4% of all the words of the "English" genre of the Google Books Corpus. This total percentage, 13.4%, was multiplied by the average number of words per

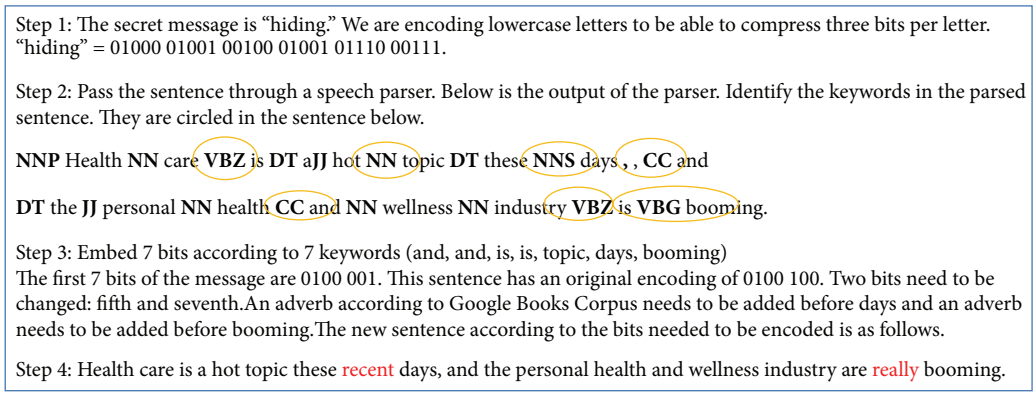

Figure 3: An example of a sentence being encoded.

sentence, which is 23.85 words according to [21]. As a result, a new measuring rubric was generated: the average appearance of a keyword, per sentence of a Google Book:

$$\frac{\%\text{avg marker word appearance}}{\text{Google Book statement}} = \frac{\%\text{avg appearance}}{\text{Google Book}} \times \frac{\text{avg \# words}}{\text{statement}}. \quad (2)$$

The average number of words per sentence is 13.4% × 23.85 = 319%. This means that every 0.313 sentences can hide one bit. A page of text, which may contain approximately 20 sentences, can hide 63.8 bits of secret message. This is almost three times the encoding capacity (this does not account for abbreviations and comma insertions) achieved in [12]. Thus, on average, the proposed technique can encode each sentence with 3.2 bits of the secret message.

5. The Encoding and Decoding Process of a Cover Text

Before encoding can begin, the sender and receiver should each have the secret key, which determines which sentences have been encoded, which transforms are going to be implemented, and which encoding of the transform would represent "1," and which encoding would represent a "0." We discuss the details of the secret key and evaluate the security of the system in Section 6. Once the secret key has been established, a communication channel between the sender and receiver can be utilized. Listed below are the steps performed on the cover document in order to embed a specific secret message of bits.

Step 1. Obtain the bit representation of the ASCII letters of the secret message; remove the leading the 0 s.

Step 2. Identify a set of words as a sentence (find a capital letter and a full stop/question mark/exclamation mark) and insert the complete sentence into the parts of speech parser.

Step 3. Identify key words according to the parser output (conditional words, nouns, verbs, and adverbs). The number of key words will determine the number of message bits that can be encoded.

Step 4. Apply transformations according to the message bits required. Follow the transformations hierarchy (clause-level transforms first and then word-level transforms). For adverb insertion, use the corresponding noun or verb in Google Books Ngram Viewer for the most occurring *n*-grams. Figure 3 shows an example of how the four-steps encoding process is performed.

Now simply, the decoder reads the sentence, identifies the keywords, and quickly decodes the secret message according to the placements of the clauses and whether or not there are adverbs.

5.1. The Procedure for Applying Multiple Transformations according to the Transforms Hierarchy. The encoding process ensures that any sentence, even the simplest 2-word sentence, such as "He ran" can at least hide two bits by inserting adverbs to describe the subject and verb. This subsection shows how a sentence is encoded according to the proposed transformations hierarchy. The transformations hierarchy is shown in the left column of Table 9. We first encode conditionals, adverbs, phrasal verbs, Boolean operators, and finally commas and abbreviations. Tables 9 and 10 show examples of Step 4 (applying the transformations on a particular sentence with keywords to hide secret message bits).

We will begin by studying the sentence (which is a conditional with a subject, verb, and object): "If Sara wants strawberries, she can have some" (Table 9). At the top of the transformations hierarchy we have the conditionals because they are clause-level transforms. They are involved with swapping entire clauses. Since the subordinate clause beginning with "If Sara" is at beginning of the sentence, when a receiver receives the sentence: "If Sara wants strawberries, she can have some," the receiver will automatically be able to extract 0 00 0 0 for the bits pertaining to conditionals. In addition, the receiver will identify that this sentence has a subject, 2 verbs, and an object. Since the sentence received does not have any adverbs before the subject (Sara), before the two verbs (wants and can) and after the object (strawberries),

Table 9: Example of the transformation process applied to the sentence "If Sara wants strawberries, she can have some." The sentence was transformed to "Since Sara wants strawberries, she can definitely have some" which embeds the 9 bits of a secret message: 0 00 0 1 0 10 1.

Category	Number of bits	Transform	Bit encoding of the sentence: "If Sara wants strawberries, she can have some"	Bit
(I) Conditional	1	(1) Subordinate conjunction (if) in 1st clause	**If Sara wants strawberries, she can have some**	**0**
			She can have some, if Sara wants strawberries	1
	2	(2) Pronoun (she) in the 2nd clause	**If Sara wants strawberries, she can have some**	**00**
			If she wants strawberries, Sara can have some	01
			If Sara wants some, she can have strawberries	10
			If she wants some, Sara can have strawberries	01
	1	(3) p-type or q-type (if p, q)	**If Sara wants strawberries, she can have some**	**0**
			For Sara to have some strawberries, she has to want them	1
	1	(4) Synonym pair (if and since are interchangeable)	If Sara wants strawberries, she can have some	0
			***Since* Sara wants strawberries, she can have some**	**1**
(II) Adverb insertions	1	(5) Adverb before the subject "she"	**If Sara wants strawberries, she can have some**	**0**
			If Sara wants strawberries, tomorrow she can have some	1
	2	(6) Adverbs before the main verbs "wants" and "can"	If Sara wants strawberries, she can have some	00
			If Sara really wants strawberries, she can have some	01
			If Sara wants strawberries, she can *definitely* have some	**10**
			If Sara really wants strawberries, she can definitely have some	11
	1	(7) Adverb after the object "some"	If Sara wants strawberries, she can have some later	0
			If Sara wants strawberries, she can have some later	**1**
	Total 9 bits	**Final sentence**	*"Since Sara wants strawberries, she can definitely have some later"*	

Table 10: Example of the transformation process applied to the sentence "He can try on the plaid suit or the striped suit." The sentence was transformed to "He can quickly try the striped or plaid suit on" which embeds the 5 bits of a secret message: 0 1 0 1 1.

Category	Number of bits	Transform	Bit encoding of the sentence: "He can try on the plaid suit or the striped suit"	Bit
(I) Adverb insertions	1	(5) Adverb before the subject "He"	**He can try on the plaid suit or the striped suit**	**0**
			Today he can try the plaid suit or the striped suit on	1
	1	(6) Adverb before the main verb "try"	He can try on the plaid suit or the striped suit	0
			He can quickly try on the striped suit or the plaid suit	**1**
	1	(7) Adverb after the object "suit"	**He can try on the plaid suit or the striped suit**	**0**
			He can try the plaid suit or the striped suit on today	1
(II) Phrasal verb	1	(8) Separable Phrasal Verb	He can try on the plaid suit or the striped suit	0
			He can try the plaid suit or the striped suit on	**1**
(III) Boolean operators	1	(9) Operands around "or"	He can try on the plaid suit or the striped suit	0
			He can try on the striped suit or the plaid suit	**1**
	Total 5 bits	**Final sentence**	*"He can quickly try the striped suit or the plaid suit on"*	

the receiver will be able to extract 0 00 0 pertaining to adverb insertion.

This sentence therefore originally encodes 0 00 0 0 0 00 0 (5 bits are read from conditionals and 4 bits are read from adverb insertion), achieving 9 bits in a simple 8 word sentence. If a sender wants to send this cover sentence but does not want the blind receiver to receive 000000000, the sender needs to apply transforms to encode the particular bits to be sent. Table 9 shows the step-by-step encoding process, according to the keywords of the sentence "If Sara wants strawberries, she can have some," so that a blind receiver is able to extract 0 00 0 1 0 10 1.

6. Security Analysis

In the previous sections we have detailed the manipulation and coding of English sentences for hiding information. Through a statistical analysis, we have also shown that the keywords which trigger the sentence manipulations are widely found in English text. Our system is based on 13 reversible transforms such that a specific transform found in a sentence can be coded as "0" or "1." Most of these transforms are explained in Tables 9 and 10.

This section will first describe the secret key that the sender and receiver agree upon. Next, a security discussion will illustrate the robustness of the proposed the system against possible attacks. The discussions include the cases where the attacker knows in advance how the transforms are applied and try to retrieve secret messages. Finally, we will describe our experiments whose results show that our transforms are inconspicuous enough that a passive attacker would most likely not suspect a hidden message in the first place.

6.1. Secret Key. In this section we will show how the sender and receiver use a secret key to secure their transmission of secret messages against passive attacks. Firstly, the sender and receiver should agree on which manipulation of the keywords of a sentence (transform) encodes "0" and which manipulation encodes "1." For example, having the Boolean operand around the keyword "or" with the lower ASCII value appear first may encode "0," while having the Boolean operand around the keyword "and" with the lower ASCII value appear first may encode "1." Table 9 gives sample sentences with sample bit encodings; however, the true encoding given is part of the secret key and is decided between the sender and receiver. Let m be the number of transforms our system provides. Therefore, there are 2^m possible encodings. Since we have listed 13 different transforms that can be found, then we have 2^{13} different encoding combinations.

The other element of the secret key is which transforms the sender and receiver will agree to consider and which transforms they will perform, simply for obscurity. Let n be the number of transformations a cover text can provide; in other words, the total possible number of bits that can be hidden. One page of English text contains at least 20 sentences, while each sentence is about 23.5 words. Each sentence can definitely take on a descriptive adverb to describe the subject, verb, and object; therefore each sentence encodes easily at least 3 bits. According to the adverb transforms shown in this paper, alone, we have shown that one page of English text can take on more than 60 transforms or encode more than 60 secret message bits. Also, conditional keywords have been shown to appear in at least 3% of English words; therefore in a page of about 20 sentences that are 23.5 words long, we can encode about 14 bits. From adverbs and conditionals alone, we can encode 74 bits per page or 3.7 bits per sentence. In order to provide more security, the sender and receiver agree on only a certain number of bits to be transmitted per page; say, for example, 25. Part of the secret key is that the 74 possible encodings are they going to choose to carry the actual 25 bits of the secret key. They may agree to consider one transform and skip three bit encodings or to consider the first 25 transforms as the actual encodings and keep the 49 remaining transforms for obscurity.

6.2. Security Discussion. Let h be the number of bits of a secret message that a sender wants to send a receiver. As mentioned previously, n is the total number of keywords found in a cover text; that is, the total number of bits that can be hidden in a cover text. The sender will only send h bits and will discard n-h bits. Therefore, the security provided by choosing to encode some bits and choosing to discard other bits is ${}^nC_h$ (n choose h). Since 2^m is the obscurity provided by the agreement between the sender and receiver for each transform instance to represent either "0" or "1," then we can say that the total security provided by the secret key is $2^m({}^nC_h)$. To illustrate, let us say that Alice wants to send Bob a secret message of 25 bits in one page of text which can handle 74 transforms. The security provided is $2^m(n!/(h! * (n-h)!)) = 2^{13}(74!/(25! * (74-25)!)) = 287{,}203{,}585{,}453{,}527{,}824{,}400{,}384$ possible encodings for a brute force attack to attempt to decode the secret message. That is of course, after actually suspecting that this page of text may contain a secret message. Our transforms have been carefully chosen and tested in order not to make Wendy the Warden suspicious. However, in the event that Wendy decides to check a page of text for the secret message, we have shown that it will be quite difficult for her to retrieve the secret message correctly.

6.3. Measuring the Inconspicuousness of a Cover Text. In order to measure whether or not our transforms greatly impacted the grammar and meaning of ordinary English language sentences, we devised an experiment, which involved encoding sentences from the most widely read English blog, the Huffington Post [22] that has an average monthly visitors rank of approximately 110 million visitors.

To prove that our encoding and decoding of sentences would not raise the suspicion of a passive attacker, we carried many experiments by asking subjects to assess the quality of the English statements before and after encoding the secret message. The subjects that helped testing the experiment hypnosis are English teachers, technical professionals, and university students who are native English speakers or are using in English in daily basis. We carried two types of experiments to check the inconspicuousness of the cover text after applying various transformations:

(1) preservation of the style;

(2) Turing-like test for steganography.

The purpose of the first experiment is to check if the embedding process has affected the writing style, in the sense that whether a careful reader would find the text a bit unusual (recall that the proposed transformation guarantees grammar correctness and semantically preserving the meaning). In this experiment, subjects are given a set of statements, some of which have a secret message encoded. The statements are taken from The Huffington Post blog, the most widely visited blog on the Internet. The subjects are requested to fill a rubric about the style of the statements. The subject checks one of the choices as follows:

(i) wrong grammar;

(ii) odd: sounds like wrong English structure;

(iii) correct but very unusual;

(iv) think about it for a moment then pass it;

(v) perfectly normal.

Note that the subject does not know which is the default statement (taken directly from the Huffington Post) and which is the transformed one. Many statements are taken from the same article in the Huffington Post so that we can test whether or not the writing still remains connected and the meaning is preserved even after a few of the statements are transformed. We repeated this experiments many times with more than ten different subjects. We collected the statistics about the subject response after segregating the response of the default statements (statements drawn from the blog before subjecting them to transformation) and the response of the transformed statements. The results show that the opinions of the subjects are always with ±1% from each other. This shows that the proposed transformations do not have notable effect on the language style.

The purpose of the second set is to check whether the proposed manipulations raise any suspicion of information hidden in a text. Subjects are informed that some of the statements might have hidden secret messages and others are not. Watermarked and nonwatermarked texts are presented to test subjects who judge whether the text has embedded information or not. Subjects, in this set of experiments, consist of English teachers, technical professionals, and students who native English speakers or are using in English in daily basis. After analyzing the answers of the subjects, the results show that there is no statistical difference between the two classes of statements, namely, the default and transformed. This indicates that the proposed transformations, and thus the encoding method employed, do not introduce changes that raise suspicions of hidden text messages.

7. Conclusion and Future Work

In this paper, we proposed a new technique for hiding secret messages in ordinary English text. The proposed scheme exploits the redundancy that exists in English language sentence constituents without jeopardizing the integrity of the grammar and meaning of the sentence. We have shown the feasibility of automating the different transforms by using a sentence parser for sentence manipulation. Using Google Books Corpus, we have shown that these transforms are widely found in English text. As the transforms are performed on different sentence parts and at different levels of granularity, they are independent and thus can be applied on the same statement. One of these transformations, adverb insertion, guarantees that each sentence can hide at least 2 bits. However, most ordinary English sentences can be encoded by more than one transform. Therefore, the average encoding capacity of the proposed technique is 3.2 bits per sentence. The security analysis showed that the proposed secret key is good enough to thwart the possibility of extracting the secret message by a malicious third party. Finally, we showed experiments that measure the effect of the proposed transforms on the inconspicuousness of transformed sentences. We asked subjects (English teachers and other native English speakers from different professions) to compare the language style of the original sentence and the transformed sentences. We used excerpts, from the most widely read blog *The Huffington Post*, that covers various topics and impeded secret messages in some. Subjects could not find noticeable difference between the transformed and original sentences.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

[1] C. Y. Chang and S. Clark, "Linguistic steganography using automatically generated paraphrases," in *Proceedings of the Annual Conference of the North American Chapter of the ACL*, pp. 591–599, Los Angeles, Calif, USA, June 2010.

[2] L. Y. Por and B. Delina, "Information hiding: a new approach in text steganography," in *Proceedings of the 7th WSEAS International Conference on Applied Computer & Applied Computational Science*, pp. 689–695, Hangzhou, China, April 2008.

[3] T.-Y. Liu and W.-H. Tsai, "A new steganographic method for data hiding in microsoft word documents by a change tracking technique," *IEEE Transactions on Information Forensics and Security*, vol. 2, no. 1, pp. 24–30, 2007.

[4] Z. Fu, X. Sun, J. Zhang, and B. Li, "A novel watermark embedding and detection scheme based on zero- knowledge proof," *International Journal of Digital Content Technology and Its Applications*, vol. 5, no. 3, pp. 273–286, 2011.

[5] A. Desoky, "Innocipher: a novel innocent-cipher-based cryptography paradigm-high level of security for fooling the enemy," *Information Security Journal*, vol. 22, no. 2, pp. 83–97, 2013.

[6] M. Chapman, "NICETEXT," 2014, ftp://www.zedz.net/pub/security/steganography/nicetext/.

[7] Spammimic, February 2014, http://www.spammimic.com.

[8] V. Chand and C. O. Orgun, "Exploiting linguistic features in lexical steganography: design and proof-of-concept implementation," in *Proceedings of the 39th Annual Hawaii International Conference on System Sciences (HICSS '06)*, p. 126, January 2006.

[9] M. Grosvald and C. Orgun, "Human- versus computer-generated text-based steganography: real-world tests of two algorithms," *Journal of Information Hiding and Multimedia Signal Processing*, vol. 3, no. 1, 2012.

[10] G. R. S. Weir and M. Morran, "Hiding the hidden message: approaches to textual steganography," *International Journal of Electronic Security and Digital Forensics*, vol. 3, no. 3, pp. 223–233, 2010.

[11] U. Topkara, M. Topkara, and M. J. Atallah, "The hiding virtues of ambiguity: quantifiably resilient watermarking of natural language text through synonym substitutions," in *Proceedings of the Multimedia and Security Workshop (MM '06)*, pp. 164–174, Geneva, Switzerland, September 2006.

[12] I. Kamel and S. Banawan, "Hiding information in the placement of maneuverable words," in *Proceedings of the International Conference on Innovations in Information Technology (IIT '12)*, pp. 255–260, Al Ain, UAE, March 2012.

[13] M. J. Atallah, V. Raskin, M. C. Crogan et al., "Natural language watermarking: design, analysis, and a proof-of-concept implementation," in *Workshop on Information Hiding*, vol. 2137 of *Lecture Notes in Computer Science*, pp. 185–200, Springer Berlin Heidelberg, Pittsburgh, Pa, USA, 2001.

[14] J. M. Topkara, U. Topkara, and M. J. Atallah, "Words are not enough: sentence level natural language watermarking," in *Proceedings of the 4th ACM International Workshop on Contents Protection and Security (MCPS '06)*, pp. 37–46, Santa Barbara, Calif, USA, October 2006.

[15] H. M. Meral, B. Sankur, A. Sumru Özsoy, T. Güngör, and E. Sevinç, "Natural language watermarking via morphosyntactic alterations," *Computer Speech and Language*, vol. 23, no. 1, pp. 107–125, 2009.

[16] Dictionary of English Phrasal Verbs, 2013, http://www.usingenglish.com/.

[17] Oxford Dictionaries, December 2013, http://www.oxforddictionaries.com.

[18] K. Rosen, *Discrete Mathematics and Its Applications*, McGraw-Hill, New York, NY, USA, 7th edition, 2012.

[19] Cognitive Computational Group and University of Illinois at Urbana Champaign, "Part of Speech Tagger Demo Results," http://cogcomp.cs.illinois.edu/demo/pos/results.php.

[20] Google Books Ngram Viewer, 2015, http://books.google.com/ngrams/datasets.

[21] J. Nivre, *Inductive Dependence Parsing*, vol. 34 of *Text, Speech and Language Technology*, Springer, Berlin, Germany, 2006.

[22] The Huffington Post, 2015, http://www.thehuffingtonpost.com.