A New Property Coding in Text Steganography of Microsoft Word Documents

Ivan Stojanov, Aleksandra Mileva, Igor Stojanović

University of Goce Delčev
Štip, Macedonia
Email: {ivan.stojanov, aleksandra.mileva, igor.stojanovik}@ugd.edu.mk

***Abstract*—Electronic documents, similarly as printed documents, need to be secured by adding some specific features that allow efficient copyright protection, authentication, document tracking or investigation of counterfeiting and forgeries. Microsoft Word is one of the most popular word processors, and several methods exist for embedding data specially in documents produced by it. We present a new type of methods for hiding data in Microsoft Word documents, named as Property coding, which deploys properties of different document objects (e.g., characters, paragraphs, and sentences) for embedding data. We give four different ways of Property coding, which are resistant to save actions, introduce very small overhead on the document size (about 1%), can embed up to 8 bits per character, and of course, are unnoticed by readers. Property coding belongs to format based methods of text steganography.**

Keywords–Data Hiding; Microsoft Word.

I. INTRODUCTION

Steganography is the art of undetectably altering some seemingly innocent carrier to embed or hide secret messages. Modern digital steganography utilizes computers and new information technologies, and one can use an image, text, video, audio, file, protocol header or payload, or similar, as a carrier. Watermarking, on the other hand, is the art of imperceptibly altering some carrier, to embed a message about that carrier. Each steganographic and watermarking system consist of an embedder and a detector, the carrier is called cover work, and the result of embedding is called stego (watermarked) work [1]. Information hiding (or data hiding) is a general term encompassing a more wide range of problems, and it includes steganography and watermarking also.

Text steganography refers to the hiding of information within text (see surveys [2][3]). Text is one of the oldest media used for hiding data, and before the time of digital steganography, letters, books, and telegrams were used to hide secret messages within their texts. Also, text documents are the most present digital media today, which can be found in the form of newspapers, books, web pages, source codes, contracts, advertisements, etc. So, development of text steganography and steganalysis is very important. From one side, data hiding methods in text documents are big threats to cybersecurity and new communication tools for terrorists and other criminals. On the other side, these methods can have legal application in document tracking, copyright protection, authentication, investigation of counterfeiting and forgeries, etc. [4][5][6].

Microsoft Word is one of the most popular document and word processing software, which comes as a part of the Microsoft Office package. It is attractive for average users because of the easiness of text editing and richness of text formatting features.

In this paper, we present four new methods for hiding data in MS-Word documents. We use properties of different document objects, like characters, paragraphs, and sentences, for data hiding. Additionally, these techniques can be adjust for using in the documents produced by other word processors, like Apache OpenOffice, Corel WordPerfect, etc. Section II is devoted to different techniques used in text steganography and Section III gives several existing methods and techniques specially designed for MS-Word documents. Our four new methods are presented in Section IV, and experimental results and discussion are given in Section V.

II. TEXT STEGANOGRAPHY

There are three main categories of text steganography: format based methods, random and statistical generation, and linguistic methods [7].

A. Format based methods

Format based methods generally format and modify existing text to conceal the data. There are several different techniques for hiding data in text documents presented bellow. Some of them like line shift coding or inserting of spacial characters can pass unnoticed by readers, but can be detected by computer; and other like font resizing, can pass undetected by computer, but human can detect it. Hidden information usually can be destroyed for example by character recognition programs.

1) Line Shift Coding: In line shift coding, each even line is shifted by a small predetermined amount (e.g., 1/300 inch and less) either up or down, representing binary one or zero, respectfully [8][9][10]. The odd lines are used as control lines for detection of shifting of the even lines, and their position is static. In this way, the original document is not needed for decoding.

2) Word Shift Coding: Similarly to line shifting coding, in word shifting coding, each even word is shifted by a small predetermined amount (e.g., 1/150 inch and less) left or right, representing binary one or zero, respectfully [9][10]. Again, each odd word serves as a control word, which is used for measuring and comparing distances between words. Since the word spacing in the original document is not uniform, the original document is needed for decoding. Low, Maxemchuk, Brassil, and O'Gorman [8] use combination of line and word shifting, and each even line additionally is divided in three blocks of words and only middle block is shifted left or right. In [11], line is divided in segments of consecutive words, and neighbouring segments share one word. By shifting only middle words of the segment, 1 or 2 bits can be coded per one segment.

3) Feature Coding: In feature coding (or character coding), the feature of some characters in the text are changed [9][10]. For example, change to an individual character's height or its position relative to other characters; extending or shortening

 ISBN: 978-1-61208-376-6

of the horizontal line in the letter t; increasing or decreasing the size of the dot in letters i and j, etc. The last technique can be applied for 14 letters in Arabic alphabet [12]. Another feature coding methods for Arabic alphabet [13][14] use the redundancy in diacritics to hide information.

4) Open method: In this group of techniques, some special characters are inserted in the cover text. For example, spaces can be inserted at the end of each sentence, at the end of each line, between words [15], or at the end of each paragraph [16]. A text processor can change the number of spaces and destroy the hidden message. There are several software tools, which implement some variants of the open method, like SNOW [17], WhiteSteg [18], UniSpaCh [19] which uses Unicode space characters, etc.

Other techniques [20][21], which can be put in this group, use widening, shrinking or unchanging an inter-word space to encode the text format.

5) Luminance Modulation Coding: This coding uses character luminance modulation for hiding data. Borges and Mayer [22] embed data by individually altering the luminance of each character from black to any value in the real-valued discrete alphabet of cardinality S, so that each symbol represents $log_2 S$ bits. One previous method [23], instead of whole character, modulates the luminance of particular pixels from the characters in scanned text document for hiding bits. Similarly in [24], quantization of the color intensity of each character is used, in such a way the HVS cannot make the difference between original and quantized characters, but it is possible for a specialized reader. This technique works well on printed documents, too.

B. Random and Statistical Generation

In methods of random and statistical generation, a new text is generated, which tries to simulate some property of normal text, usually by approximating some arbitrary statistical distribution found in real text [7].

C. Linguistic Methods

Linguistic methods manipulate with lexical, syntactic, or semantic properties of texts for hiding data, while their meanings are preserved as much as possible. Known linguistic methods are syntactic and semantic methods.

With syntactic methods, data can be hidden within the syntactic structure itself. They sometimes include changing the diction and structure of text without significantly altering meaning or tone. Some of them use punctuation, because there are many circumstances where punctuation is ambiguous or when mispunctuation has low impact on the meaning of the text. For example, one can hide one or zero by putting or not, a comma before ”and” [15]. One disadvantage is that inconsistent use of punctuation is noticeable to the readers. In Arabic language, there is one special extension character, which is used with pointed letters, without effect on the content. The authors of [25] suggest to use pointed letters with extension as binary one and pointed letters without extension as binary zero. Wayner [26] proposed Context-Free Grammars (CFGs) to be used as a basis for generation of syntactically correct stego texts. Another method [27] manipulates with sentences by shifting the location of the noun and verb to hide data.

Semantic methods change the words themselves. One method uses the synonym substitution of words for hiding information in the text [15]. Two different synonyms can be used as binary one and zero. Similar is use of paraphrasing of text for hiding messages [28], for example ”can” for binary 0, and ”be able to” for binary 1. Another method [29] changes word spelling, and in order to code zero or one, the US and UK spellings of words are used. One example is the word ”color”, which has different spelling in UK (colour) and US (color). Other semantic methods are given in [5][30]. Semantic methods sometimes can alter the meaning of the sentence.

Different miscellaneous techniques that use typographical errors, using of abbreviations and acronyms, free form formatting, transliterations, use of emoticons for annotating text with feelings, mixed use of languages, and similar ones are given in [31].

III. Existing Methods Specially Designed for MS-Word Documents

Besides the previous more general text steganographic methods that can be applied, there are several methods for data hiding, specially designed for Microsoft Word documents. The most closest technique to ours, is usage of invisible characters, suggested by Khairullah [32]. This technique sets foreground color on invisible characters such as the space, the tab or the carriage return characters, obtaining 24 bits per character.

Another technique, called Similar English Font Types (SEFT) [33], use similar English fonts for hiding data. First, three different similar fonts are chosen (e.g., Century751 BT, CenturyOldStyle, CenturyExpdBT), and then, 26 letters and space character are represented by triple of capital letters, each in one of the chosen fonts.

Liu and Tsai [34] use Change Tracking technique for hiding data in MS-Word documents. First, a cover document is degenerated with different misspellings and other mistakes usual for users, and then, corrections with Change Tracking are added, so it seems like the document is the product of a collaborative writing effort. The secret message is embedded in the choices of degenerations using Huffman coding.

From MS-Office 2007, Microsoft has adopted a new format of its files, and introduced the Office Open XML (OOXML) format. In order to guarantee higher level of privacy and security, it has also presented the feature Document Inspector, which is used for quickly identifying and removing of any sensitive, hidden and personal information. Castiglione et al. present in [35] four new methods for hiding data in MS-Word documents, which resist the Document Inspector analysis. Two of them (with different compression algorithms or revision identifier values) exploit particular features of the OOXML standard, have null overhead, but do not resist to save actions. Other two (with zero dimension image or macro), resist to save actions, but they have an overhead.

IV. Property Coding

We present four new format based methods for hiding data in MS-Word documents, that use some text formattings that are invisible for human eye. They use different choices for some text properties, and because of that, we can name them as Property Codings. Methods presented in [32] and [33] can be also classified as Property codings, because they use font color and font type properties of a given character, respectfully. The novelty of our methods is twofold. First, we introduce other character properties that can be used for hiding data, and second, we show that properties of document objects other than characters (e.g., paragraphs and sentences), can be used for hiding data.

 ISBN: 978-1-61208-376-6

A. Method 1 - Character Scale

When we work in MS-Word, by default text character scale is set to 100%. Increasing the character scale will make your letters larger and scale further apart with more white space between each character. Decreasing the scale will shrink and squish letters closer together. Big differences in character scale are noticeable for human reader. But, if some of the characters are with scale 99% and others with 101%, human eye can not make the differences.

So, in the first method, we use scale of 99% to represent binary one, and scale of 101% to represent binary zero. Scale of 100% can be used for non-encoded characters. In this way, in the cover document, we can hide maximum the same number of bits as the number of characters in the document.

Variants of this method are also possible. For example, instead of using two very close scale values, one can uses four very close scale values **(e.g.,** 97%, 98%, 99% and 101%), and every value will represents two binary digits. In this way, we duplicate the hiding capacity of the same document, and still normal reader won't notice it. Another variant is to change scale on every word, not on every character.

B. Method 2 - Character Underline

One common feature of MS-Word is character underlining. There are 16 different underline styles, with potential of carrying 4 bits, and 2^{24} different underline colors. Because we need underlining to go unnoticed by the user, we use 16 variants of white color, with potential of carrying 4 bits.

In this way, we can hide 8 bits per character. Some characters, as g, j, p, q, and y, have noticeable changes in the look when we use every type of underlining. Because of that, we excluded this group of 5 characters from hiding data.

C. Method 3 - Paragraph Borders

In MS-Word, one can add border to the paragraph, sentence, picture, table, individual page, etc. Border can be left, right, top, bottom, etc. There are 24 different border styles, and only two of them (wdLineStyleEmboss3D and wdLineStyleEngrave3D) are noticeable to human reader. We can use 16 out of the rest 22, with potential of carrying 4 bits.

In this method, we use left and right borders on paragraph for hiding data. Again, we use 16 variants of white color for borders. Each paragraph in the cover document can hide 16 bits, in the following way - 4 bits from left border style, 4 bits from left border color, 4 bits from right border style, and 4 bits from right border color. This is done in our implementation.

We can increase hiding capacity of this method, by using different border width also. There are 13 border styles with 9 different border widths, two border styles with 6 different border widths, three border styles with 5 different border widths, one border style with 8 different border widths, one border style with 2 different border widths, and two border styles with 1 border width, or summary 155 possibilities. Potentially, we have 7 bits per combination border style/width. With experiments, we obtained that RGB colours represented with (R, G, B) components, where $R, G, B > 249$ can not be distinguished from the white color $(255, 255, 255)$. There are 216 different possibilities for colour, which can be used for representing 7 bits. Combining these two techniques, we can hide 28 bits, in the following way - 7 bits from left border style, 7 bits from left border color, 7 bits from right border style, and 7 bits from right border color.

TABLE I. CHARACTERISTICS OF THREE COVER DOCUMENTS

	Document 1	**Document 2**	**Document 3**
Pages	1	11	110
Words	340	2381	30907
Characters	2252	15493	190833
Paragraphs	13	82	802
Lines	42	328	3445
Sentences	21	134	2028
Original size (B)	31122	923090	4589312

TABLE II. COMPARISON OF MAXIMAL NUMBER OF EMBEDDED BITS/CHARACTERS IN OUR METHODS AND METHODS PRESENTED IN [32] AND [33]

	Document 1	Document 2	Document 3
Characters without q, j, p, q, y	2154	14823	182470
Invisible Characters	364	2515	31422
Percent of Invisible Characters	16,2	16,2	16,5
Capital Letters	40	286	4704
Max No. of embedded bits in Method 1	2252	15493	190833
Max No. of embedded bits in Method 2	17232	118584	1459760
Max No. of embedded bits in Method 3	364	2296	22456
Max No. of embedded bits in Method 4	147	938	14196
Max No. of embedded bits in [32]	8736	60360	754128
Max No. of embedded characters in [33]	13	95	1568
Max No. of embedded bits in [33]	104	760	12544

D. Method 4 - Sentence Borders

The final method uses sentence outside border for hiding data. We use only 8 border style out of 16, because other 8 can be noticed by human reader, and only the smallest border width of 0.25pt. Used border styles are wdLineStyleDashDot, wdLineStyleDashDotDot, wdLineStyleDashLargeGap, wdLineStyleDashSmallGap, wdLineStyleDot, wdLineStyleInset, wdLineStyleOutset and wdLineStyleSingle. Each sentence in the cover document can hide 7 bits, with 3 bits from outside border style, and 4 bits from outside border color.

V. EXPERIMENTAL RESULTS AND DISCUSSIONS

Each new presented method has implementation in C♯ using the Microsoft.Office.Interop.Word namespace. Our implementation of these four methods, use 8 bits to represent an extended ASCII character for all methods, except for the last, were we use 7 bits to represent an ASCII character. For our experiments, we use three types of MS-Word documents as cover documents - short, medium and large documents, with properties given in Table I.

For each cover document, we hide 10, 50, 100, 500, 1000, and 5000 characters (if it is possible), and we measure the size of the obtained stego document. Normally, the new size is bigger than original size, and it is given in bytes and in percent of increase of original size.

From the results in Tables III, IV and V, one can see that all techniques have small impact of document size, less then

TABLE III. EXPERIMENTAL RESULTS FOR DOCUMENT 1 WITH ORIGINAL SIZE OF 31122B

	10 characters		50 characters		100 characters		500 characters		1000 characters		5000 characters	
	Size	**%**	**Size**	**%**	**Size**	**%**	**Size**	**%**	**Size**	**%**	**Size**	**%**
Method 1	31448	1.01047	32347	1.03936	33390	1.04074	/	/	/	/	/	/
Method 2	31249	1.00408	31530	1.01310	31986	1.02776	34482	1.10796	37517	1.20548	/	/
Method 3	31295	1.00555	/	/	/	/	/	/	/	/	/	/
Method 4	31356	1.00751	/	/	/	/	/	/	/	/	/	/

TABLE IV. EXPERIMENTAL RESULTS FOR DOCUMENT 2 WITH ORIGINAL SIZE OF 923090B

	10 characters		50 characters		100 characters		500 characters		1000 characters		5000 characters	
	Size	**%**	**Size**	**%**	**Size**	**%**	**Size**	**%**	**Size**	**%**	**Size**	**%**
Method 1	923609	1.00056	924750	1.00179	925472	1.00258	934834	1.01272	946697	1.02557	/	/
Method 2	924243	1.00124	924605	1.00164	925180	1.00226	926341	1.00352	928582	1.00624	953474	1.03291
Method 3	923455	1.00039	924398	1.00141	924547	1.00157	/	/	/	/	/	/
Method 4	923587	1.00053	924013	1.00099	925290	1.00238	/	/	/	/	/	/

TABLE V. EXPERIMENTAL RESULTS FOR DOCUMENT 3 WITH ORIGINAL SIZE OF 4589312B

	10 characters		50 characters		100 characters		500 characters		1000 characters		5000 characters	
	Size	**%**	**Size**	**%**	**Size**	**%**	**Size**	**%**	**Size**	**%**	**Size**	**%**
Method 1	4589321	1.00000	4589363	1.00001	4591027	1.00037	4595001	1.00123	4605370	1.00349	4682285	1.02025
Method 2	4589313	1.00000	4589356	1.00000	4589574	1.00005	4592093	1.00060	4595782	1.00140	4608077	1.00408
Method 3	4589512	1.00004	4589567	1.00005	4589597	1.00006	4591231	1.00041	4593443	1.00090	/	/
Method 4	4589376	1.00001	4589396	1.00011	4591778	1.00010	4595859	1.00142	4603958	1.00319	/	/

1.206% for Document 1, less then 1.033% for Document 2, and less then 1.021% for Document 3 for evaluated message lengths. Method 2 has the smallest influence on the size for the short and large documents, and Method 3 has the smallest influence on the size for the medium document.

From the Table II, one can see that Method 2 has the highest embedding capacity, followed by Method 1, and the smallest embedding capacity has Method 4. The number of invisible characters is only a small portion of the number of all characters in every document, and in our three documents is less then 17% (see Table II). So, if we compare our Method 2 with the method introduced by Khairullah [32] (Table II), we can embed more characters by Method 2. One can see that for all three documents, the maximal number of embedded bits by [32] ('number of invisible characters' $\times 24$) is almost a half than the maximal number of embedded bits by Method 2 ('number of characters, without q, p, j, y, and g' $\times 8$). For the method proposed by Bhaya et al. in [33], we have that three consecutive capital letters in the document serve to embed one character, so, the maximal number of embedded bits depends strongly of number of capital letters. If we use 8 bits per character, we have that this method has the smallest embedding capacity compared to other analyzed methods (Table II). Even in the case that all characters are capital letters, we can embed almost three times less characters, than in the case of Method 2. Bhaya et al. in [33] suggested to use only three similar font types, which limits the maximal number of different characters that can be embedded to 27. This can be changed if we use four or five similar font types, resulting in up to 64 and 125 different characters. But finding bigger number of similar fonts is very difficult, and at the end, user may notice the differences in the font used for capital letters. Additional problem can arise if non-Latin language is used and if selected font is not present on the machine. For example, if you use Cyrillic letters, and font is not present, the capital letters will be displayed as Latin,

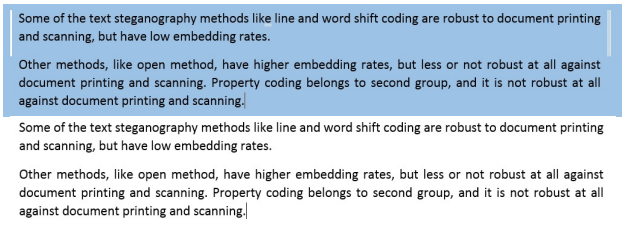

Figure 1. Detection of hiding with Method 2 and 3 by changing page background color

and coding will be visible to human eyes.

A. *Robustness and Steganalysis*

Some of the text steganography methods like line shift coding, word shift coding, and luminance modulation coding are robust to document printing and scanning, but have low embedding rates. Other methods, like open method, have higher embedding rates, but are less or not robust at all against document printing and scanning. Property coding belongs to second group, and it is not robust at all against document printing and scanning. Property Coding is resistant to save actions, compared to two methods presented in [35], and also has smaller overhead compared to other two methods from [35].

Hidden text with Property Coding can be changed or destroyed by text editing. The presence of Methods 2, 3, and 4 can be easily detected if somebody changes intentionally the background color of the document, causing the borders and underlining to became visible (see Figure 1). Method 1 is resistant to this kind of attack.

Property Coding is not entirely suitable for copyright protection applications where robust data-hiding is required, because the attacker can always use Optical Character Recog-

 ISBN: 978-1-61208-376-6

nition (OCR) to completely remove the hidden data.

VI. Conclusion

Four new format based methods specially designated for hiding data in MS-Words documents are given. Because they change the properties of some document objects offered by MS-Word, we called the new type of methods Property Coding. These methods are resistant to saving actions, introduce very small overhead on the document size, and can embed up to 8 bits per character.

References

[1] I. J. Cox, M. L. Miller, J. A. Bloom, J. Fridrich, and T. Kalker, Eds., Digital Watermarking and Steganography. Elsevier Inc., Burlington, MA, 2008, ISBN: 978-0-12-372585-1.

[2] H. Singh, P. K. Singh, and K. Saroha, "A Survey on Text Based Steganography," in Proceedings of the 3^{rd} National Conference INDIACom-2009 2009, New Delhi, India, 2009, pp. 3–9.

[3] M. Agarwal, "Text Steganographic Approaches: A Comparison," International Journal of Network Security & Its Applications, vol. 5(1), 2013, pp. 91–106.

[4] M. J. Atallah et al., "Natural language watermarking: design, analysis, and a proof-ofconcept implementation," in Proceedings of the 4^{th} International Workshop on Information Hiding April 25-27, 2001, Pittsburgh, USA. Springer Berlin Heidelberg, Apr. 2001, pp. 185–200, Moskowitz, I. S., Ed., LNCS: 2137, ISBN: 978-3-540-45496-0.

[5] M. Atallah et al., "Natural Language Watermarking and Tamperproofing," in Proceedings of the 5^{th} International Workshop on Information Hiding October 7-9, 2002, Noordwijkerhout, Netherlands. Springer-Verlag Berlin Heidelberg, Oct. 2003, pp. 196–212, Petitcolas , F. A. P., Ed., LNCS: 2578, ISBN: 3-540-00421-1.

[6] M. Topkara, C. M. Taskiran, and E. J. Delp, "Natural language watermarking," in Proceedings of the SPIE Electronic Imaging: Security, Steganography, and Watermarking of Multimedia Contents VII, 2005, vol. 5681, 2005, doi: 10.1117/12.593790.

[7] K. Bennett, "Linguistic steganography: Survey, analysis, and robustness concerns for hiding information in text," 2004, cERIAS Tech Report 2004-13.

[8] S. H. Low, N. F. Maxemchuk, J. T. Brassil, and L. O'Gorman, "Document marking and identification using both line and word shifting," in Proceedings of the 14^{th} Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '95) April 2-6, 1995, Boston, Massachusettes, Apr. 1995, pp. 853–860.

[9] J. T. Brassil, S. Low, N. F. Maxemchuk, and L. O'Gorman, "Electronic Marking and Identification Techniques to Discourage Document Copying," IEEE Journal on Selected Areas in Communications, vol. 13 (8), 1995, pp. 1495–1504.

[10] J. T. Brassil, S. Low, and N. F. Maxemchuk, "Copyright protection for the electronic distribution of text documents," Proceedings of the IEEE, vol. 87 (7), 1999, pp. 1181–1196.

[11] Y. Kim, K. Moon, and I. Oh, "A Text Watermarking Algorithm based on Word Classification and Interword Space Statistics," in Proceedings of the 7^{th} International Conference on Document Analysis and Recognition (ICDAR'03) August 3–6, 2003, Edinburgh, Scotland. IEEE Computer Society Washington, DC, USA, Aug. 2003, pp. 775–779.

[12] M. Shirali-Shahreza and S. Shirali-Shahreza, "A New Approach to Persian/Arabic Text Steganography," in Proceedings of the 5^{th} IEEE/ACIS international Conference on Computer and Information Science and 1st IEEE/ACIS July 2006, Honolulu, USA, Jul. 2006, pp. 310–315.

[13] M. Aabed, S. Awaideh, A.-R. Elshafei, and A. Gutub, "Arabic Diacritics Based Steganography," in Proceedings of the IEEE International Conference on Signal Processing and Communications (ICSPC 2007) November 24–27, 2007, Dubai, UAE, Nov. 2007, pp. 756–759.

[14] A. A. Gutub, L. M. Ghouti, Y. S. Elarian, S. M. Awaideh, and A. K. Alvi, "Utilizing Diacritic Marks for Arabic Text Steganography ," Kuwait Journal of Science & Engineering, vol. 37 (1), 2010, pp. 1–16, ISSN: 1024-8684.

[15] W. Bender, D. Gruhl, N. Morimoto, and A. Lu, "Techniques for data hiding," IBM Systems Journal, vol. 35, 1996, pp. 313–336.

[16] A. M. Alattar and O. M. Alattar, "Watermarking electronic text documents containing justified paragraphs and irregular line spacing," in Proceedings of the SPIE - Security, Steganography, and Watermarking of Multimedia Contents VI June, 2004, San Jose, California, USA. Society of Photo Optical, Jun. 2004, pp. 685–695.

[17] M. Kwan, "The SNOW Home Page," 2006, URL: http://www.darkside.com.au/snow/ [accessed: 2014-03-03].

[18] L. Y. Por and B. Delina, "Whitesteg: a new scheme in information hiding using text steganography," WSEAS Transaction on Computers, vol. 7, 2008, pp. 735–745.

[19] L. Y. Por, K. Wong, and K. O. Chee, "UniSpaCh: A text-based data hiding method using Unicode space characters," The Journal of Systems and Software, vol. 85, 2012, pp. 1075–1082.

[20] C. Chen, S. Z. Wang, and X. P. Zhang, "Information Hiding in Text Using Typesetting Tools with Stego-Encoding," in Proceedings of the First International Conference on Innovative Computing, Information and Control August 30 - September 1, 2006, Beijing, China, 2006, pp. 459–462.

[21] I.-C. Lin and P.-K. Hsu, "A Data Hiding Scheme on Word Documents using Multiple-base Notation System," in Proceedings of the 6^{th} International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP'10) October 15-17, 2010, Darmstadt, Germany, Oct. 2010, pp. 31–33.

[22] P. V. K. Borges and J. Mayer, "Document Watermarking via Character Luminance Modulation," in Proceedings of the IEEE International Conference of Acoustics, Speech and Signal Processing (ICASSP 2006) May 14–16, 2006, Toulouse, France, Jul. 2006, pp. II–317, ISBN: 1-4244-0469-X.

[23] A. K. Bhattacharjya and H. Ancin, "Data embedding in text for a copier system," in Proceedings of the IEEE International Conference on Image Processing (ICIP 99) October 24-28, 1999, Kobe, Japan, Oct. 1999, pp. 245–249.

[24] R. Villán et al., "Text Data-Hiding for Digital and Printed Documents: Theoretical and Practical Considerations," in Proceedings of the SPIE Electronic Imaging: Security, Steganography, and Watermarking of Multimedia Contents VIII, 2006, vol. 6072, 2006, doi: 10.1117/12.641957.

[25] A. Gutub and M. Fattani, "A Novel Arabic Text Steganography Method Using Letter Points and Extensions," in Proceedings of the WASET International Conference on Computer, Information and Systems Science and Engineering (ICCISSE), vol. 21 May, 2007, Vienna, Austria, May 2007, pp. 28–31.

[26] P. Wayner. Elsevier Inc., 2009, 3rd edition, ISBN: 978-0-12-374479-1.

[27] B. Murphy and C. Vogel, "The syntax of concealment: reliable methods for plain text information hiding," in Proceedings of the SPIE International Conference on Security, Steganography, and Watermarking of Multimedia Contents 2007, vol. 6505, 2007, doi: 10.1117/12.713357.

[28] Nakagawa, H. and Matsumoto, T. and Murase, I., "Information Hiding for Text by Paraphrasing," 2002, URL: http://www.r.dl.itc.u-tokyo.ac.jp/ nakagawa/academic-res/finpri02.pdf [accessed: 2014-03-03].

[29] M. Shirali-Shahreza, "Text Steganography by Changing Words Spelling," in Proceedings of the 10^{th} International Conference on Advanced Communication Technology (ICACT 2008) February, 2008, Kitakyushu, Japan, vol. 3, Feb. 2008, pp. 1912–1913.

[30] M. Niimi, S. Minewaki, H. Noda, and E. Kawaguchi, "A Framework for a Simple Sentence Paraphrase Using Concept Hiearchy in SD-Form Semantics Model," in Proceedings of the 13th European-Japanese Conference on Information Modelling and Knowledge Bases (EJC 2003), June 3-6, Kitakyushu, Japan. IOS Press, 2004, pp. 55–66.

[31] M. Topkara, U. Taskiran, and M. J. Atallah, "Information Hiding Through Errors: A Confusing Approach," in Proceedings of the SPIE Electronic Imaging: Security, Steganography, and Watermarking of Multimedia Contents 2007, vol. 6505, 2007, doi: 10.1117/12.706980.

[32] M. Khairullah, "A Novel Text Steganography System Using Font Color of the Invisible Characters in Microsoft Word Documents," in Proceedings of the Second International Conference on Computer and Electrical Engineering (ICCEE '09) December, 2009, Dubai, Dec. 2009, pp. 482–484, ISBN: 978-0-7695-3925-6.

[33] W. Bhaya, A. M. Rahma, and D. Al-Nasrawi, "Text Steganography

 ISBN: 978-1-61208-376-6

based on Font Type in MS-Word Documents," Journal of Computer Science, vol. 9 (7), 2013, pp. 898–904, ISSN: 1549-3636.

[34] 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 (1), 2007, pp. 24–30.

[35] A. Castiglione, B. D'Alessio, A. De Santis, and F. Palmieri, "New steganographic techniques for the OOXML file format," in Proceedings of the IFIP WG 8.4/8.9 international cross domain conference on Availability, reliability and security for business, enterprise and health information systems August 22-26, 2011, Vienna, Austria. Springer, Aug. 2011, pp. 344–358, Tjoa, A. M., Quirchmayr, G., You, I., Xu, L. Eds., LNCS: 6908, ISBN: 978-3-642-23299-2.

 ISBN: 978-1-61208-376-6