ENGINEERING TRANSACTIONS • *Engng. Trans.* • **58**, 1–2, 75–89, 2010
Polish Academy of Sciences • Institute of Fundamental Technological Research

SENDING HIDDEN DATA THROUGH WWW PAGES: DETECTION AND PREVENTION

L. Polak [1)], Z. Kotulski [2)]

[1)] **Warsaw University of Technology**
Institute of Telecommunications
Nowowiejska 15/19, 00-665 Warszawa, Poland

[2)] **Institute of Fundamental Technological Research**
Polish Academy of Sciences
Pawińskiego 5B, 02-106 Warszawa, Poland

In the paper we propose a method of detecting and removing a hidden content which could be sent through the HTML code of WWW pages. We start from the overview of known methods of inserting hidden data to the mark-up languages (HTML or XML)-written texts. Next we propose a method of detecting suspicious web pages, based on statistical analysis. We also propose the method of transforming the HTML code of the web pages to remove any hidden communication.

Key words: steganography, steganalysis, HTML, internet security, data hiding.

1. Introduction

In contemporary world it is hard to imagine life without the Internet. Over the last two decades, Internet technology had really a big impact on our common activity, allowing to find remotely every information one is about to look for. Invention of the HTML, which is a language easy to learn, allowed people to create their personal web pages and contributed to the Internet success. Thus, everyone can create his or her own page and publish it in the Internet making it globally accessible. At present there are tens of billions of such web pages and their number is growing rapidly. Obviously the Internet huge size, combined with lack of an effective control, gives one an opportunity to "smuggle" some undesirable content into the ordinary web pages. Furthermore, a number of methods exist that allow to hide such a content or hidden message without changing the web page look. They are taking advantage of steganography. Steganography consists in hiding a secret data in a covert channel which can be, e.g., a graphic file, a web page or a network protocol, by exploiting some of its redundant properties, so the unaware recipients of the covert message do not notice any difference. It can be achieved, e.g., by changing the value of insignificant bits in a file or optional flags in a network protocol.

Steganography is not always used for a good purpose. Last years steganography has got great public interest as a suspicion arose that some terrorists might have used steganography to send secret communicates. Steganography is hard to detect and in the Internet it enables application of many various techniques. One of the possibilities, which we are studying in this paper, is hiding the messages in the mark-up languages, e.g., HTML web pages source code or XML documents.

Considering the difficulties of detecting steganographic communication and possible threats it can bring, we believe that there is a need for methods of detecting suspicious web pages and deleting hidden messages from them. In this paper we present the method that allows to achieve these goals.

2. Hiding Information in Mark-up Languages

Considering methods of hiding information in XML/HTML documents, two main groups can be identified. The first group comprises techniques originating from the classical text steganography while the second group includes methods which make use of mark-up languages specific properties.

The former group methods treat XML/HTML documents as text files and consist in embedding secret in a file by changing its content in a particular way, depending on information one wants to carry. The result can be achieved, e.g., by inserting additional white-spaces, making deliberate spelling errors or changing a font. Many different methods of text steganography exist and they are described in details in literature, e.g., [1–5]. Although they can be successfully applied to XML/HTML documents, they will not be analysed further in this paper because they are easy to detect. Our intention is to analyse schemes of sending secret message, which do not affect the visible content of a web page presented by a web browser.

It is an advantage of the techniques from the latter group, which exploit the ordered structure of the mark-up languages. Employing such techniques prevents the viewers from being alerted by unexpected changes of the content; in this regard, the visible web screen is entirely equivalent to the original document. The allowable modifications are constrained by the particular languages' specifications, see e.g. [6–8].

A number of approaches to HTML/XML steganography has been proposed. Now they will be briefly presented including their advantages and drawbacks. One of the most important terms which allow comparison of different steganographic methods is a steganographic capacity which describes the amount of information, which can be hidden in a given covert channel. It is typically expressed as the maximum size of a secret message in bits. Obviously there is usually a trade-off between the capacity and the security, i.e. the bigger capacity a certain method offers, the easier it is to detect.

2.1. Hiding Information in HTML documents

Each HTML document contains tags which, in turn, may contain attributes. Every tag has its own name. There is a limited number of allowed tags listed in the HTML specification [6, 9]. The tags are the tools used to describe how a web page should be presented. We can enumerate a number of methods of sending secret messages with the HTML texts. They are described in many papers regarding steganography, see e.g., in [1, 10, 11].

White-spaces in tags. In HTML we can add a white space before "$>$" sign, which marks the end of a tag. This way it is possible to send one bit of a secret message for each tag, by selecting the tags to which we want add an additional space before its end sign. Another possibility is inserting white-spaces after the tag (if there is no body text after it). Main drawback of this approach is that in consequence of its application to files, they are getting bigger, so the stego channel can be easily revealed.

Changing the case of letters in tags. HTML tags are case insensitive, hence we can take advantage of it to hide a message within a document by changing the case of specific letters in a tag's name. For example, <ID>, <id>, <Id> and <ID> mean exactly the same and we can encode two bits by choosing one of its version. Big capacity is the main advantage of this method. On the other hand, it is very easy to discover the stego channel since it is very unusual to use small and capital letters alternately.

Using default values of attributes. Some of the HTML attributes have their default values defined. An HTML document is treated by an HTML viewer the same way whether the default values have been explicitly defined or not. It gives an opportunity to hide an additional information by specifying default values in some parts of the HTML document, and skipping them in other parts. This method is hard to discover, but the limited possible number of attributes having default values is its main drawback.

Attribute order permutation. The HTML standard does not define a preferred order of attributes, what means that any order can be used without affecting a web page appearance. Since the order of attributes has no mean, this method can be applied without any restrictions. Changing the order of attributes for hiding information within an HTML document is the most interesting HTML steganography method. It does not change the original file size and it is hard to detect without computer programs analysing the HTML document's structure. Having a tag comprising 8 attributes, there are $8! = 40320$ different permutations, what lets one to hide over 15 bits of information within this single tag. This

method is probably the most often mentioned one in the context of hiding data in the mark-up language documents. Its main advantage lies in its security, but in practice it allows only to send a small amount of data, because it is limited by the number of attributes being used in the original document.

A free tool which implements this method exists: DEOGOL, which was written by Stephen FORREST in Perl in 2003 [12]. We used it to examine the efficiency of this method by calculating the capacity of WWW pages, to determine how much information can be secretly sent using the method described above. Our test set of web pages has been divided into four groups according to how the pages are administrated: pages with a content uploaded by their users, company pages, news portals and thematic pages dedicated to a specific topic. The results we obtained are presented in the Tables 1–4.

Table 1. Steganographic capacity of WWW pages (user content pages).

Page	Capacity (B)	Size (kB)	Efficiency (B/kB)
youtube.com	255	123	2.07
wrzuta.pl	118	52	2.27
joemonster.org	192	95	2.02

Table 2. Steganographic capacity of WWW pages (company pages).

Page	Capacity (B)	Size (kB)	Efficiency (B/kB)
microsoft.com	126	76	1.66
sony.com	100	7	14.29
cisco.com	53	24	2.21

Table 3. Steganographic Capacity of WWW Pages (news portals).

Page	Capacity (B)	Size (kB)	Efficiency (B/kB)
yahoo.com	133	208	0.64
cnn.com	153	96	1.59
onet.pl	90	87	1.03

Table 4. WWW Pages Steganographic Capacity (thematic pages).

Page	Capacity (B)	Size (kB)	Efficiency (B/kB)
fallout3.net	84	13	6.46
elka.pw.edu.pl	130	42	3.10
nba.com	104	79	1.32

Analysing the results we can notice that pages with user-generated content usually have the biggest steganographic capacity. These pages contain a lot of elements (especially hyperlinks) which have to be described by the HTML tags with attributes, what increases their capacity (Table 1). Table 2 shows that company pages usually have smaller capacity because their main pages are usually small ones in order to simplify navigation and finding specific information sought by the companies' customers. News portals usually do not allow to hide a lot of information comparing to their size, because the attributes do not appear so frequently in their source code. However, they are usually the big volume pages so they can constitute a decent stego-channel (Table 3). Thematic pages can vary significantly in terms of their capacity and size, therefore it is hard to extract their common features (Table 4).

Another thing to mention is that web pages created/edited in Microsoft Word have significantly bigger capacity than clear HTML pages. It is caused by an overhead added by the program: the redundant, Word-specific attributes are inserted into the document in HTML source code to format its view. After applying the Word-processing we got 462 bytes (3696 bits) of capacity for the cnn.com page what, comparing to 153 bytes before, gave us over 200% increase. For gazeta.pl we obtained 240% gain (442 B versus initial 130 B), and for fallout3.net 234% (281 B, versus 84 B before processing). These results show that this kind of processing allows to considerably increase the maximum size of hidden messages that can be sent over a HTML source code with the attribute's permutation algorithm.

Other methods. Except for the already mentioned methods, some other were proposed over the years. One of them was hiding coded data within the ID attribute of HTML tags proposed by Mohammad Shahreza [13]. This method allows to hide big quantity of information relatively to the previously discussed methods, but its usage can attract someone's attention, since ID attributes do not occur very often and even then they are usually in a meaningful and descriptive form.

2.2. Hiding information in XML

All the methods described in the previous subsection can be successfully applied to hide data in XML documents (except for method which involves changing case of letters because XML language is case-sensitive). Moreover, there are few additional techniques that can be used with XML documents, see [8].

Empty element representation. Each tag in XML has to have a closing tag. Two equivalent representations of an empty element exist what gives us an opportunity to use them to hide information.

Elements order. Next possibility is to change elements order. Obviously, it is only possible in the case when applications using XML documents do not require particular elements' order. This method resembles attributes' permutation method described before. It has similar advantages, however it is a bit easier to analyse and potentially detect, because the number and the type of attributes in a certain tag can vary significantly.

Web pages can also be saved in the XML compliant format called XHTML. Of course, all methods described above still apply. We analysed the differences in capacity between documents stored in these two different formats but they turned out to be insignificant.

3. Page structure's changes detection

Hiding information inevitably involves leaving some trace. When one embeds additional data within a document, the changes are impossible to avoid. An analysis of documents allows to detect those changes by finding particular unusual patterns and to decide upon this basis if a certain document may contain a hidden data.

As it was already mentioned, there are many possibilities of hiding information in WWW pages' HTML code. Taking into account the vast number of pages in the Internet, it is a tough task to scan all of them and find the suspicious ones. Moreover, there are almost no studies conducted in this area. We failed to find any information about effective implementations of the mark-up languages steganalysis, which would allow to examine the source code of the web pages automatically.

Good steganographic methods should neither affect visible content which is presented to a viewer, nor change properties which are very easy to check, e.g., the size of the document. The attributes permutation method meets both these requirements. It can be applied to the mark-up language documents since there is no attribute order defined and it can vary a lot across different documents. However, documents created by the same person, in a short time period should not differ too much, since web designers usually have their own style worked out through years of work, and they apply it consequently, what makes the web page design process significantly faster thanks to the improved code readability. Especially, considering a case of a single document created by one person, its consistency assumptions should be fulfilled.

On the other hand, having to do with a document with a variable inner structure (regarding the attributes order), one can state a thesis that the document has been deliberately changed by someone. One of purposes of such a change can be hiding secret data within the web page.

Next in this paper we propose the method of analysing document's structure, which allows to track changes of the structure and possibly to discover attempts of information hiding. This is not a trivial task because the attributes can occur in many different combinations in each tag, hence every possible combination have to be considered. This implies a necessity of an algorithm that would determine the expected order of attributes for each tag, such that it would be possible to compare the order of attributes in their subsequent occurrences of the tag with it.

4. Description of the method proposed

The method we would like to introduce consists in comparing all couples of attributes in each tag that occurred in a particular document. For each couple it has to be counted how many times the first attribute occurred before the second one and how many times the opposite situation happened. Then we check which case has occurred more frequently (hence it is more probable); this one is assumed to be the correct one. Afterwards we take the first couple of attributes and look for all its occurrences within a document, to check how many times the order is correct (according to the predominant order determined in the previous step) and how many times it is not. Then we repeat the same procedure with all other couples.

Fig. 1. Example of different attributes order in a tag.

In Fig. 1 we present a simple example of a tag containing 3 different attributes. We can observe the following dependences between attributes in it:

- *at1* occurs before *at2* 3 times. *at2* had not occurred before *at1*. Therefore the correct order for this couple of attributes is *<at1,at2>*,
- *at2* occurs before *at3* 2 times. *at3* before *at2* – 1 time. The correct order: *<at2,at3>*,
- *at1* occurs before *at3* 3 times. *at3* before *at1* – 0. The correct order: *<at1,at3>*,
- *at1*, *at2*, *at3* occurs before *at4* – 3 times. *at4* never occurs before any of them. The correct orders are: *<at1,at4>*, *<at2,at4>* and *<at3,at4>*.

This way we can examine the whole document and check how many unordered couples of the attributes exist. In the next subsection we present a formal description of this algorithm.

4.1. The procedure of anomalies detection

Let S be an HTML code containing a set of tags $\{T_1, T_2, T_3 \ldots\}$. Each tag occurrence has a set of attributes denoted by $T_{n,i}\{a_1, a_2, a_3 \ldots\}$, where n is the tag identifier and i defines the specific tag's occurrence. For the tag T_n containing attributes a_1 and a_2, let us denote that a_1 precede a_2 in a particular tag's occurrence by $T_{n,i}(a_1, a_2) = (a_1, a_2)$ which means they consist of an ordered pair of attributes. In that case $P_{n,i}(a_x, a_y) = 1$, see (4.1). Furthermore, $O_n(a_x, a_y)$ describes the more frequent order of the couples of attributes in a particular tag, see (4.2).

$$P_{n,i}(a_x, a_y) = \begin{cases} 1 & \text{if } T_{n,i}(a_x, a_y) = (a_x, a_y), \\ 0 & \text{otherwise;} \end{cases} \tag{4.1}$$

$$O_n(a_x, a_y) = \begin{cases} (a_x, a_y) & \text{if } \sum_i P_{n,i}(a_x, a_y) \geq \sum_i P_{n,i}(a_y, a_x), \\ (a_y, a_x) & \text{otherwise.} \end{cases} \tag{4.2}$$

The value of $R_{n,i}(a_x, a_y)$ in Eq. (4.3) describes, if for a specific tag occurrence, the order of attributes is compliant with the previously determined predominant order.

$$R_{n,i}(a_x, a_y) = \begin{cases} 1 & \text{if } T_{n,i}(a_x, a_y) = O_n(a_x, a_y), \\ 0 & \text{otherwise.} \end{cases} \tag{4.3}$$

The sum of $R_{n,i}$ for all pairs of attributes determines the number of their occurrences in the predominant order. Total number of all occurrences of all couples of attributes in S is denoted by C. Hence, we can calculate W, which expresses the fraction of couples of attributes which occurred in a given S in a different order than the predominant one: O_n, see (4.4).

$$\forall x \forall y \quad x < y : W = 1 - \frac{\sum_n \sum_i R_{n,i}(a_x, a_y)}{C}. \tag{4.4}$$

Basing on that statistics, we can predict what is a chance that the document has been altered by someone. If W is high, we can suspect that someone changed the code of the page or that the page was created by more than one person. In the next section we will present results of tests that we performed to evaluate the efficiency of the method proposed.

5. Examples of application

To test the proposed algorithm we chose a set of web pages falling into different categories and we hide text messages of different lengths within them, so we can check how efficiently the steganalytic algorithm works.

Among the pages the following groups were distinguished:

- *pages gathering users uploaded content*, e.g., youtube.com,
- *company pages*, e.g., microsoft.com,
- *news portals*, e.g., yahoo.com, onet.pl, cnn.com,
- *thematic sites focused on a specific topic: games, music, etc.*, e.g., fallout3.net.

5.1. Test 1: sensitivity of the method

To perform the tests we generated 7 text messages (from 5 to 151 characters). At first we inserted the messages into the downloaded pages using DEOGOL (the attribute permutation algorithm). Because of different capacities of the web pages, in some cases it was impossible to embed the longest messages.

Afterwards we examined how the W value depends on the message length. The obtained results are presented in Fig. 2. The plot presents the unordered attributes couples percentage W versus the relative steganographic occupation of each site (the ratio of the length of a hidden message and the page steganographic capacity).

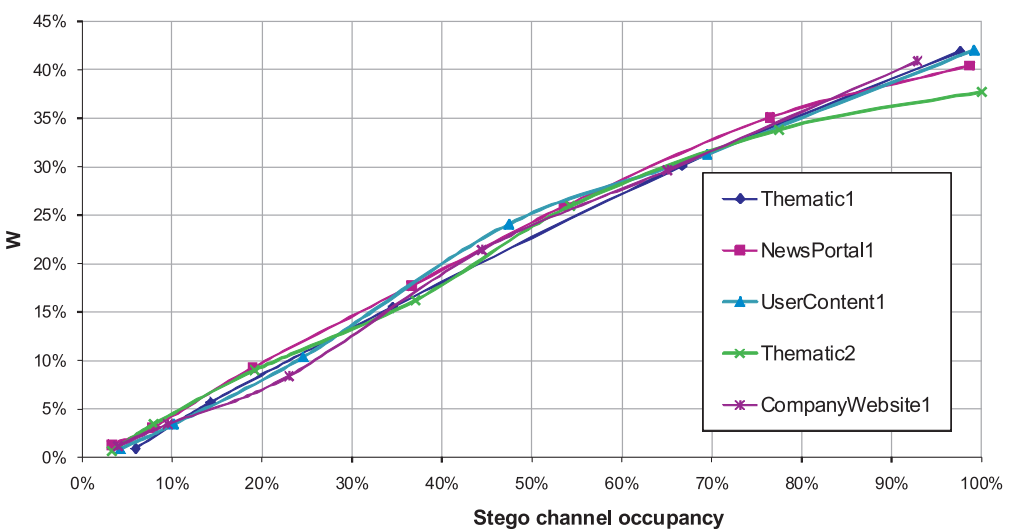

Fig. 2. Dependence of W on the stego channel occupancy for different WWW pages.

As we could expect, the increased length of the message causes the growth of the unordered couples number. For maximum occupancy, for the examined pages we have got results between 38% and 42%, whereas theoretical maximum is 50%, which in fact in the described case is a bit lower, since the pairs which are unique in a certain document have not been excluded (they are always considered as ordered). For the pages artificially modified, with the completely randomized order of attributes, we obtained very similar results; it means that they are equivalent to pages containing the maximum amount of secret data hidden inside.

For all pages under analysis the obtained results were nearly the same. That means that they were practically independent of a chosen site. We can notice a linear growth of W for the linearly growing length of a hidden message.

In Fig. 3 we compare a steganographically processed web page (in a way described in the previous paragraph) with pages that have not been converted. For readability, in Fig. 3 unprocessed web pages are marked by dashed lines. In fact there should be isolated points instead, as their stego channel occupancy is constant.

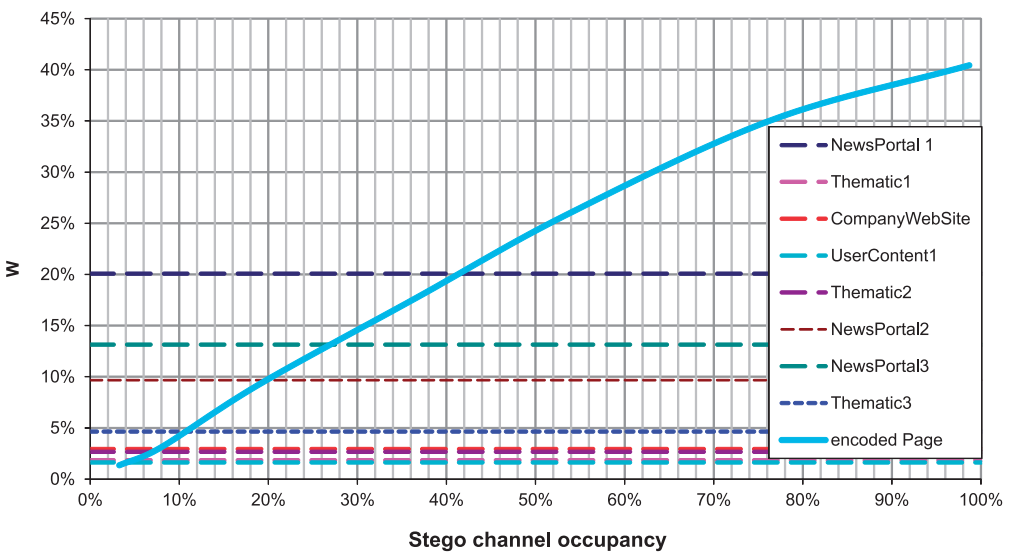

FIG. 3. Comparison of a steganographically modified page with original web pages.

We can see that the most of pages have little values of W ($< 5\%$), there is a smaller group with bigger values (10–14%) and one page with the result of 20% (a big news portal). We can notice that for the news portals we are getting bigger results than for other web pages. It can be explained as follows: usually these pages are created by many people who in turn may represent different styles

what may lead to higher results for W, since predominant order in one part is not necessarily such as in the parts created by different people. Such documents are not compliant with our assumption of the page coherence. The problem is to distinguish such pages from these really modified ones.

On the other hand, for other types of pages, we are able to obtain a good accuracy of the steganalysis and we can quite precisely distinguish unmodified pages from the ones containing hidden data.

In Fig. 4 we present exemplary decision thresholds. The higher one is set at 21%, hence it is bigger than any result received during the tests. For such a threshold we are able to detect a message occupying a bit more than 40% of the stego channel space. For the second one, which is 15%, we can find shorter messages, having the length of 30% of the web page stego channel capacity. The choice of the threshold is a trade-off between the precision and the recall. Having a high threshold, we can achieve good precision but we can omit some pages containing a steganographic data. On the contrary, for a lower threshold we can detect virtually all pages carrying secret messages (good recall), but some of the pages evaluated as suspicious ones may be classified wrongly.

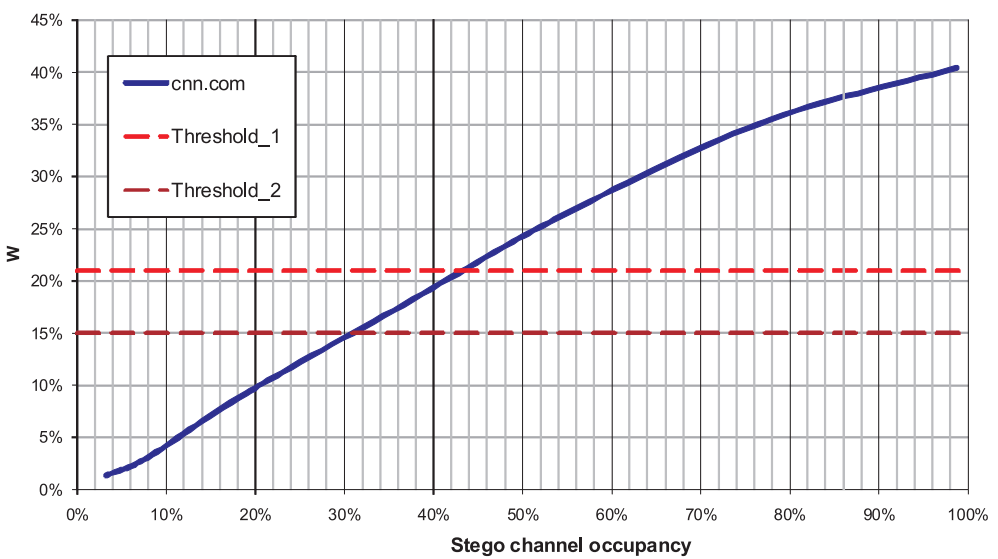

FIG. 4. A steganographically modified page with an exemplary decision threshold.

5.2. Test 2: long-term observations of web pages

Another kind of test that is carried out was a long-term test. Such a test is meant to check how the proposed measure W is changing over a long time

period. We assumed that hidden messages may not be transmitted constantly, i.e., there are some quiet periods when the steganographic channel is empty and the number of unordered couples of attributes is considerably lower.

The test was performed for four pages: two of them were news portals (Figs. 5–6), one was a page with user-uploaded content (Fig. 7) and one was a thematic page (Fig. 8). They were analysed every hour for almost one month (from 6.05.2009 to 2.06.2009). The irregular data in Fig. 6 for probes 216–234 should be considered as measurement errors caused by errors during the page download process and they will not be taken into account during our analysis.

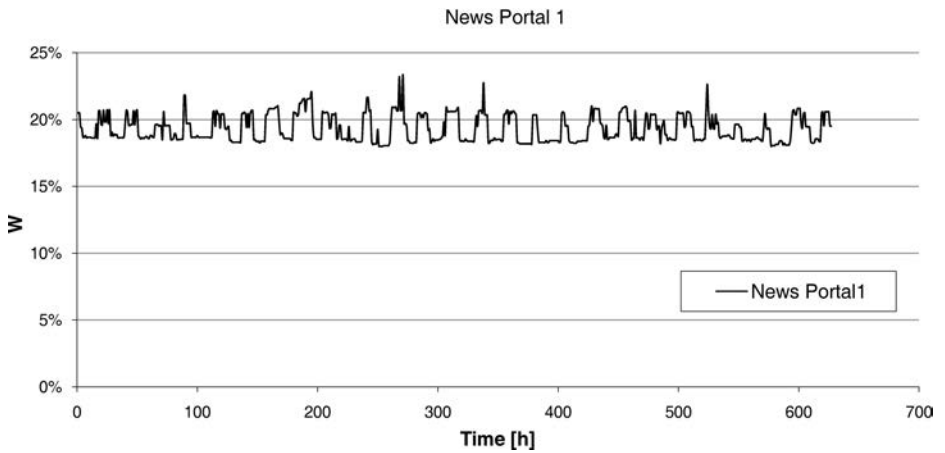

FIG. 5. Long-term test for news portal 1.

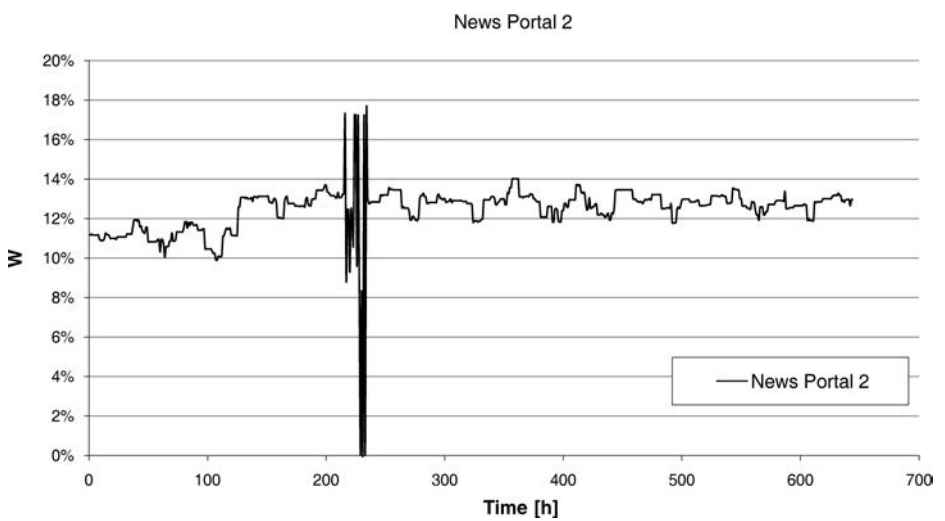

FIG. 6. Long-term test for news portal 2.

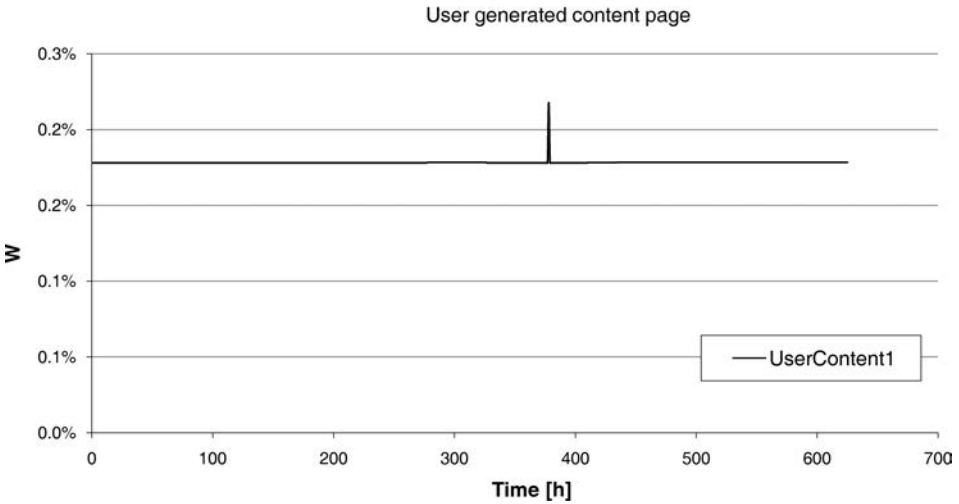

FIG. 7. Long-term test for a web page with user generated content.

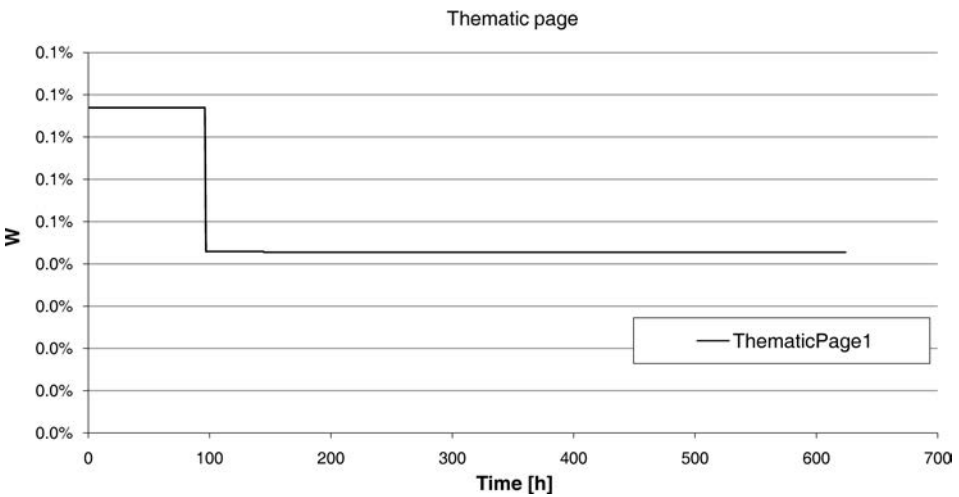

FIG. 8. Long-term test for a thematic web page.

In the charts we can see how the value of W has been changing over the examined period of time. For the last two pages there was only a single small change (about 0.5%). Completely different results were obtained for informational portals: the examined ratio W changes much more (up to 4–5%). Analysing those results it is easy to see a difference between large news pages which are frequently updated and the pages having a fixed structure, which are changing rarely. The news web pages, created by many people, are characterized by a higher level of unordered couples ratio W and its higher dynamics.

The proposed algorithm can be used to monitor changes in a web page structure. Because of the algorithm is the attributes' order-sensitive, it allows to detect changes only in a page structure, not in its content. Thus, we come to the conclusion that a long-term tests would allow to distinguish big informational pages from the pages really modified. We deduce that pages containing stego messages should have a constant, high value of the ratio W (when the message is transmitted all the time) or conversely, it should fluctuate heavily (when the message is transmitted only in certain periods of time). On the other hand, normal changes should look like on the charts presented above: regular and periodical. However, it is very hard to verify this hypothesis, because obviously there is no information available regarding pages exploiting this kind of steganography.

Another thing to mention is that since we have calculated the expected order of the attributes O_n for a given document, we can easily change the order of all attributes in the document in this way that they are compliant with O_n. Hence, we are able to 'clean' it without changing visual appearance of the web page and to be sure that no additional information is sent within the page. Since it does not change the visual content, it can be done always, even if we are not sure whether the page was modified. Similarly, other methods can also be applied to remove suspicious features that are not visible in a web browsers and can be used for sending hidden messages, e.g., white-spaces at the end of line.

6. Conclusion

In this paper we proposed the procedure which allows to monitor the changes of web pages structure. We introduced the measure which describes how the attributes of tags are ordered in HTML/XML documents. It allows us to discover potential secret communication which uses the attributes' permutation method. Our experiments show that in some cases it might be hard to determine if a page was really changed or if it was created by several different people and therefore it might be inconsistent. Long-term tests can be used to distinguish both kinds of pages and to trace the changes in a web page inner structure.

Furthermore, we are always able to protect web pages from the steganographic techniques involving attributes' permutation by setting all the attributes in a fixed order. Our method can be applied to control web pages and to assure that nobody could exploit them as stego channels.

References

1. A. Brainos, *A Study of Steganography and the Art of Hiding Information*, Security Writer, 2004.
2. J. R. Krenn, *Steganography and Steganalysis* (January 2004), http://www.krenn.nl/univ/cry/steg/article.pdf (last visited: 04.2010).

3. F. A. P. Petitcolas, R. J. Anderson, and M. G. Kuhn, *Information Hiding. A Survey*, Proceedings of the IEEE, **87**, 7, 1062–1078, July 1999.

4. T. G. Handel and M. T. Sandford II, *Hiding Data in the OSI Network Model, Information Hiding*, LNCS 1174, pp. 23–28, Springer Berlin, 1996.

5. N. F. Johnson and S. Katzenbeisser, *A Survey of Steganographic Techniques, Information Hiding: Techniques for Steganography and Digital Watermarking*, pp. 43–75, Artech House, 1999.

6. HTML standard specification, http://www.w3.org/TR/html4/ (last visited: 04.2010).

7. XML standard specification, http://www.w3.org/TR/REC-xml/ (last visited: 04.2010).

8. A. G. Memon, S. Khawaja, and A. Shah, *Steganography: A New Horizon for Safe Communication through XML*, Pakistan Journal of Theoretical and Applied Information Technology, **4**, 3, 187–202, March 2008.

9. D. Goodman, *Dynamic HTML: The Definitive Reference*, Second Edition, OReilly & Associates, Inc., 2002.

10. S. Inoue, K. Makino, I. Murase, O. Takizawa, T. Matsumoto, and H. Nakagawa, *A Proposal on Information Hiding Methods using XML* (2001), http://takizawa.ne.jp/nlp_xml.pdf (last visited: 04.2010).

11. J. Corinna, *Steganography 13 – Hiding binary data in HTML documents*, http://www.codeproject.com/csharp/steganodotnet.asp (last visited: 04.2010).

12. S. Forrest, Introduction to Deogol, http://wandership.ca/projects/deogol/ (last visited: 04.2010)

13. M. Shirali Shahreza, *New Method for Steganography in HTML Files, Advances in Computer, Information, and Systems Sciences, and Engineering*, Proceedings of IETA 2005, TeNe 2005 and EIAE 2005, pp. 247–252, Springer Netherlands, 2006.

Received October 14, 2010.