Techniques with Statistics for Web Page Watermarking

Mohamed Lahcen BenSaad, and Sun XingMing

***Abstract*—**Information hiding, especially watermarking is a promising technique for the protection of intellectual property rights. This technology is mainly advanced for multimedia but the same has not been done for text. Web pages, like other documents, need a protection against piracy. In this paper, some techniques are proposed to show how to hide information in web pages using some features of the markup language used to describe these pages. Most of the techniques proposed here use the white space to hide information or some varieties of the language in representing elements. Experiments on a very small page and analysis of five thousands web pages show that these techniques have a wide bandwidth available for information hiding, and they might form a solid base to develop a robust algorithm for web page watermarking.

***Keywords*—** Digital Watermarking, Information Hiding, Markup Language, Text watermarking, Software Watermarking.

I. Introduction

If you find your own property with another person, and you claim that is yours, no one can believe you, unless you give proof. Watermarking is a technique used to prove intellectual property rights over certain content by hiding some information in it, the information that is hidden in a content is referred to as the watermark [1] and can contain information such as the name of the owner, company, and timestamp. Later, when the ownership is under dispute, it will be easy to prove the legitimate owner of the document by showing the watermark.

Digital watermarking can prohibit the unauthorized use, duplication and distribution of digital documents, but it must not change the value of the content to keep it useful for the intended purpose. The watermark must be robust to resist any attack that tries to remove it or destroy it without changing the value of the content [2].

Web page designers spend days designing a page, but unfortunately, when this page is hosted in an internet server, it can be easily stolen in one moment by one click. A whole site can be downloaded by tools like Teleport and Webzip and its content can be used in an unauthorized way and it can be distributed without referring to its owner. Watermarking can help in such situation, but until now, there are no good algorithms developed for web page watermarking.

Some techniques of information hiding using XML (eXtensible Markup Language) are proposed [3], but they are very few and they don't exploit very well the features of the language. This paper gives more other techniques that use the white space and line break character in many places, and tries to exploit efficiently the features of the markup language and its specifications to come with new techniques in order to make the bandwidth as large as possible. These techniques are based on changing the source code of the web page without changing its appearance when it is displayed. We shall see in details these techniques in section IV, but before we have to understand HTML, which is described in section II, and some details on information hiding and watermarking, which are described in section III. Section V will describe some experiments and some statistics.

II. HTML and Web Pages

HTML (Hyper-Text Markup Language) is the language that Web pages are written in. Its syntax is based on a list of elements that describe the page's format and what is displayed on the Web page. Web pages are what make up The World Wide Web, they must conform to the rules of HTML in order to be displayed correctly in a Web browser. HTML is written in SGML (Standard Generalized Markup Language), a formal system designed for building text markup languages [4].

Web pages can be either static or dynamic; Static pages show the same content each time they are viewed while dynamic pages have content that can change each time they are accessed. These pages are typically written in scripting languages such as PHP, Perl, ASP, or JSP. The scripts in the pages run functions on the server that return information as HTML code, so when the page gets to the browser, all the browser has to do is translate the HTML code [5]. It is obvious that in both cases the web browser receives the document in HTML format.

HTML includes element types that represent paragraphs, hypertext links, lists, tables, images, etc. Each element type declaration generally describes three parts: a start tag, an end tag, and a content that appears between these two tags. The element's name appears in the start tag (written <element-name>) and the end tag (written </element-name>). Some element types allow authors to omit end tags and few of them also allow the start tags to be omitted. Some element types have no content. Elements may have associated properties, called attributes, which may have values (by default, or set by authors or scripts). Attribute/value pairs appear before the final ">" of an element's start tag in any order [6].

III. Information Hiding

Information hiding is a technology that uses a cover data to

Manuscript received May 20, 2005. This work was supported by the National Science Foundation of China (NSFC No.60373062), Hunan Provincial National Science Foundation of China (HPNSFC No.02JJYB012). Key Foundation of Science and Technology of Minister of Education of China (No.03092).

Mohamed Lahcen BenSaad and Sun XingMing are with the Department of Computer Science, Hunan university 410082, China (e-mails: bml13@hotmail.com and sunnudt@123.com respectively).

hide secret information in it. This hidden information can be extracted later when it is needed. Fig. 1 shows the general model of information hiding [3]. For different kinds of data, we need different methods of information hiding. For example, for images, we can use the least significant bits of pixels in this image to hide information, these changes are difficult for the naked eye to see them. For text, changing the location of the punctuations, the choice between synonyms, or spacing between words are simple ways to add information without changing the value of the original text.

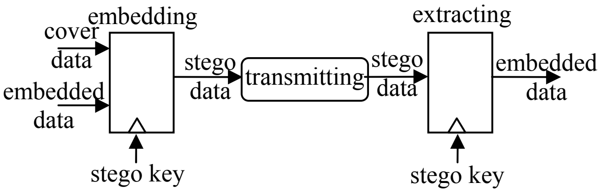

Fig. 1 General Model of Information Hiding

Watermarking as part of information hiding uses these model to embed or extract a watermark that can serve the protection of documents against piracy.

IV. Techniques for Web Page Watermarking

We explain here some techniques that can be used to embed or hide information in HTML files. The proposed techniques are verified to be compatible with HTML specification recommended by W3C [6]. Most of the cases have an example to illustrate how to hide bits.

A. *White Space*

We can use white space in many ways, by adding or removing it, we can embed data without changing the meaning of the original file.

1) Varieties of White Space Characters

The following characters are defined as white space characters [6]:

ASCII space (&#x0020;)
ASCII tab (&#x0009;)
ASCII form feed (&#x000C;)

We can hide some information using these varieties alternatively in representing white spaces.

2) Replacing White Space by Its Named Character “&nbsp;”

White space can be indicated by the named white space character “&nbsp;”, this alternative can be used to hide information as shown in the following example:

Use the code : <P>text text</P> to hide 0
And use : <P>text&nbsp;text</P> to hide 1

If the text is quite long, it will be difficult to see this changing even when the HTML source code is viewed.

3) White Space Between the Element’s Name and the First Attribute:

A first place where we can insert a white space in a start tag having attributes is directly after the element’s name and before the first attribute, here is an example of this case:

<font face=”verdana” size=”3”> 0
<font face=”verdana” size=”3”> 1

4) White Space Between Attributes

We can also insert spaces between the attributes of an element as shown in the following example:

<font face=”verdana” size=”3”> 0
<font face=”verdana” size=”3”> 1

5) White space between tags:

If there are some tags appear in the same line, we can add a space between any two of them as shown in the following example:

<td>colomn1</td><td>colomn2</td> 0
<td>colomn1</td> <td>colomn2</td> 1

6) White Space After “=”

The attributes of an element have values affected by the symbol “=”, if we insert a space after the “=” it will not change the rendering of the document, we can use this as shown in the following example:

<font size=”3”> 0 <font size= ”3”> 1

7) White Space Before New Line Character

If we have a new line character in the source code, we can choose whether to insert or not a white space before it to hide 1 or 0, as shown in the following example(new line character is unprintable character so we represent it by: nl):

<font size=”3”>nl 0 <font size=”3”> nl 1

B. *Line Break*

1) Varieties of Line Breaks

Single carriage return, single line feed and carriage return/line feed pairs are considered as a single line break[6]. We can exploit this feature by using these varieties alternatively in representing line break to hide some bits.

2) Adding Line Break After or Before Tags

A line break or more occurring immediately following a start tag must be ignored, as must a line break occurring immediately before an end tag. This applies to all HTML elements without exceptions [6]. This also can be used for information hiding. The following examples must be rendered identically:

i. <P>text</P>	ii. <P>text </P>	iii. <p> text</p>	iv. <p> text </p>

3) Line Break Between Two consecutive Tags

We can add one or more line break between any two tags without affecting the rendering of the document. The two following examples must be rendered in the same way:

i. <td>colomn1</td><td>colomn2</td> 0
ii. <td>colomn1</td>
<td>colomn2</td> 1

C. *Order of Attributes*

Elements may have associated properties, called attributes, which may appear in any order [6]. A technique has been proposed is to change the appearing order of the elements [3]. We propose here an ascending or descending order according to the ASCII representation of attributes' names or their values as shown in the following example:

i. Ordering according to attributes' names:
Ascending : <font face=”verdana” size=”3”> 0
Descending: <font size=”3” face=”verdana”> 1
ii. Ordering according to attributes' values:
Ascending : <font size=”3” face=”verdana”> 0
Descending: <font face=”verdana” size=”3”> 1

D. *The Default Value of an Attribute:*

Some attributes of some tags have default values [6], writing or omitting these attributes will not change anything, so if an attribute is omitted we can add it and set its value to the default value, here is an example:

<font> 0 <font size=”3”> 1

E. *Optional end tags:*

Some of the tags have optional end tags, it means that we can write the end tag or we can remove it, this can be used also to hide information as shown in the following example:

<td>colmn1</td><td>colomn2</td> 0
<td>colmn1<td>colomn2 1

F. *The string delimiters:*

Both double quotes and single quotes are accepted as string delimiters [6], so we can use the first to hide 0 and the second to hide 1 as follows:

<p align=”center”> 0
<p align=’center’> 1

G. *Changing Color Values*

1) Using Color Names

A color value may be either a hexadecimal number (prefixed by a hash mark) or one of the defined sixteen color names[6]. This difference of color representation can be used to hide information as follows:

<font color=”blue”> 0
<font color=”#0000FF”> 1

2) Altering Color Values

Altering color values has been proposed for information hiding in other contents, we can use it also with HTML. A color value represented as hexadecimal number is composed of three sub-values: Red, Green and Blue ("#RRGGBB"). We can hide one bit in each sub-value by altering its least significant bit, for example if we want to hide the three bits 110 in the value "#A560FF", we have to change it to "#A561FE".

V. Implementation and Experiments

Following the typology of web pages, we can say that web pages watermarking can be classified in two types; static watermarking (permanent) and dynamic watermarking (on-the-fly). For a static web page, we can watermark it once and save it, and when this page is being requested, it will be sent to the browser with the watermark. A dynamic web page is generated in run-time, so we need to catch it before sending it and watermark it on-the-fly then send it. In both cases, we can extract the watermark from the HTML document when it reaches the client [7], [8].

A. *Experiments*

We wrote a Java program to try some of the techniques proposed in this paper. We took, as example page to watermark, the main Google's English page. Although this page is very small (it is one of the smallest and simplest pages in internet), the result was great and the bandwidth was large enough for information hiding. We were able to hide a few times the word “Google” in that very small page, for example 67 bits could be hidden using techniques *A*.3 and *A*.4 (space in start tags), and 56 bits using technique *A*.5 (space between two tags).

B. *Statistics*

In order to have a wide vision about the structure and format of web pages source code, and to study the possibility to use these proposed techniques, we have analyzed 5000 web pages, that is more than 138 MB of HTML code. We downloaded these pages from internet, the links to these pages were the results of 10 first links from the result of every query of 500 Google queries using 500 different key words. To have different views, we divided at random the pages on 5 samples each one with 1000 pages. We wrote a Java program to scan these files, the result of all the samples were almost the same, which means that these samples are really representing the studied domain i.e. web pages. Table I gives some general information about the analyzed pages.

From Table I we see that start tags represent more than 56% of the analyzed HTML code, it is even larger than the content (text) itself, this feature gives more chance to the bandwidth to be larger, because most of the proposed techniques are applied on start tags. Table II gives us more details about the analyzed files, the results can be used to know the probability of some techniques to be applied and how large is the bandwidth.

From Table II we conclude the following:

1. Appearance of attributes: Is the number of the attributes that appear in a start tag divided by the number of attributes that can be applied on that tag. We see that only 6% of the attributes are set by the code, which means that the rest (94%) are set to their default values (if they have), this feature gives more chance to technique *D* to be used.
2. Number of attributes: On average, there are 595 attributes in one file. This gives the chance to techniques *A*.3 and *A*.4 to be used. The bandwidth here is 595 bits.
3. Number of tags having attributes: Statistics show that there is a good average of start tags that have attributes, it is

useful for techniques using attributes reordering and white space in start tags. The bandwidth here is 276.

4. Number of color values: Techniques in section *G* can be used here, the bandwidth is 3*28=84.
5. Number of the named space character “&nbsp”: This gives more chance to technique *A*.2 to be applied. We can hide 29 bits exploiting this feature.
6. Number of extra space in start tags: We can use this space and rearrange it as we like the start tag.
7. Number of new line characters: It gives chance to techniques *A*.7 and *B* to be used. 280 bits can be hidden here.
8. Number of omitted optional end tags: It is useful for applying technique E. We have 11 bits available here.
9. Number of tags appearing in the same line: It is useful to hide bits using techniques *A*.5 and *B*.3 The bandwidth here is 285.

TABLE I. GENERAL INFORMATION ABOUT THE ANALYZED FILES

Information	Samples (Si)					All together
	S1	S2	S3	S4	S5	
Number of files	1000	1000	1000	1000	1000	5000
Total files size (1000 kB)	28.14	26.48	27.28	29.96	26.35	138.20
Average size / file (KB)	28.14	26.48	27.28	29.96	26.35	27.64
Start tags size* (%)	59.51	56.65	54.67	57.73	56.09	56.93
End tags size* (%)	05.95	06.23	05.94	05.96	05.87	05.99
Contents (text) size** (%)	31.79	34.36	36.87	33.46	35.66	34.43
Remark tags size* (%)	02.75	02.76	02.53	02.84	02.38	02.65

* size = 2+number of characters between ‘<’ and ‘>’ that delimit a tag.
** size = number of characters between ‘>’ of a tag and ‘<’ of the next tag.

TABLE II. DETAILS ABOUT THE ANALYZED FILES. IT HELPS US TO SEE WHICH TECHNIQUE HAS MORE CHANCE TO BE USED, AND TO ASSIGN PRIORITIES TO TECHNIQUES IN USING THEM

Information		Samples (Si)					Average
		S1	S2	S3	S4	S5	
1	Appearance of attributes (%)	6.01	5.96	6.08	5.92	6.08	6.00
2	Number of attributes	624	577	579	650	544	595
3	Number of tags that have attributes	288	271	268	298	254	276
4	Number of color values	028	027	029	029	029	028
5	Number of named space char “&nbsp;”	026	026	033	030	032	029
6	Number of extra space in start tags	085	058	054	051	096	069
7	Number of new line character	288	278	276	304	255	280
8	Number of omitted optional end tag	012	009	010	011	011	011
9	Number of tags appear in the same line	289	280	282	311	264	285

Table II gives us a good statistic about the bandwidth available for some techniques, we can say that the total bandwidth available for all techniques is more than 1500 bits per file, it is big enough to hid a good amount of information.

VI. CONCLUSION

This paper proposes many techniques for information hiding using HTML, and it shows that these techniques have a good chance to be applied. The techniques can form a very good base to develop an algorithm for web page watermarking. A future works may focus more on a watermarking algorithm using the techniques presented here, and try to find other new techniques by digging deeply in HTML specifications to find more features to exploit. Moreover, the most important thing is to increase the robustness of the watermark, and try to adapt these techniques for using them with other markup languages like XML.

REFERENCES

[1] J. Nagra, C. Thomborson, and C. Collberg, “A functional taxonomy for software watermarking,” In M. J. Oudshoorn, Twenty-Fifth Australasian Computer Science Conference (ACSC2002), Melbourne, Australia, 2002. ACS.

[2] Radu Sion, Mikhail Atallah, and Sunil Prabhakar, “Rights protection for relational data,” SIGMOD Conference 2003: 98-109.

[3] Shingo Inoue, Kyoko Makino, Ichiro Murase, Osamu Takizawa, Tsutomu Matsumoto, and Hiroshi Nakagawa, “A proposal on information hiding methods using XML,” 1st Workshop on NLP and XML, Nov.2001.

[4] Rick Darnell, *HTML Unleashed*, Sams.net Publishing, August 1997. Available: http://www.webreference.com/dlab/books/html/. Chapter 3.

[5] The Sharpened.net Computer and Internet Glossary. Available: http://www.sharpened.net/glossary/, (last visit: May 20, 2005).

[6] W3C Recommendation, HTML Specification 4.01. Available: http://www.w3.org/TR/1999/REC-html401-19991224/, (May 20, 2005).

[7] D. Curran, N.J. Hurley, and M. O. Cinneide, “Securing Java through software watermarking,” In Proceedings of the 2nd international conference on Principles and practice of programming in Java, pages 311-324, 2003.

[8] Christian Collberg and Clark Thomborson, "Software watermarking: models and dynamic embeddings," In Proceedings of Symposium on Principles of Programming Languages, POPL'99, pages 311-324, 1999.