文章编号:1674-7070(2015)01-0040-06

付章杰[1,2]　孙星明[1,2]

基于冗余属性的 OOX 文本数字水印算法

摘要

针对流行最广的 Office Open XML 格式文档(即 MS Office 2007—2013),提出一种基于冗余属性的文本数字水印算法.利用 OOX(Office Open XML)文档包中部件的冗余信息来嵌入水印,并使水印与文档的格式信息绑定,可有效抵抗针对文本内容的攻击.该方法具有较强的鲁棒性和较大的嵌入容量,能够抵抗"另存为"、"清除格式"等攻击,且没有改变原始文档显示字符的任何格式信息,这是以往针对格式文档提出的水印方法做不到的,同时,也没有改变原始文档的任何语义信息,这也是以往的基于语义(自然语言)的水印方法做不到的.实验结果说明了该算法的可行性.

关键词

数字水印;Office Open XML;冗余属性;鲁棒性;隐蔽性

中图分类号 TP309

文献标志码 A

收稿日期 2014-01-24

资助项目 国家自然科学基金(61232016,61373133);公益性行业科研专项(GYHY201206033);江苏高校优势学科建设工程(PAPD)资助项目

作者简介

付章杰,男,博士,讲师,主要研究领域为网络与信息安全.wwwfzj@126.com

孙星明(通信作者),男,博士,教授,博士生导师,主要研究领域为网络与信息安全.sunnudt@163.com

1 南京信息工程大学 计算机与软件学院,南京,210044

2 南京信息工程大学 江苏省网络监控工程中心,南京,210044

0 引言

数字水印技术能够为数字载体(文本、图片、视频等)提供版权保护和泄密追踪,它可以不可感知地把作者版权信息嵌入到载体作品中,且对载体不造成任何影响[1].在产生版权纠纷时,通过相应的提取算法提取出该水印,可以达到版权保护、票证防伪、泄密追踪和隐蔽信息传输等目的[2-3].

一个理想的数字水印算法必须具有较强的鲁棒性、较好的隐蔽性和较大的嵌入容量,但这三者之间存在着天生的矛盾.实际应用中数字水印方案往往通过牺牲某一种或两种指标来提高其他指标,而不可能同时满足这三种性能指标.一般来说,基于文本内容的数字水印方法,其水印是与文本内容相关联的,当文本内容受到恶意攻击(删除或修改)时,其水印就会丢失;而当其格式信息受到攻击(清除格式或修改格式)时,其水印信息一般不会丢失.基于文本格式的数字水印方法则相反,其水印信息是与文本格式相关联的,当文本格式受到恶意攻击(清除格式或修改格式)时,其水印信息就会丢失;而当其内容信息受到攻击(删除或修改)时,其水印信息一般不会丢失.

文本文档的格式和文字信息中所包含的冗余空间非常有限,所以在文本文档中嵌入水印相对困难.Microsoft 的 Office 2007—2013 文档使用了一种新的格式标准 Office Open XML(OOX).这种新格式文档使用了 ZIP 压缩技术和 XML 技术相结合的方法,使得文档中可以利用的冗余空间更加有限.

本文在深入研究 OOX 文档格式特征的基础上,分别提出了 2 种数字水印方法,一是基于添加删除修改标识符的方法,二是基于添加修改标识符和文档变量的数字水印方法.这两种方法具有较强的鲁棒性和较大的嵌入容量,能够抵抗"另存为"、"清除格式"等攻击.

1 Office Open XML 格式

MS Office 2007—2013 采用了新的 OOX 文件格式,每一个新的 OOX 文件都由一系列压缩的部件组成,这些压缩部件存储在一个叫 package 的容器里,并且每一个压缩包都符合 OOX 文件格式标准[4-8].一个 MS Word 2007 文件实际是一个 ZIP 压缩包,该压缩包包含了组成文档的各个部件[5].打开一个 Word 2007 文档,并使用 Windows 资源管理器重命名该文档的扩展名".docx"为".zip"后,点击该 ZIP 文件

就会发现其中包含了许多不同的元素[6-7](包含图片、图、表等),如图1所示.

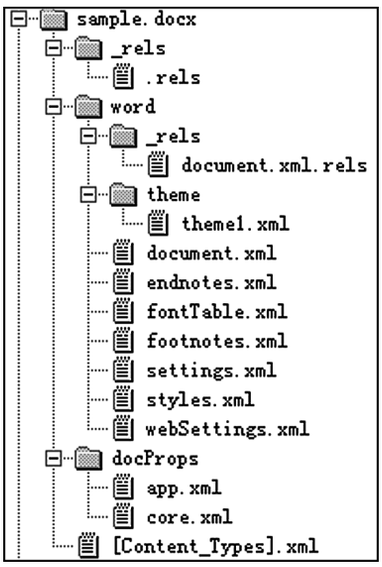

图1 一个MS Word 2007文件的目录结构

Fig. 1 The directory structure of an MS Word 2007 file

一个OOX格式文档由主文档部件和其他部件组成.主文档部件是document.xml部件[8],该部件包含了几乎所有应该显示在MS Office应用程序输出界面上的字符.

在setting部件中,rsid属性记录了主文档部件产生的所有修改标识符的属性值,每个属性值记录一次.同时在setting部件中还有一个文档变量属性docVar,该属性记录了文档中的变量名及其属性值.

2 水印嵌入方法

本文提出一种鲁棒性强的数字水印方法,该方法既可以在OOX文档包的主文档部件中实现,也可以在OOX文档包的setting部件中实现.

2.1 主文档部件

每一个电子文档都不可避免地被编辑很多次直到最终版本的形成.这些修改编辑行为包括删除、插入、格式修改等.针对OOX格式文档的每一次修改行为都会产生一定的痕迹,这些痕迹会被主文档等部件中的修改标识符(Revision Identifiers)及其属性值所记录.

如图2所示,w:rsidP,w:rsidR,w:rsidRPr,w:rsidRDefault和w:rsidDel就是修改标识符.这些修改标识符是由w:p元素或w:r元素所定义的,它们的属性值是由随机的8位16进制数组成的[9].其中w:rsidDel叫做删除修改标识符,一般不出现在主文档部件中.

```
<w:p  w:rsidP="008A5A97"  w:rsidR="008A5A97"
w:rsidRPr="008A5A97"
w:rsidRDefault="008A5A97">
  <w:pPr>
    <w:rPr>
      <w:szw:val="28" />
    </w:rPr>
  </w:pPr>
  <w:r                          w:rsidRPr="008A5A97"
w:rsidDel="00FFAABB">
      <w:rPr>
        <w:rFontsw:hAnsi="Times New Roman" />
        <w:color w:val="FF0000" />
      </w:rPr>
      <w:t>steganographic method for data hiding</w:t>
    </w:r>
  </w:p>
```

图2 主文档部件document.xml中的部分代码

Fig. 2 Partial code in document.xml

实验发现,删除修改标识符w:rsidDel可以被添加到主文档部件的w:r元素中,其属性值能用来隐藏编码后的水印信息,这种变化不会影响文档的使用,因此本文定义删除修改标识符w:rsidDel为冗余属性之一.一般来说,绝大多数修改标识符属性值的前两位都是以"00"开头的,最后6位是随机产生的.因此,水印信息经编码转换为16进制数后可以嵌入到删除修改标识符属性值的最后6位中去.每一个w:r元素可添加一个删除修改标识符,即可隐藏6位16进制数,即24位水印信息.

2.2 setting部件

实验发现,可以向setting部件里面添加任意数量的rsid属性和docVar属性及其属性值.水印信息编码后可伪装成rsid和docVar属性的属性值嵌入到setting部件中,如图3所示.

```
<w:rsid w:val="00087D05" />
<w:rsid w:val="000F094D" />

<w:docVar w:name="example1" w:val="value1" />
<w:docVar w:name="example2" w:val="value2" />
```

图3 setting部件中的水印嵌入方法

Fig. 3 Watermark embedding method in setting

3 算法描述

3.1 水印嵌入算法

1) 输入:原始载体文档D、原始水印信息M和

私钥 k.

2) 输出:隐写文档 S.

步骤 1:利用私钥 k 和非对称加密算法 RSA,对原始水印信息进行加密,得到加密后的水印信息 I':$I'=E(k,I)$,并将之转换为 16 进制编码 $H=H_1H_2\cdots H_i\cdots$;

步骤 2:计算 H 的 16 进制编码长度 $len(H)$,并将 $len(H)$ 转换为 16 进制数据附加在 H 前面得到 A';

步骤 3:利用 XML 文档解析技术,从原始载体文档 D 的 ZIP 包中读取主文档部件 document.xml 的所有内容给 C:$C=C_1C_2\cdots C_i\cdots$,读取 setting 部件的所有内容给 G:$G=G_1G_2\cdots G_i\cdots$;

步骤 4:从 C:$C=C_1C_2\cdots C_i\cdots$中提取一对 run 元素"〈w:r〉〈/w:r〉"给 R;

步骤 5:增加一个删除修改标识符及其属性值到 R 中;

步骤 6:从 $H=H_1H_2\cdots H_i\cdots$中读取 6 位 16 进制信息,替换刚增加的删除修改标识符属性值的后 6 位;

步骤 7:如果所有的 run 元素"〈w:r〉〈/w:r〉"都增加了删除修改标识符,而水印信息还没有嵌完,则向 setting 部件 G:$G=G_1G_2\cdots G_i\cdots$中增加一个修改标识符 rsid 及其属性值,该属性值的前 4 位 16 进制数都用 0 代替,并从 $H=H_1H_2\cdots H_i\cdots$中读取 4 位 16 进制信息,替换刚增加的修改标识符属性值的后 4 位;

步骤 8:向 setting 部件 G:$G=G_1G_2\cdots G_i\cdots$中增加一个文档变量属性 docVar,并读取 8 位 16 进制水印信息来作为其属性值;

步骤 9:重复步骤 4 到步骤 8,直到编码后的水印信息完全嵌入为止.

3.2 水印提取算法

1) 输入:含水印信息文档 S 以及私钥 k.

2) 输出:水印信息 M.

步骤 1:利用 XML 文档解析技术,从含水印信息文档 S 的 ZIP 包中读取主文档部件 document.xml 的所有内容给 C:$C=C_1C_2\cdots C_i\cdots$,读取 setting 部件的所有内容给 G:$G=G_1G_2\cdots G_i\cdots$;

步骤 2:依次读取 run 元素中含有的删除修改标识符属性值的后 6 位 16 进制数给 M;

步骤 3:读取 setting 部件 G:$G=G_1G_2\cdots G_i\cdots$中的以"0000"开头的修改标识符及其属性值,并把该属性值的后 4 位数据增加到 M 中;

步骤 4:读取 setting 部件 G:$G=G_1G_2\cdots G_i\cdots$中的以 rsid 开头的文档变量,并把该变量的属性值信息赋给 M;

步骤 5:重复步骤 3 到步骤 4 直到所有的信息被提取出来;

步骤 6:利用私钥 k 和非对称加密算法 RSA,对水印信息 M 进行解密,得到解密后的信息 I:$I=D(k,I')$.

4 实验结果与讨论

4.1 实验结果

实验中用到的 OOX 文档全部来源于互联网.为了能够准确地计算出建议方法的嵌入容量和文件大小变化率,本文用 Google 搜索引擎搜索并下载了 1 000 篇 OOX 格式文档作为测试文档.

在测试集中,水印信息被嵌入到测试文档所有可嵌入的地方,即满嵌.

4.1.1 添加删除修改标识符的方法

向主文档部件中添加删除修改标识符的方法所获得的嵌入容量如表 1 所示.比特率 1(R_{b1}) 表示文档中的每个单词可嵌入的比特数,可用如下公式表示:

$$R_{b1}=\frac{\text{平均嵌入容量}}{\text{平均单词数}}=\frac{A_c}{A_w},\tag{1}$$

比特率 2(R_{b2}) 表示每比特的文档可嵌入的比特数,可用如下公式表示:

$$R_{b2}=\frac{\text{平均嵌入容量}}{\text{平均文件大小}}=\frac{A_c}{A_s\times 8}.\tag{2}$$

从表 1 中可以看出,平均 2.26 bit 的水印信息能够嵌入到每个单词中,0.036 bit 的水印信息能够嵌入到每比特的文档中.

嵌入水印前后文件大小的变化率如表 2 所示."+"表示文件长度增加了,"-"表示文件长度减小了.从表 2 中可以看出,在嵌入水印信息后,一些文件长度增加了,一些文件长度减小了.嵌入水印后平均文件长度增加了 0.27%.文件大小的变化率(C)用下式计算:

$$C=\frac{B_e-A_e}{A_e},\tag{3}$$

其中,A_e 表示嵌入水印前文件的大小,B_e 表示嵌入水印后文件的大小.

表 1　添加 rsidDel 水印方法的嵌入容量和比特率

Table 1　Embedding capacity and bit rate for rsidDel method

文件大小/k	文件数	平均文件大小 A_s/bytes	平均单词数 A_w	平均嵌入容量 A_c/bits	R_{b1}/(bit/words)	R_{b2}/(bit/file size)
0~100	654	27 183	3 291	8 072	2.452 66	0.037 12
100~200	83	121 379	17 448	41 156	2.358 74	0.042 38
200~300	45	231 558	21 758	44 074	2.025 70	0.023 79
300~400	40	371 273	37 425	84 566	2.259 62	0.028 47
400~500	38	450 516	53 542	118 488	2.213 00	0.032 88
500~600	29	546 370	80 873	174 576	2.158 64	0.039 90
600~700	33	637 638	77 890	168 044	2.157 46	0.032 94
700~800	29	742 327	109 486	225 210	2.056 98	0.037 92
800~900	16	842 569	149 961	338 358	2.256 31	0.050 20
900~1 000	7	937 453	176 943	399 444	2.257 47	0.053 26
1 000~	26	3 058 694	346 982	822 296	2.369 85	0.033 61
平均值	91	228 226	28 861	65 292	2.262 29	0.035 76

表 2　添加 rsidDel 水印方法的文件大小变化率

Table 2　Change rate of file size for rsidDel method

文件大小/k	平均文件大小/bytes		变化率/%
	嵌入前(A_e)	嵌入后(B_e)	
0~100	27 183	27 403	+0.809 3
100~200	121 379	122 142	+0.628 6
200~300	231 558	232 896	+0.577 8
300~400	371 273	370 354	−0.248 0
400~500	450 516	452 452	+0.429 7
500~600	546 370	548 638	+0.415 1
600~700	637 638	639 249	+0.252 7
700~800	742 327	745 486	+0.425 6
800~900	842 569	839 965	−0.309 0
900~1 000	937 453	938 458	+0.107 2
1 000~	3 058 694	3 056 591	−0.069 0
平均变化率			+0.27

与其他类似方法相比,该值远远低于利用复合文档或网页来隐藏信息的方法,建议方法的文件大小变化率几乎可以忽略,如表 3 所示.例如 Castiglione 等[9]在 Microsoft 复合文档中隐藏信息,使之增大了 25%的长度,Katzenbeisser 等[10]利用网页隐藏信息使之增大 25%的长度.

表 3　文件大小变化率的对比

Table 3　Comparison of file size change rate

嵌入方法	变化率/%
添加删除修改标识符的方法	+0.27
Castiglione 等[9]的方法	+25
Katzenbeisser 等[10]的方法	+25

表 2 中的文件大小平均变化率 0.27%是在全部满嵌的情况下获得的,即嵌入率为 100%.同时本文也做了另外一个实验来验证嵌入水印后的文件大小是否随着嵌入比特率的变化而变化,结果如表 4 所示.从表 4 中可以看出,嵌入水印后的文件大小随着嵌入比特率的减小而减小.当嵌入率约在 90%的时候,文件大小变化率约为 0,即嵌入水印前后文件大小无变化.

表 4　不同嵌入率下的文件大小变化率(添加 rsidDel 的水印方法)

Table 4　File size change rate with different embedding rates (rsidDel method)　%

文件大小/k	不同的嵌入比特率/%					
	10	30	50	70	90	100
0~100	−4.43	−2.57	−1.08	−0.36	+0.02	+0.81
100~200	−3.58	−2.83	−1.18	−0.30	−0.02	+0.63
200~300	−3.93	−2.43	−1.13	−0.28	+0.05	+0.58
300~400	−4.32	−2.37	−1.27	−0.36	−0.03	−0.25
400~500	−3.89	−2.59	−1.26	−0.45	+0.03	+0.43
500~600	−4.81	−2.71	−1.01	−0.42	−0.06	+0.42
600~700	−4.04	−1.93	−0.88	−0.50	−0.03	+0.25
700~800	−4.59	−1.99	−0.99	−0.39	−0.02	+0.42
800~900	−5.02	−1.85	−0.99	−0.40	−0.02	−0.31
900~1 000	−4.65	−3.01	−1.13	−0.42	+0.06	+0.11
1 000~	−5.35	−2.88	−1.06	−0.50	−0.04	−0.07
平均值	−4.42	−2.47	−1.09	−0.40	−0.01	+0.27

4.1.2　添加修改标识符和文档变量的方法

理论上来讲,向 setting 部件中添加的修改标识符和文档变量的数量是无限制的,因此该方法的嵌

入容量也是无限的,但在实际应用中,嵌入过多的信息会显著增加文件的长度,并且容易引起攻击者的怀疑.

4.2 讨论

4.2.1 隐蔽性分析

本文建议的基于冗余属性的水印方法都没有对文档的显示效果和文本的内容产生任何影响,也没有显著改变文件的大小.与其他类似方法相比,建议方法的抗检测能力如表 5 所示.

表 5 各种水印方法的抗检测能力

Table 5 Anti-detection ability of different watermarking methods

嵌入方法	文件结构	文本内容	文字特征检测	
			字体字型字号等	修饰
添加删除修改标识符的方法	是	是	是	是
添加 rsid 的方法	是	是	是	是
添加文档变量的方法	是	是	是	是
Castiglione 等[9]的方法	否	是	是	是
Katzenbeisser 等[10]的方法	否	是	是	是
Liu 等[11]的方法	是	否	是	是
Bolshakov 等[12]的方法	是	否	是	是

4.2.2 嵌入容量

基于删除修改标识符的水印方法的嵌入容量与文档的编辑次数相关,与文档长度关系不大.编辑使命越复杂,产生 run 元素就越多,那么可增加的删除修改标识符也越多,水印的嵌入容量越大.

一般来说,数字水印的嵌入容量和水印的隐蔽性和鲁棒性是相互矛盾的,嵌入容量越大,其隐蔽性就越低,鲁棒性越差.因此在实际应用中应该根据实际情况选择合适的嵌入容量.与其他类似方法相比,本节建议方法的嵌入比特率如表 6 所示.

表 6 各种水印方法的嵌入比特率

Table 6 The embedded bit rate of different watermarking methods

嵌入方法	R_{b1}	R_{b2}
添加删除修改标识符的方法	2.26	0.036
添加 rsid 的方法	理论无限	1%(理论无限)
添加文档变量的方法	理论无限	1%(理论无限)
Castiglione 等[9]的方法	无	8.45%~25%
Liu 等[11]的方法	0.33	0.001 3
Bolshakov 等[12]的方法	0.25	0.004

从表 6 中可以看出,基于添加 rsid 的水印方法和添加文档变量的水印方法的嵌入容量在理论上是无限的,但在实验中,为了增加水印的隐蔽性,防止攻击者的破坏,本文根据嵌入水印后文件大小的变化率,设定了这 2 种方法的嵌入容量为原始文档大小的 1%,即 R_{b2}为 1%.这一嵌入容量可以有效避免攻击者的怀疑.

4.2.3 鲁棒性

常见的针对文本文档的攻击类型有文本内容攻击(包括插入、替换、删除等)和格式属性攻击(包括修改字体字型字号等).本节建议的水印方法没有改变任何显示文本内容和格式信息,建议方法的抗攻击能力如表 7 所示.

表 7 各种方法的抗攻击能力

Table 7 Anti-attack ability of different watermarking methods

水印方法	修改文本内容			修改文字属性	
	插入	替换	删除	字体字型字号等	修饰
添加删除修改标识符的方法	是	是	否	是	是
添加 rsid 的方法	是	是	是	是	是
添加文档变量的方法	是	是	是	是	是
Castiglione 等[9]的方法	是	是	是	是	是
Katzenbeisser 等[10]的方法	是	是	否	是	是
Liu 等[11]的方法	否	否	否	是	是
Bolshakov 等[12]的方法	是	否	否	是	是

5 结论

本文针对 OOX 文档提出一种基于冗余属性的文本数字水印算法,该算法能够增强数字水印的鲁棒性、隐蔽性和嵌入容量.在研究 OOX 格式文档的逻辑存储结构以及各部件和各节点之间关系的基础上,通过向 OOX 文档包中的有关部件增加冗余属性等方式,把水印信息嵌入到有关部件中去.其中基于添加删除修改标识符、修改标识符和文档变量的方法具有较强的鲁棒性和较大的嵌入容量,能够抵抗“另存为”、“清除格式”等攻击.这些水印方法都没有改变原始文档显示字符的任何格式信息,这是以往针对格式文档提出的水印方法做不到的,同时也没有改变原始文档的任何语义信息,这也是以往的基于语义(自然语言)的水印方法做不到的.实验结果说明了算法的可行性.

参考文献
References

[1] Wang X,Reeves D S.Robust correlation of encrypted attack traffic through stepping stones by flow watermarking [J].IEEE Transactions on Dependable and Secure Computing,2010,8(3):434-449
[2] Jaseena K U,Anita J.An invisible zero watermarking algorithm using combined image and text for protecting text documents[J].International Journal on Computer Science and Engineering,2011,3(6):2265-2272
[3] FU Zhangjie,SUN Xingming,LIU Yuling.Text split-based steganography in OOXML format documents for covert communication [J]. Security and Communication Networks,September,2012,5(9):957-968
[4] Microsoft Corp.2007 Microsoft office system[EB/OL]. [2011-07-10]. http://msdn. microsoft. com/ en-us/library/bb726436(v=office.12).aspx
[5] Van Vugt W.2007 Office document:Open XML markup explained[EB/OL].[2011-07-10].http://www.microsoft.com/downloads/details.aspx? FamilyID=6f264d0b-23e8-43fe-9f82-9ab627e5eaa3&displaylang=en
[6] Microsoft Corp.Microsoft core XML services (MSXML) 6.0 [EB/OL]. [2011-07-10]. http://www. microsoft.com/download/en/details. aspx? displaylang = en&id=3988
[7] ECMA.Office open xml file formats standard-final draft [EB/OL]. [2011-07-10]. http://www. ecma-international.org/news/TC45_current_work/TC45-2006-50_final_draft.htm
[8] Rice F.Introducing the office (2007) open xml file formats [EB/OL]. [2011-07-10]. http://msdn2.microsoft.com/en-us/library/aa338205.aspx
[9] Castiglione A,Santis A D,Soriente C.Taking advantages of a disadvantage:Digital forensics and steganography using document metadata[J].The Journal of Systems and Software,2007,80(5):750-764
[10] Katzenbeisser S, Petitcolas F. Information hiding techniques for steganography and digital watermarking [J].The EDP Audit,Control,and Security Newsletter,2000,28(6):1-2
[11] Liu T Y,Tsai W H.A new steganographic method for data hiding in Microsoft word documents by a change tracking technique [J]. IEEE Transactions on Information Forensics and Security,2007,2(1):24-30
[12] Bolshakov I A.A method of linguistic steganography based on collocationally:Verified synonymy[C]//Proc 6th Information Hiding Workshop, Toronto, ON, Canada,2004

Digital watermarking algorithm based on redundant attributes for OOX documents

FU Zhangjie[1,2] SUN Xingming[1,2]
1 School of Computer and Software,Nanjing University of Information Science & Technology,Nanjing 210044
2 Jiangsu Engineering Center of Network Monitoring,Nanjing University of Information Science & Technology,Nanjing 210044

Abstract For the most widely used Office Open XML (OOX) format documents (MS Office 2007—2013),this paper proposes a robust watermarking algorithm based on the redundant attribute.Watermarking information,which is linked to the format information of document,can be imperceptibly embedded into the redundancy space of OOX parts.The method can resist text content-based active attacks.The method this paper proposed has better robustness and embedding capacity,which can resist "Clean Format","Save As" and other active attacks.The proposed watermarking method does not change any format or semantic information of the original documents,which no existing methods would do.The experiment demonstrates the feasibility of the algorithm.

Key words digital watermarking;Office Open XML;redundant attributes;robustness;imperceptibility