An Attack-localizing Watermarking Scheme for Natural Language Documents

Gaurav Gupta Josef Pieprzyk Hua Xiong Wang

Centre for Advanced Computing - Algorithms and Cryptography
Department of Computing
Macquarie University
Sydney, NSW 2109, Australia
{ggupta,josef,hwang}@ics.mq.edu.au

ABSTRACT

We present a text watermarking scheme that embeds a bitstream watermark W_i in a text document P preserving the meaning, context, and flow of the document. The document is viewed as a set of paragraphs, each paragraph being a set of sentences. The sequence of paragraphs and sentences used to embed watermark bits is permuted using a secret key. Then, English language sentence transformations are used to modify sentence lengths, thus embedding watermarking bits in the Least Significant Bits (LSB) of the sentences' *cardinalities*. The embedding and extracting algorithms are public, while the secrecy and security of the watermark depends on a secret key K. The probability of *False Positives* is extremely small, hence avoiding incidental occurrences of our watermark in random text documents. *Majority voting* provides security against text addition, deletion, and swapping attacks, further reducing the probability of *False Positives*. The scheme is secure against the general attacks on text watermarks such as reproduction (photocopying, FAX), reformatting, synonym substitution, text addition, text deletion, text swapping, paragraph shuffling and collusion attacks.

Keywords

watermarking, permutation, copyright

1. INTRODUCTION

With the emergence of Internet and electronic communication, illegal distribution of media has become a cause of concern for the publishers and legitimate owners. To dissuade people from getting involved in such activities, organizations insert copyright marks in media that they sell. These marks establish the company's ownership over the media in events of disputes. Identification of the person involved in illegal distribution of copyrighted material is called *fingerprinting*. Digital watermarking techniques are used to accomplish both *Copyright Protection* and *Fingerprinting*.

Text documents [1, 6, 5, 7, 8, 11, 12, 13, 14, 18], images [9], audio [2, 4] and video [10, 16] files are common media objects that are watermarked. Schemes have also been proposed to watermark media like music sheets [15] and numeric sets [17]. The information that companies want to copyright is mostly in the form of text documents. Text documents are, however, the most difficult to watermark because text manipulations are guided by strict rules in terms of grammar, syntax, semantics, context-based selection of a word from a set of synonymous words, etc; while in the case of other media, there is large amount of redundant information to manipulate. For example, human visionary system cannot distinguish between an original image and a watermarked image with the last few LSBs in certain pixels flipped. Similar is the case with audio and video files. But in text documents, grammatical rules need to be preserved while making any changes. There has been significant work done in format-based text watermarking using inter-word and interspace spacing, justification, alignment, character height and width, etc [6, 5, 7, 8, 11, 14, 18]. The common problem these techniques have is that watermark cannot survive reformatting and reproduction attack as they introduce loss of formatting information in the document. Alternatively, the attacker can simply re-type the entire document which would be watermark-free.

Synonym substitution watermarking schemes [12] are resilient to the above mentioned trivial attacks but not to random synonym substitutions made by the attacker. Even more importantly, words can not always be replaced by their *exact* synonyms. Hence, the quality of the documents is depreciated by synonym substitution.

1.1 Current scenario

The current focus in text watermarking is on syntactic watermarking [1, 19] where language syntax structures are modified to embed watermarks. Notable progress has been made in [1], where watermark bits are embedded in sentences using the following transformations -

1. **Adjunct movement**: Inserting an adjunct (example, "*Usually*", "*Generally*", etc) at many of the possible

ASIACCS '06, March 21-24, 2006, Taipei, Taiwan.

positions in a sentence.

2. **Clefting**: Explicit emphasis on the mandatory subject in the sentence. (example, "**We are concerned with** < *subject* >" to "***it is*** < *subject* > **we are concerned with**")

3. **Passivization**: Changing of voice from active to passive and vice versa. (example "**He led me**" to "**I was led by him**")

4. **Combination of the above**.

This scheme has certain drawbacks such as 1) overhead introduced because of parsing each sentence, numbering the nodes and creating a hash for each node. 2) requirement of *marker* sentences reduces the capacity. 3) not resilient to multiple sentence transformation attacks.

Table 1 summarizes the central ideas of the current watermarking schemes.

1.2 Outline of the proposed scheme

In the proposed watermarking scheme, the sequence of the paragraphs and sentences used to embed the watermark is permuted. This results in effect of attacks getting localized to a small region and not getting spread across the document. Watermark bits are physically embedded by modifying the sentences' word counts. Error-correcting codes and majority voting are used to embed watermark bits at multiple locations providing increased security against attacks. The watermark contents are signed using private key of user and publisher which prevents the publisher from framing an innocent user. The watermark bitstream contains a collusion-secure code (described in detail later) to identify colluding users.

1.3 Type of adversary

The attacker is assumed to have the capabilities to -

1. Add and/or delete sentences from the document.

2. Swap sentences within the same or between different paragraphs.

3. Make natural language transformations on sentences.

4. Shuffle paragraphs in the document.

5. Collude with other users to compare and modify the document.

1.4 Organization of paper

In Section 2, we discuss the general mathematical model of the scheme and definitions used for rest of the paper. In Section 3, we propose our scheme, the sequence-permutation, watermark composition, watermark embedding, extracting and verification step. In Section 4, we discuss these attacks in greater detail, the probability of False Positives and capacity analysis. Experimental results are given in Section 5. Conclusion and future work follow in Section 6.

2. MATHEMATICAL MODEL AND DEFINITIONS

2.1 Mathematical Model

We represent the watermarking scheme as WS, where

$$WS = < \{P, W_i, K\}, \{\xi, \zeta, \psi\} > \tag{1}$$

P=y-paragraph text $\{p_1, p_2, \ldots, p_y\}$

p_i=i^{th} paragraph with x_i-sentences $\{s_{i1}, s_{i2}, \ldots, s_{ix_i}\}$

s_{ij}=j^{th} sentence in i^{th} paragraph

d_{ij} number of tokens/words in s_{ij}

$W_i = \{w_1, w_2, \ldots, w_n\}$ is the watermark to be inserted where $\forall i, w_i \in \{0, 1\}$

$K = k$-bit secret key

Watermark insertion $\xi : W_i \times P \times K \to P^{(w)}$, where $P^{(w)}$ is watermarked text

Watermark extraction $\zeta : P^{(w)} \times K \to W_e$ (extracted watermark)

Watermark verification $\psi : W_e \times W_i \to \{true/false\}$

2.2 Definitions

$d_i = |s_i|$ gives the number of words in sentence s_i

$d_i = \{b_{i,1}, b_{i,2}, \ldots, b_{i,k_i}\}$ where $b_{i,j} \in \{0, 1\}$, $k_i = \lceil \log_2 d_i \rceil$ is the binary representation of d_i with b_{i1} as the LSB and so on.

Watermark $W = \{w_1, w_2, \ldots, w_m\}$ where w_i is the i^{th} bit of the watermark.

Lexicographically sorted permutations for a set of n elements are $\rho_1^n, \rho_2^n, \ldots, \rho_{n!}^n$. ρ_i^n gives the i^{th} permutation of n elements. $\varrho_{i,j}^n$ gives the value of the j^{th} element in ρ_i^n.

Majority Voting - $\forall i, a_i \in \{0, 1\}$.

$$majority(a_1, a_2, \ldots, a_n) = \begin{cases} 1 & \text{if } |a_i = 1| > \frac{n}{2} \\ 0 & \text{otherwise} \end{cases}$$

Text document $P = \{p_1, p_2, \ldots, p_y\} = \Big\{\{s_{\alpha_1+1}, \ldots, s_{\alpha_1+x_1}\}, \ldots, \{s_{\alpha_y+1}, \ldots, s_{\alpha_y+x_y}\}\Big\}$ where p_i is the i^{th} text paragraph and s_i is the i^{th} text sentence.

$p_i = \{s_{\alpha_i+1}, s_{\alpha_i+2}, \ldots, s_{\alpha_i+x_i}\}$

$$\alpha_i = \begin{cases} 0 & \text{if } i = 1 \\ \sum_{j=1}^{i-1} x_j & \text{if } 2 \le i \le y \end{cases}$$

$|p_i|$ defines number of sentences it contains.

τ = number of paragraphs in which each watermark bit is embedded.

3. PROPOSED SCHEME

In order to limit distortions caused by modifications made by the attacker, we permute the sequence of sentences and

Table 1: General ideas of current watermarking schemes

Modifications made based on watermark bit	Scheme
Interword spacing (example - 10-pixels if bit=0; 11-pixels otherwise)	[6, 7, 14, 18]
Interline spacing (example - 10-pixels if bit=0; 11-pixels otherwise)	[6, 14, 18]
Abbreviation and Synonym substitution x (example - “must” if bit=0; “should” otherwise) (example - “a.m.” if bit=0; “A.M.” otherwise)	[12]
Sentence structures (example - “He led me” if bit=0; “I was led by him” otherwise)	[1]

paragraphs used to embed the watermark. It needs to be emphasized that sentences/ paragraphs are not physically permuted but only the sequence in which they will be “picked” to embed the watermark is permuted. Embedding each watermark bit in multiple paragraphs (say μ) results in any $\frac{\mu}{2}+1$ unmodified bits leading to successful recovery of the watermark.

3.1 Sequence permutation

In the current implementation, we use AES outputs to generate permutations that results in higher cryptographic security and more importantly introduces an “uncertainty effect” that is described in Section 4. With AES, the key size $k \in \{128, 192, 256\}$. However, one can use any other method to generate permutations.

1. The set of paragraph indices $\{1, 2, \ldots, y\}$ is sorted in ascending order of the number of sentences they contain to $G = \{g_1, g_2, \ldots, g_y\}$ such that $|p_{g_i}| \leq |p_{g_j}|, i < j$. If two paragraphs, p_i, p_j contain number of sentences, p_i precedes p_j if $i < j$. This step nullifies paragraph shuffling attacks.

2. In binary notation, $\lceil \log_2 y \rceil$ (say δ) bits are required to represent index of any given paragraph in a set of y paragraph.

3. Vector V is a k-bit vector initialized to secret V_{iv}.

4. $\forall i$, input to AES is $V \bigoplus g_i$ and key is K. The first δ bits of encrypted output (mod y) gives the paragraph's position θ_i in new sequence.

5. If V generates a *valid permutation* ($\forall(i,j),\ i \neq j, 1 \leq i \leq y, 1 \leq j \leq y, \theta_i \neq \theta_j$), final test vector V_f=V, otherwise reject V, repeat step 3,4 with $V = V + 1$.

6. The new paragraph sequence is given by $\{\theta_1, \theta_2, \ldots, \theta_y\}$. This essentially means that p_{θ_i} is used before p_{θ_j} if $i < j$. As an example, if the sequence set is {5, 1, 2, 3, 4} such that $\theta_1 = 5$ and $\theta_2 = 1$, then paragraph 5 is used **before** paragraph 1 in watermark embedding.

7. For $1 \leq i \leq y$, $\rho^{x_i!}_{(\theta_i)^K(mod(x_i!))}$ is the new sequence of the sentences to be used within the paragraph i. This permutation is generated using Algorithm 1.

8. The resulting paragraph sequence is $\Theta = \{\theta_1, \theta_2, \ldots, \theta_y\}$ and the sentence sequence is given in Table 2

```
$x = x_i$;
for $l = 1; l \leq x_i; l = l + 1$ do
    $oldindex[l] = l$;
end
$j = \theta_i^K(mod(x!))$;
$q = 1$;
if $x > 0$ then
    $s = \lceil \frac{j}{(x-1)!} \rceil$;
    $j = j\%(x-1)!$;
    $newindex[q] = oldindex[s]$;
    for $l = s; l \leq x - 1; l = l + 1$ do
        $oldindex[l] = oldindex[l+1]$;
    end
    $q = q + 1$;
    $x = x - 1$;
end
```

Algorithm 1: Sentence sequence generation: Generating ${x_i}^{th}$ permutation from a lexicographically sorted set of permutation

As an illustration, let a document contain 5 paragraphs $\{a, b, c, d, e\}$ with 7, 8, 5, 3, and 6 sentences respectively. Let the new paragraph sequence be {4, 1, 2, 5, 3} and the new sentence sequence be {2, 1, 3} for paragraph d (which is now in first position), {5, 3, 7, 2, 4, 1, 6} for paragraph a, {8, 1, 4, 2, 3, 7, 5, 6} for paragraph b, {3, 4, 6, 2, 5, 1} for paragraph e, and {1, 4, 3, 2, 5} for paragraph c. This means that the sequence of paragraphs used to embed watermark will be paragraph d, then paragraph a, b, e and finally c. While using d (which contains 3 sentences) the sequence of sentences used for watermark embedding will be sentence 2, sentence 1 and finally sentence 3; and so on for sentences in other paragraphs.

For generating a permutation of a set containing y elements, the first element can be chosen in y ways, the second in $(y-1)$ ways and so on, and the total combinations (with repetitions) are y^y, hence, the probability of getting a permutation when choosing elements with repetitions is given be the following equation:

$$P(\theta_i \neq \theta_j, \forall i, \forall j, i \neq j) = (\frac{y!}{y^y}) \qquad (2)$$

Results of the experiments conducted to generate permutations using AES-128 confirm the results. Table 3 provides the comparison of empirical results with theoretical values.

3.2 Watermark composition

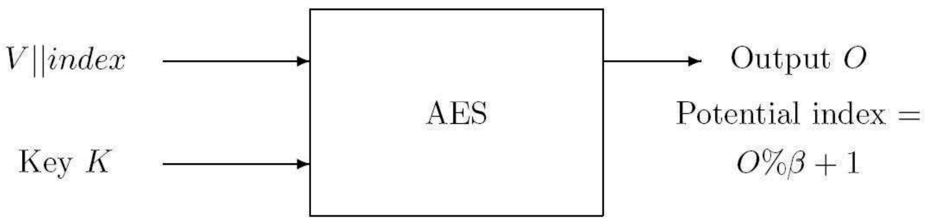

Figure 1: Generating a paragraph permutation using AES

Table 2: New sequence of sentences that will be used to embed the watermark

$$\begin{array}{lcl}
\Big\{\{\rho^{x_1!}_{(\theta_1)^K(mod(x_1!)),1},\ldots,\rho^{x_1!}_{(\theta_1)^K(mod(x_1!)),x_1}\}, & & \Big\{\{t_{(1,1)},t_{(1,2)},\ldots,t_{(1,x_1)}\}, \\
\{\rho^{x_2!}_{(\theta_2)^K(mod(x_2!)),1},\ldots,\rho^{x_2!}_{(\theta_2)^K(mod(x_2!)),x_2}\},\ldots, & = & \{t_{(2,1)},t_{(2,2)},\ldots,t_{(2,x_2)}\},\ldots, \\
\{\rho^{x_y!}_{(\theta_y)^K(mod(x_y!)),1},\ldots,\rho^{x_y!}_{(\theta_y)^K(mod(x_y!)),x_y}\}\Big\} & & \{t_{(y,1)},t_{(y,2)},\ldots,t_{(y,x_y)}\}\Big\}
\end{array}$$

Table 3: Comparison of empirical results with theoretical values

Number of elements	Keys searched	
	Empirical result	Theoretical Value
2	1.88	2
3	4.44	4.5
4	7.33	10.6667
5	16.16	26.04171
6	64.27	64.8
7	145.22	163.401
8	516.55	416.102
9	1140.77	1067.63
10	3381.77	2755.73
11	6240.94	7147.66
12	15307.72	18613.9
13	37694.88	48638.8
14	108803.61	127463
15	433622.72	334865
16	1097114.16	881658
17	2004049	2330000
18	6203832.22	6150000
19	16376576.78	16300000

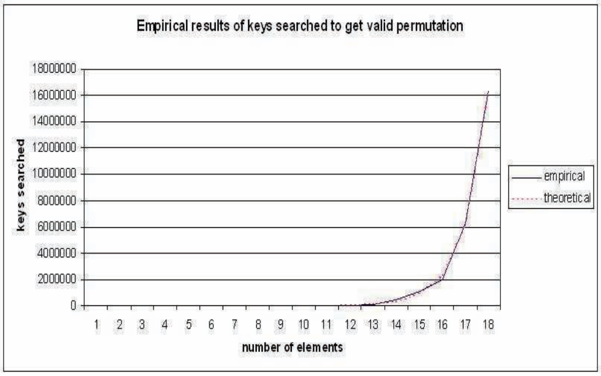

Figure 2: Keys required to get a valid permutation using AES-128: Experimental results are in accordance with theoretical values

It is crucial to construct the watermark such that

1. Watermark can identify the publisher and user successfully.
2. Publisher cannot frame an innocent user.
3. Watermark can withstand collusion attacks.

To satisfy the first requirement, the watermark simply needs to have two components - a publisher component and a user component. But by this method, the publisher can generate any desired watermark and thus frame an innocent user. Hence we adopt the following protocol -

1. The publisher sends user a watermark W_u carrying the user identity.
2. User signs W_u with his private key Pr_u and sends publisher the "signed user component" $S_{Pr_u}(W_u)$.
3. Publisher verifies the correctness by verifying the signed user component with the user's public key Pu_u. He then appends the document specific publisher component W_p to the signed user component and signs it with his private key Pr_p.
4. Final watermark W_i is $S_{Pr_p}(W_p||S_{Pr_u}(W_u))$.

The court can verify the watermark with the public keys of publisher and user. Neither the publisher, nor the user can tamper with the watermark without the knowledge of the other person's private key. A small problem with this scheme is that since the "user components" of various users will differ, hence multiple users can collude and destroy the watermark. Hence the physical W_u should be chosen such that colluding users can successfully be identified.

For this purpose, we use the logarithmic length *c-secure* codes proposed in [3]. These codes can successfully identify at least one of the c colluding users from a group of n users. Given integers N and c, and an error tolerance metric $\epsilon > 0$, set $n = 2c$, $L = 2c\log(2N/\epsilon)$, and $D = 2n^2\log(4nL/\epsilon)$. The code $\Gamma'(L, N, n, d)$ (details in [3]) is $c - secure$ with ϵ-error.

Let the codeword for the user for which the document is being watermarked be $W_u = \{w_1, w_2, \ldots, w_{Ld(n-1)}\}$. Boneh-code enables us to identify colluding parties of at most $c = n/2$ users with a probability of $1 - \epsilon$. For further details about these collusion-secure codes, please refer to [3].

Now the watermark $S_{Pr_p}(W_p||S_{Pr_u}(W_u))$ satisfies all three requirements mentioned at the beginning of the section and can be embedded.

3.3 Watermark embedding step

Before proceeding to the watermark embedding algorithm, we describe how watermark bits will be physically carried in the document. Let the number of words in a sentence s_i be d_i and the binary representation of d_i be $d_{i,1}, d_{i,2}, \ldots, d_{i,z}$ such that $d_{i,1}$ is the LSB. We utilize $d_{i,1}$ and $d_{1,2}$ to carry the watermark. If we want to embed two bits w_1 and w_2 in a sentence s_i, then

1. Set $d_{i,1} = w_1$, $d_{i,2} = w_2$. Let the new value of d be d'.
2. Transform the sentence such that it contains d' number of words using one or more of the following (and other) transformations -
   (a) Change of voice from active to passive and vice versa. Example, *"The cops rewarded Anjali"* ↔ *"Anjali was rewarded by the cops"*.
   (b) Addition/deletion of an adjunct to/from the sentence. Example, *"The company praised Gunjan"* ↔ *"**It was the** company which praised Gunjan"*.
   (c) Addition/Removal of optional articles. Example, *"Maya was cutting up the trees for Christmas"* ↔ *"Maya was cutting up trees for Christmas"*.
   (d) Grouping of multiple subjects. Example, *"Ravi married Tina"* ↔ *"Ravi and Tina got married"*.
   (e) Addition/removal of coordinate conjunctions. Example, *"Mohit started to sing **and** Gaurav began playing the guitar"* ↔ *"Mohit started to sing, Gaurav began playing the guitar"*.
   (f) Introducing, or eliminating *"then"* from the **if ... then** pair of correlative conjunctions. Example *"If this is what you want, **then** this is what you'll get"* ↔ *"If this is what you want, this is what you'll get"*.

Given the information on how we are going to store watermarking bits in the document, the watermark embedding algorithm is given below -

1. All the sentences and sentences are marked as *"unused"*.
2. Choose the paragraphs corresponding to the next τ *"unused"* indices from the new paragraph sequence. (Go to start of sequence if end of sequence reached).
3. Take the first available *"unused"* sentences (using new sentence sequence) from the τ paragraphs and embed the first γ bits in them using Algorithm 2, where each watermark bit is inserted $\mu = \frac{\tau\beta}{\gamma}$ times. The watermark bit is physically embedded using English language transformations (discussed in [1]). For example, for a sentence **"This is not so difficult to understand"** having word count of 7 (0111) if we need

to reduce one word from it to embed the watermark bits (10) in the 2 LSBs of its word count, preserving its meaning, we can change the sentence to **"Understanding this is not so difficult"** which has a word count of 6 (0110).

4. Delete the γ watermark bits embedded in the first step from the watermark.

5. Mark the sentences chosen in step 2 as *"used"* and if all the sentences of a paragraph are marked as *"used"*, mark the paragraph as *"used"*.

6. Repeat steps 2-5 till the entire watermark is embedded.

The pseudo-code for the above procedure is provided in Algorithm 2.

```
$counter = 1$;
for $l = 1; l < y; l++$ do
    $q_l = \{s_{t_{(l,1)}}, s_{t_{(l,2)}}, \ldots, s_{t_{(l,x_l)}}\}$;
end
$Q = \{q_1, q_2, \ldots, q_y\}$;
for $i = 1; i \leq m; i+= \beta$ do
    for $j = 1, j \leq \tau; j++$ do
        $temp = (j + counter)(\%y)$;
        $s_{t'_j} = s_{t_{(temp,1)}}$;
        $b_j = |s_{t'_j}|$;
        $q_{temp} = q_{temp} - s_{t_{(temp,1)}}$;
        if $q_{temp} = \phi$ then
            $Q = Q - q_{temp}$;
            $y = y - 1$;
        end
    end
    for $j$=1;$j \leq \frac{\tau}{2}$;$j$++ do
        for $l$=1;$l \leq \beta$;$l = l + 1$ do
            $b_{jl} = w_{((j+l-1)\%\gamma)+((counter-1)\times\gamma)}$;
            $b_{(j+\frac{\tau}{2})(\beta-l+1)} = b_{jl}$;
        end
        transform sentences according to new $d$ by
        applying English language transformations;
    end
    $counter = (counter + \tau)(\%y)$;
end
```

Algorithm 2: Watermark embedding

3.4 Watermark Extracting and Verification

The watermark bits are extracted in the same permuted sequence used while embedding (Algorithm 1). The embedding process is similar to embedding process except that in this case we set the watermark bits to the LSBs of the word counts. Finally, *majority-voting* is applied on the multiple instances of each watermark bit. Table 4 illustrates how majority voting works.

The extracted watermark W_e is compared to the inserted watermark W_i and if the Hamming Distance is less then a maximum tolerance value Ω, the watermark is acceptable, otherwise it is rejected (in which case collusion detection is performed using algorithm suggested in [3].

4. ANALYSIS

4.1 Attacks

We discuss the various attacks possible on the watermarked document and degree of resilience offered by our scheme:

1. **Reformatting/Reproducing attacks**: The watermark is carried in the structure of the sentences and not the formatting information (such as interword/ interline spacing, font characteristics, indentation, etc). Hence, changing these attributes does not alter the watermark.

2. **Sentence addition/ deletion**: Addition/ deletion of a sentence (say s_i) results in the sentence sequence being distorted for the paragraph (say p_j) containing s_i. But each watermark bit carried in sentences of p_j is embedding in $\mu - 1$ sentences in other paragraphs and can be correctly extracted using majority voting (explained in 3.4). Hence, the watermark can withstand this attack. In the worst case, if the attacker adds/ deletes $\frac{\mu}{2}$ sentences carrying the same watermark bits, the watermark might be destroyed. Thus, the watermark can survive at least $\frac{\mu}{2} - 1$ additions/deletions (**lower bound**).

3. **Text swapping**: Text swapping refers to selecting two sentences $s_i \in p_j$ and $s_{i'} \in p_{j'}$ from a document and swapping them. The sentence sequence is not disturbed in this case and only watermark bits corresponding to the swapped sentences are affected. Like in sentence addition/ deletion, the other $\mu - 1$ instances of the watermark bits result in correct watermark retrieval. Here also, the watermark can withstand at least $\frac{\mu}{2} - 1$ swaps.

4. **Paragraph shuffling**: In 3.1, we first sort the paragraph sequence according to cardinality before carrying out the permutation operation. Hence, even if the paragraphs are shuffled by the attacker, the original permutation will be restored when extracting the watermark. Hence, the scheme is totally secure against paragraphs being shuffled.

5. **Collusion attack**: Boneh-code is inserted as the user component W_u. If an illegal copy is discovered, then the algorithm described in [3] is executed which outputs the member(s) of the collusion with high probability.

6. **Cryptographic attacks**: AES lies at the core of our scheme as it is used to generate permutations. First the attacker needs O(2^k) time to perform an exhaustive search on K. For each potential K, however, the attacker would need to generate potential index sets, which requires O(2^k) time. Hence the time complexity of an exhaustive search attack is O(2^{2k}). More importantly, a key K' different to key used to embed the watermark (K) can still, with high probability, generate a valid permutation. This permutation is different to permutation generated while watermark embedding and this introduces an "uncertainty effect" where the attacker cannot be sure of the correctness of a permutation generated by a random key.

Table 4: Illustration of majority voting

Copy	Watermark bits				
	w_0	w_1	w_2	w_3	w_4
1	0	0	1	***1***	1
2	***1***	0	1	0	1
3	0	0	***0***	0	***0***
4	0	0	1	0	1
5	0	***1***	1	***1***	1
output	$w_0 = 0$	$w_1 = 0$	$w_2 = 1$	$w_3 = 0$	$w_4 = 1$

4.2 False Positive probability

The probability of an m-bit watermark matching another watermark extracted from a randomly picked document is 2^{-m}. Since each bit of the watermark is actually embedded at μ positions, $\frac{\mu}{2}+1$ of those μ bits should match corresponding bit of our watermark. This makes the actual probability of having *False Positives* $= 2^{-(m+\frac{\mu}{2}+1)}$. This is lower than probability of false positives in [1].

4.3 Watermarking Capacity

The optimal capacity utilization is when a document contains $\sum_{j=1}^{y} x_j$ sentences and each sentence carries β bits. Every watermark bit is embedded in $\mu = \frac{\tau\beta}{\gamma}$ sentences. Hence the watermarking capacity of our scheme is $\frac{\beta\times\sum_{j=1}^{y} x_j}{\mu}$ $= \frac{\gamma\times\sum_{j=1}^{y} x_j}{\tau}$.

5. EXPERIMENTAL RESULTS

5.1 Implementation details

The experiments were carried out in Unix using C language on Pentium 4 2.4 GHz processor. Usage of C as a programming language makes the implementation extremely efficient in terms of time. Quartz digital signature scheme was utilized for producing digital signatures since the size of these signatures is very small (128-bits). Java implementation provided by Christophe Wolf was used to generate signatures.

5.2 Results

We used 5 sample documents of varying sizes (from 16505 words to 46271 words) and paragraph structures to embed watermarks of 5 sizes constructed using quartz digital signature scheme (which produces 128-bit digital signatures) and analyzed the results of the experiments. It should be noted that the watermark embedded essentially consists of two signatures - user's and publisher's) and optionally other information like timestamp, metadata, padding and so on. The number of bits that change are proportional to the watermark size as indicated in Table 5.

The net change in document size is fairly constant for a specific document. The change in document size is less than 1% in most of the cases (refer to Table 6). Hence, quantitatively speaking, there is minimal distortion to the document. It was observed that the documents with larger paragraphs had fewer changes as compared to documents with smaller paragraphs. This also suggests that the paragraph structure, and thereby the permutation we select play a key role in determining the number of words that will be added or deleted from the document.

6. CONCLUSION AND FUTURE WORK

Our scheme is shown to be resilient against document reproduction, reformatting, synonym substitution, text addition, text deletion, text swapping and paragraph shuffling. Previous watermarking schemes [6, 5, 7, 8, 11, 12, 13, 14, 18] are not secure against majority of these these attacks. Compared to [1], our scheme provides higher security (deterministic resilience to at least $\frac{\mu}{2}-1$ changes against probabilistic resilience to single change in [1]) against text addition, text deletion, text swapping and total security against paragraph shuffling. It is also secure against collusion attacks. An exhaustive cryptographic attack on the scheme takes $O(2^{2k})$ time (k being the size of key used). With high probability, the scheme can successfully identify at least one of the colluding users in event of a collusion attack. The capacity of the scheme is $\frac{\gamma\times\sum_{j=1}^{y} x_j}{\tau}$ watermark bits.

We are currently working on the following aspects of our scheme -

1. Designing indigenous collusion-secure codes: Currently, we are using collusion secure codes given by Boneh. We are trying to design alternative collusion-secure codes which have shorted length but similar security.

2. Increasing the capacity of the scheme by using an error correcting code instead of the currently used repetitive correcting code/ majority-voting: In the existing scheme, each watermark bit is embedded in multiple paragraphs making it a repititive code that reduces the watermark-carrying capacity of a document. Instead, if error-correcting codes are utilized, capacity would significantly improve.

3. Extending the scheme to multilingual documents incorporating the grammatical aspects of various languages: In the current implementation, only English documents are watermarked. Watermarking other documents would required analysis of grammer rules of that language. This is more of an implementation issue than a design issue as the underlying principle is the same.

7. ACKNOWLEDGEMENTS

We would like to thank Vijayakrishnan Pasupathinathan, Robert Dale, Krystian Matusiewicz and Christophe Tartary for their valuable contributions towards this research.

Table 5: Number of bit changes in document with increase in watermark size

Watermark Size	Bit Changes				
(in bits)	document 1	document 2	document 3	document 4	document 5
320	1802	1762	1431	1269	1280
400	1903	1895	1507	1436	1334
480	2003	2037	1589	1522	1438
560	2182	2121	1657	1631	1526
640	2301	2266	1717	1726	1604

Table 6: Number of words added to document with increase in watermark size

Watermark Size	Words Added				
(in bits)	document 1	document 2	document 3	document 4	document 5
320	-8	8	0	1	-14
400	-11	2	-4	-16	-12
480	-15	-5	-17	-1	-19
560	-14	-5	-20	-10	-26
640	-11	-7	-24	17	-14

The second author is supported by Australian Research Council grants DP0345366 and DF0451484.

8. REFERENCES

[1] M. Atallah, V. Raskin, M. Crogan, C. Hempelmann, F. Kerschbaum, D. Mohamed, and S. Naik. Natural language watermarking: design, analysis, and a proof-of-concept implementation. In *Proc. of 4th International Workshop on Information Hiding, IH 2001. LNCS*, volume 2137, pages 185–199. Springer-Verlag, Heidelberg, 2001.

[2] P. Bassia and I. Pitas. Robust audio watermarking in the time domain. In *9th European Signal Processing Conference (EUSIPCO'98)*, pages 25–28, Island of Rhodes, Greece, 8–11 1998.

[3] D. Boneh and J. Shaw. Collusion-secure fingerprinting for digital data. *Lecture Notes in Computer Science*, 963:452 – 465, 1995.

[4] L. Boney, A. H. Tewfik, and K. N. Hamdy. Digital watermarks for audio signals. In *International Conference on Multimedia Computing and Systems*, pages 473–480, 1996.

[5] J. Brassil, S. Low, N. Maxemchuk, and L. O'Gorman. Marking text features of document images to deter illicit dissemination. In *Proc. of the 12th IAPR International Conference on Computer Vision and Image Processing*, volume 2, pages 315 – 319, Jerusalem, Israel, October 1994.

[6] J. Brassil, S. Low, N. F. Maxemchuk, and L. O'Gorman. Hiding information in documents images. In *Conference on Information Sciences and Systems (CISS-95)*, 1995.

[7] N. Chotikakamthorn. Electronic document data hiding technique using inter-character space. In *Proc. of The 1998 IEEE Asia-Pacific Conference on Circuits and Systems, IEEE APCCAS 1998*, pages 419–422, Chiangmai, Thailand, November 1998.

[8] N. Chotikakamthorn. Document image data hiding technique using character spacing width sequence coding. In *Proc. of International Conference on Image Processing, ICIP 1999*, volume 2, pages 250–254, Kobe, Japan, October 1999.

[9] I. Cox, J. Kilian, T. Leighton, and T. Shamoon. Secure spread spectrum watermarking for multimedia. Technical Report 128, NEC Research Institute, August 1995.

[10] F. Hartung and B. Girod. Digital watermarking of raw and compressed video. In *Proc. European EOS/SPIE Symposium on Advanced Imaging and Network Technologies*, Berlin, Germany, October 1996.

[11] H. Ji, J. Sook, and H. Young. A new digital watermarking for text document images using diagonal profile. In *Proc. of Second IEEE Pacific Rim Conference on Multimedia, PCM 2001. LNCS*, volume 2195, pages 748 –, Beijing, China, October 2001. Springer-Verlag, Heidelberg.

[12] M. S. Kankanhalli and K. F. Hau. Watermarking of electronic text documents. *Electronic Commerce Research*, 2(1-2):169–187, 2002.

[13] S. Low, N. Maxemchuk, J. Brassil, and L. O'Gorman. Document marking and identification using both line and word shifting. In *Fourteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Bringing Information to People, INFOCOM 1995*, volume 2, pages 853–860, Boston, USA, April 1995.

[14] N. Maxemchuk and S. Low. Marking text documents. In *Proc. of International Conference on Image Processing*, page 13, Washington, USA, 26-29 October 1997.

[15] M. Monsignori, P. Nesi, and M. Spinu. Watermarking music sheets. In *Proc. of Second IEEE Pacific Rim Conference on Multimedia, PCM 2001. LNCS*, volume 2195, pages 646–653, Bejing, China, 2001.

[16] T.-S. K. K.-R. K. Seung-Jin Kim, Suk-Hwan Lee and K.-I. Lee. A video watermarking using the 3-d wavelet transform and two perceptual watermarks. In *Proc. of Fourth International Workshop on Digital Watermarking, IWDW 2002. LNCS*, volume 3304, pages 294 – 303, Seoul, Korea, October 2004.

Springer-Verlag, Heidelberg.

[17] R. Sion, M. Atallah, and S. Prabhakar. On watermarking numeric sets. In *Proc. of First International Workshop on Digital Watermarking, IWDW 2002. LNCS*, volume 2163, pages 130–146, Seoul, Korea, November 2002. Springer-Verlag, Heidelberg.

[18] I.-S. O. Young-Won Kim, Kyung-Ae Moon. A text watermarking algorithm based on word classification and inter-word space statistics. In *Conference on Document Analysis and Recognition (ICDAR03)*, 1995.

[19] W.-T. H. Yuei-Lin Chiang, Lu-Ping Chang and W.-C. Chen. Natural language watermarking using semantic substitution for chinese text. In *Proc. of Second International Workshop on Digital Watermarking, IWDW 2002. LNCS*, volume 2939, pages 129–140, Seoul, Korea, October 2003. Springer-Verlag, Heidelberg.