Natural Language Watermarking: Design, Analysis, and a Proof-of-Concept Implementation*

Mikhail J. Atallah[1] and Victor Raskin[2]
with
Michael Crogan[1], Christian Hempelmann[2], Florian Kerschbaum[1], Dina Mohamed[2], Sanket Naik[1]

[1] CERIAS and Dept. of Computer Science, Purdue University, West Lafayette, IN 47906, USA.
{mja,mcrogan,kerschf,sanket}@cerias.purdue.edu
[2] CERIAS, Interdepartmental Program in Linguistics, and Natural Language Processing Lab.
raskin@cerias.purdue.edu, {kikihemp,mohammdh}@purdue.edu

Abstract. We describe a scheme for watermarking natural language text by embedding small portions of the watermark bit string in the syntactic structure of a number of selected sentences in the text, with both the selection and embedding keyed (via quadratic residue) to a large prime number. Meaning-preserving transformations of sentences of the text (e.g., translation to another natural language) cannot damage the watermark. Meaning-modifying transformations have a probability, of damaging the watermark, proportional to the watermark length over the number of sentences. Having the key is all that is required for reading the watermark. The approach is best suited for longish meaning- rather than style-oriented "expository" texts (e.g., reports, directives, manuals, etc.), of which governments and industry produce in abundance and which need protection more frequently than fiction or poetry, which are not so tolerant of the small meaning-preserving syntactic changes that the scheme implements.

1 Introduction

Although Natural Language (NL) watermarking differs from image, video, or software watermarking in that the hidden watermark is embedded in natural language text, the same principles apply: The watermark should be resilient, undetectable to anybody but the author/owner of the text, easily produced by the watermarking software, etc. This paper describes and analyzes a scheme for natural language watermarking, and describes the current state of the prototype

* *Portions of this work were supported by Grant EIA-9903545 from the National Science Foundation, and by sponsors of the Center for Education and Research in Information Assurance and Security.*

implementation. To build this application requires a mix of different techniques, ranging from tree encodings, cryptographic tools, and specially constrained partial natural language analysis and generation. The rest of this section defines the problem and describes our model of the adversary. The next section reviews previous work on NL watermarking, and is followed by a section that describes the ontological semantic approach that is the foundation of our scheme. After that comes a section where we describe our scheme, in its most general version, and analyze its properties; the impatient reader can skip directly to that section–it is written in a fairly self-contained way–but will then miss the details of some crucial ingredients (specifically, those dealing with ontological semantic issues).

1.1 The Problem

Let $\mathcal{T}$ be a natural language text, and let W be a string that is much shorter than $\mathcal{T}$. We wish to generate natural language text $\mathcal{T}'$ such that:

- $\mathcal{T}'$ has essentially the same meaning as $\mathcal{T}$.
- $\mathcal{T}'$ contains W as a secret watermark, and the presence of W would hold up in court (e.g., W could say, "This is the Property of X, and was licensed to Y on date Z"); note that this means that the probability of a "false positive" should be extremely small (recall that a false positive is when the watermark text occurs randomly, i.e., even though it was never inserted).
- The watermark W is not readable from $\mathcal{T}'$ without knowledge of the secret key that was used to introduce W.
- For someone who knows the secret key, W can be obtained from $\mathcal{T}'$ without knowledge of $\mathcal{T}$ (so there is no need to permanently store the original, non-watermarked copy of the text).
- Unless someone knows the secret key, W is very hard to remove from $\mathcal{T}'$ without drastically changing its meaning.
- The process by which W is introduced into $\mathcal{T}$ to obtain $\mathcal{T}'$ is not secret, rather, it is the secret key that gives the scheme its security.
- There is built-in resistance to collusion by two people who have differently watermarked versions of the same text. That is, suppose person A has $\mathcal{T}'_{\mathcal{A}}$, where W_A is hidden using a key that is not known to A, and person B has $\mathcal{T}'_{\mathcal{B}}$, where W_B is hidden using a key that is not known to B, then even if A and B were to share all the information they have they would not be able to either read or delete the watermark (from either $\mathcal{T}'_{\mathcal{A}}$ or $\mathcal{T}'_{\mathcal{B}}$).

The solution we later sketch will satisfy, to a degree that is quantified later in the paper, all but the last of the above requirements, although one modification to the scheme brings it somewhat closer to also satisfying that last requirement.

1.2 Model of Adversary

The adversary is interested in damaging (ideally, destroying) the watermark without drastically changing the meaning of the natural language text in which it is hidden. For this purpose, the adversary is allowed to:

- Perform meaning-preserving transformations on sentences (including translation to another language).
- Perform meaning-modifying transformations on sentences (but note that this cannot be applied to too many sentences, because of the requirement that the overall meaning of the text should not be destroyed).
- Insert new sentences in the text.
- Move sentences from one place of the text to another (including moving whole paragraphs, sections, chapters, etc.).

2 State of the Art in NL Watermarking

Many techniques have been proposed for watermarking multimedia documents because, of course, most of the research in watermarking has focused on image, video, and audio sources (see, for instance, [1–6]). Some of the most successful methods operate in the frequency domain, i.e., on the Fourier or Discrete Cosine transform of an image or audio document (see [7, 8] and the papers referenced there). Such methods do not work on text unless the text is represented as a bitmap image. The problem with that is that "[u]nlike noisy data, written text contains less redundant information which could be used for secret communication" ([9, p. 36]). To apply the watermarking messages developed for images, some features of the text format have to be manipulated: these may include spaces between the letters (kerning), between the words (e.g., proportional spacing) or between the lines (see, for instance, [10–12]) as well as manipulating such text formats as HTML, LaTeX, or Postscript, and their parameters ([9, pp. 36-37]).

The alternative is to develop different methods for text watermarking–those that can embed the watermark in the text itself and are thus unique for natural language texts. Primitive methods, such as inserting spelling and/or punctuation peculiarities and/or deliberate errors and/or synonym substitutions, while still being used, turn out to be not very effective: besides degrading the text, they are even less resilient than our own initial approach [13] was, where we applied Atallah's and Wagstaff's use of quadratic residues [14] to the ASCII number corresponding to each word of the text, thus making it carry a bit of the watermark and necessitating lexical synonym substitutions (of words or even phrases). While our approach was automatic, not manual as its predecessors in the substitution business, and did not degrade the text much, and while we worked to enhance its robustness by embedding the watermark in a small discontinuous portion of the text, it still fell short of our requirements.

The literature seems to favor automatic methods for creating cover texts for secret messages with the help of such primitive techniques as mimic functions [15, 16], which generate statistically correct but meaningless and thus conspicuous texts (for humans but not necessarily for machines sensitive only to the statistical parameters of text) or, more sophisticatedly, context-free grammars (ibid., cf. [17]) that generate primitive but less conspicuously meaningless texts, in which each individual sentence may be sort of meaningful, even if rather unrealistically simple, that hide the secret message in the grammar code rather than in the text. Our present approach implemented in this paper and its further developments shares two desirable features with this latter approach–its automaticity and encoding not the text itself but rather a representation of it–and take them much further. Ours has never, however, been a cover-text approach.

As already mentioned above, the NL watermarking techniques developed earlier were not very resilient. Thus, for the spacing techniques, the watermark can easily be erased by using OCR (optical character recognition) to change the representation of the text from a bitmap to ASCII. Multiple meaning-preserving substitution attacks compromise many existing techniques for storing a watermark in the texts themselves, including, of course, our own initial approach. We would like to come up with a method of watermarking natural language text that is as successful as the frequency-domain methods (such as [7] and related work) have been for image and audio.

Evaluations of watermarking techniques remain a yet unattained goal even in audio, image, and video work [18, 19]; there is definitely no benchmarking available for NL watermarking. We attempt to contribute to the development of NL watermarking evaluation by suggesting methods that significantly raise the ante for the attacker.

3 The Ontological Semantic Implementation

3.1 Why NLP?

The goal of NLP is to develop systems which process texts in natural language automatically for a specific application, such as machine translation, information retrieval, summarization, information extraction, data mining, or intelligent searches. To be successful such systems should either emulate understanding by somehow going around it syntactically or statistically or–much more promisingly–representing and manipulating meaning explicitly. Meaning-based NLP, of which ontological semantics is the best developed and most successful approach, is, in an important sense, a steganographic deciphering system because natural language is a system, often rather arcane and oblique, of encoding meaning. Thus, two English sentences, *John is easy to please* and *John is eager to please*, look identical on the surface but describe very different events: in the former, somebody pleases John and it is easy to do; in the latter, it is John who pleases somebody and is eager to do so.

Be it due to syntactic syncretism, as in the two examples above, or to homonymy, ambiguity, or ellipsis, the main problem of automatic text analysis is to discover the meaning from and under the surface of the text. NLP has accumulated a great deal of experience in unhiding meaning information. It makes perfect sense, then, to try and utilize those resources, reversing them, as it were, for hiding information as required by watermarking, steganography, fingerprinting, and possibly other future assurance and security information.

It should be noted also that the formal resources developed and used by NLP in general and ontological semantics in particular may look similar to the formal mathematical objects referred to as context-free grammars in [15–17] but they are, in fact, very different. That context-free grammars can be interpreted with NL words is, actually, incidental to the nature of these grammars, which admit any number of non-linguistic applications as well, and the interpreted sentences generated by these grammars are primitive, crude, and often ungrammatical. The formal resources, including "grammars," developed by NLP strive to represent each appropriate level of language structure accurately, and while, formally, these resources may look somewhat similar to context-sensitive grammars, they are not interpretable meaningfully by anything other than NL. Besides, most rules in NLP must be context-sensitive. To put it bluntly, we are dealing with much more adequate representations of NL.

3.2 Resources of Ontological Semantics

The ontological semantic approach [20] uses three major static resources, the ontology (O) [20, 21], the lexicon (L) [20, 22–24], and the text-meaning representation (*TMR*) language [20, 25], each of which is defined formally in the BNF representation [20] that can be seen as the formal syntax for each of the resources. The dynamic resources in ontological semantics include a syntactic parser (P), an analyzer (A), and a generator (G).

The ontology is a tangled hierarchy of conceptual nodes, $O = \{o\}$, with each node being a pair of a node label and a set of property and filler pairs, $o = (nl, \{(p_i, f_i)\})$, such that $\forall p \forall f \exists o_m \exists o_n (nl_m = p \& nl_n = f)$, in other words, every property and filler name is an ontological node in its own right. o_{inform}, then, is INFORM(AGENT HUMAN)(THEME EVENT)(BENEFICIARY HUMAN) or

```
(inform
  (agent human)
  (theme event)
  (beneficiary human))
```

The lexicon is a set of lexical entries, $L = \{l\}$, whose meanings are explained in terms of an ontological concept and/or its property. Each lexical entry, l, is a pair consisting of a word (w) and a set of lexicon zones (z), each of which contains a different type of information about the word: $l = (w, \{z_i\})$. When $i = sem$, the semantic zone consists of standard expressions over ontological items: $z_i = E(o_1, o_2, \ldots, o_n)$, where each o is, of course, an ontological node. The TMR of a sentence is an ontological representation of its meaning in the syntax defined by the TMR BNF. Figures 1 and 2 show an example for S_1 = *Mary said to John that she was driving to Boston.*

For each sentence in the ontological semantic approach, the syntactic parser determines the syntactic structure: $P(S) = SynStruc_S$ and the analyzer determines its TMR: $A(S) = TMR_S$. The generator, incorporating a reverse syntactic parser of sorts, generates a sentence for each TMR: $G(TMR) = S$. See `http://crl.nmsu.edu/Research/Projects` for more on ontological semantics and its resources.

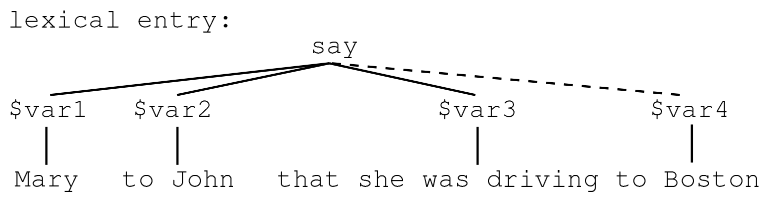

Fig. 1. One of the lexical entries for say, l_{say-1} = INFORM((AGENT ^$\$var1$) (THEME ^$\$var2$) (BENEFICIARY ^$\$var3$)), where ^$\$varN$ is the meaning of the syntactic variables identified for $say - 1$ in its syntactic zone, z_{syn}, as per this very simplified tree for S_1.

```
ontological concept:

(inform
   (Agent ^$var1)
   (Theme ^$var3
       (Agent ^$var1)
       (Destination ^$var4))
   (Beneficiary ^$var2))
```

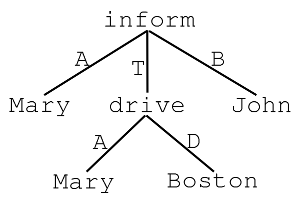

Fig. 2. $TMR(S_1)$ in much simplified form

3.3 The use of NLP Resources for NL Watermarking

The main emphasis in adapting the available resources of ontological semantics for watermarking purposes is to develop an innovative approach for partial use of the ontological semantic approach for selective TMR representation, TMR-lite, as it were, both in terms of accounting for just parts of a sentence, such as a word or a phrase, and by limiting the power and/or grain size of the representations.

Curiously, the NLP problem in NL watermarking is similar to MT in that it is essentially that of translation of one language entity into another on the basis of meaning identity. In MT, the language entities are sentences in two different natural languages. In NL watermarking, they are words or phrases in the same language, which, for the purposes of this research, will be strictly English. In MT, the translation covers the entire text; in NL watermarking, it applies only to the selected words or sentences.

The word, phrase, syntactic structure or TMR that needs to be replaced because it does not satisfy some non-linguistic condition (such as not yielding the required secret bitstring fragment) is:

- automatically detected as needing to be replaced;
- automatically represented syntactically and/or semantically; and
- automatically replaced by an equivalent or near-equivalent entity satisfying a non-linguistic condition: such alternative entity is sought and found automatically as well.

3.4 Meaning-Preserving and Near-Meaning Preserving Text Substitutions for NL Watermarking

For the rejected initial approach [13], it was primarily synonym substitutions. Let us assume, for instance, that the word *freedom* in *America is a society based on freedom* gave the wrong quadratic residue. The system attempted then to replace *freedom* with *liberty.* If this did not produce the necessary bit, it sought replacements for the whole phrase *based on freedom*, such as *which is free*, that *is freedom-loving*, etc. As an extreme measure, it was supposed to replace the whole sentence but was never extended that far. For the syntactic-tree-based approach proposed here, the sentence needs to be manipulated so that it change its syntactic structure without any or any essential change of meaning.

The Parser Instead of writing our own parser, we chose to use one of the many available on the Web. Ironically, most of them were too "sophisticated" for our purposes in the sense that they carried much ballast of the overall systems within which they were developed and yielded information that had no value outside of them. We work with the Link parser (`http://bobo.link.cs.cmu.edu/`), which is not flawless but a little faster, relates its parses to standard syntactic trees of the UPenn treebank, and also has clean documentation. Some of the more serious "ballast" problems had to be cushioned by additional programming, others could be ignored.

Below is Link's actual parse of the sentence *the dog chased the cat*, as a constituent tree:

```
(S (NP the dog)
 (VP chased
     (NP the cat)))
```

This kind of constituent tree is the desired output: It is both easily translated into a bit sequence with the mathematic procedure chosen, as well as provides a transparent interface for the implementation of transformations to manipulate the syntactic structure.

Transformations If a selected sentence does not yield the bit(s) we need it to yield we attempt to generate the correct bit sequence by transforming the sentence without any serious meaning change. We try to use some standard transformations described in post-Chomskian generative syntax. There are few but very productive syntactic transformations that change the structure of a sentence while preserving its overall meaning. Among these the most widely applicable ones appear to be adjunct movement, clefting, and passive formation. Note that to each transformations, there is a similar inverse operation that can be applied to sentences that are using the syntactic construct already, e.g. activization is the inverse of passivization.
Adjunct Movement. In contrast to a complement, an adjunct, like a prepositional phrase or adverbial phrase, can occupy several well-definable positions in

a sentence. For example, the adverbial phrase *often* can be inserted in any of the positions marked by [ADVP] for Adverbial Phrase, and when originally found in one of these, can be moved to any of the others (the simplistic formalism is that of the parser we used):

```
(S (ADVP often)
 (S (NP the dog)
    (VP (ADVP often)
        chased
        (NP the cat)
        (ADVP often))))
```

Clefting. Clefting can most easily be applied to the mandatory subject of a sentence:

```
(S (NP it)
 (VP was
     (NP (NP the dog)
         (SBAR (WHNP that)
               (S (VP chased
                      (NP the cat)))))))
```

Passivization. Any sentence with a transitive verb can be passivized. Identifying the syntactic structure of such a sentence is simple even in the output of a very basic syntactic parser. A transitive verb has a subject [NP1] and an object [NP2] which is the complement that occupies the sister-node of the verb. Ignoring factors like tense, aspect, number, and modal auxiliaries, which are easily implemented, the passive sentence generated out of this input is

```
(S (NP the cat)
 (VP was
     (VP chased
         (PP by
             (NP the dog)))))
```

A change to the syntax of a sentence can also be achieved through sentence-initial insertion of semantically empty "transitional" phrases like *generally speaking*, *basically*, or *it seems that.*

```
(S (NP it)
 (VP seems
     (SBAR that
           (S (NP the dog)
              (VP chased
                  (NP the cat))))))
```

Should the application of all transformation not yield the desired change in the syntactic structure, they can be also applied in all possible combinations.

```
(S (NP It)
 (VP seems
     (SBAR that
           (S (NP it)
              (VP was
                  (NP (NP the cat)
                      (SBAR (WHNP that)
                            (S (VP was
                                   (VP (ADVP often)
                                       chased
                                       (PP by
                                           (NP the dog)))))))))))
```

Text Selection After running different texts through Link in order to adjust our formalism of the transformations to the output of the parser, we chose a government document as the sample corpus to prove our concept on. Our selection from the manual for "Nuclear Weapon Accident Response Procedures" (`http://web7.whs.osd.mil/html/31508m.htm`) has a sufficient length (5980 words) and a typical distribution of syntactic structures. This is the text on which the syntactic-tree-based system has been implemented.

4 Description and Analysis of our Scheme

In this section we describe the overall design in its most general form–the way the prototype will ultimately be–while pointing out how the current prototype differs from that. The secret key that is used to insert the watermark, and to later read it from the watermarked text, is a large prime p. Before we describe how it is used, we need some terminology.

Let the text to be watermarked consist of n sentences $s_1, \ldots, s_n$. To each sentence s_i corresponds a tree T_i that represents s_i either syntactically, as in the current prototype implementation, or semantically as in its future implementation. To each such tree T_i corresponds a binary string B_i that is obtained as follows (where H is a hash function, and the labels of T_i's nodes are ignored):

1. Give the nodes of T_i numbers according to a pre-order traversal of that tree (so the root gets 1, the root's leftmost child gets 2, etc.).
2. Replace every number i at a node by a bit: 1 if $i + H(p)$ is a quadratic residue modulo p, 0 otherwise.
3. B_i is a listing of the above-obtained bits at the nodes of T_i, according to a post-order traversal of that tree (so the root's bit is at the end of B_i, the leftmost leaf's bit is at the beginning of B_i, etc.).

Note 1: Of course, tree nodes are usually labeled with strings, and we may in later prototypes involve each tree node's label in the computation of its bit in Step 2. The current prototype does not do this.

Note 2: Alternative definitions for B_i easily come to mind; later in this section, we will revisit this issue and point out advantages of the above definition for B_i.

Let $\hat{B}_i = H(B_i)$ where H is a one-way hash function. Let S be a list of the s_i's, sorted lexicographic according to their $\hat{B}_i$ value. Duplicates (in the sense of having the same $\hat{B}_i$) are eliminated from S and the corresponding sentences are skipped in watermark insertion and reading. We say that s_i *has rank* r in S if it is the rth smallest in S. In what follows a *marker* is a sentence whose successor in the text is used to store bits from the watermark (a marker might itself be used to carry watermark bits, although typically it is not–more on this below).

4.1 Watermark Insertion

To insert the watermark we repeat the following steps 1–3 until we have inserted into the text all the bits of the watermark:

1. We locate a least-ranked sentence in S, call it s_j. If s_j's successor in the text (i.e., s_{j+1}) was chosen as a marker in an earlier iteration of these steps (1)–(3) then we skip s_j, delete it from S, and repeat this Step 1 (we pick the new least-ranked sentence in S, etc.). Let s_{i-1} be the marker sentence that ends up being chosen in this step.
2. We insert the next chunk (say, β bits) of the watermark in the sentence s_i that follows, in the text, the marker sentence s_{i-1} just chosen in Step 1. The watermark bits are stored within B_i by applying transformations (described in earlier sections) to s_i until (i) the relevant bits of its B_i match the desired values, and (ii) the new rank of s_i is still large enough that it is not mistaken for a marker. What we do in the unlikely case when all such transformations fail to satisfy (i) and (ii) is discussed below.
3. We move the position of the just-modified sentence s_i in the sorted sequence S, so its new position is consistent with its new rank.

Discussion of Step 1 Let α be the number of sentences that carry watermark bits, say, β bits per sentence (so the watermark has $\alpha\beta$ bits). For a long text, it is quite likely (but not needed) that the leftmost α sentences of the initial (yet unmodified) S are the markers; this is because α is much smaller than n and hence it is unlikely that a candidate for marker in Step 1 is rejected. The watermark consists, of course, of the concatenation of $\alpha\beta$ bits stored in the α sentences that are successors of markers (at the rate of β bits per sentence).

Discussion of Step 2 Which portion of B_i is used for the watermark bits is variable, modulated by $\hat{B}_{i-1}$, i.e., it is the (marker) sentence that immediately precedes s_i in the text that determines which bits of B_i carry the β watermark bits. For example, it could be the β bits of B_i that start at position ($\hat{B}_{i-1}$ mod ℓ) for some small integer ℓ, or perhaps the parity of $\hat{B}_{i-1}$ determines whether it is a prefix or a suffix of B_i that holds the watermark bits. The current prototype uses a fixed rule ("prefix of B_i"), but we will use $\hat{B}_{i-1}$ to achieve more variability.

In the current prototype we use $\beta = 1$, but larger values of β are easily achievable by applying more transformations to a target sentence s_i (until its B_i ends up saying what we want it to). If we were applying these transformations randomly and if (hypothetically) each such transformation affected all of B_i, then we would need to apply, on average, $2^{\beta-1}$ transformations to a target sentence before it "confesses". Of course, transformations do not affect all of B_i: some of them affect primarily the right side of s_i's tree T_i (and leave the leftmost bit of B_i unchanged), others affect primarily the left side of T_i, etc. So in practice we would need fewer (respectively, more) transformations than $2^{\beta-1}$ if we applied them in an order that has a higher (respectively, lower) probability of modifying the relevant portion of B_i. For example, if $\beta = 1$ and the watermark bit is the first in B_i, then we would favor those transformations that affect the left side of that sentence's tree T_i, whereas an attacker modifying that sentence has no such information advantage and may use a transformation that affects only the right (and, for this particular s_i, unimportant) side of T_i. Note that this asymmetry in our favor would not exist had we used such an alternative definition of B_i as "B_i is the hash of the concatenation of p with the post-order listing of the pre-order numbers of the nodes of T_i";

The process of modifying s_i (so that its B_i changes) involves making *slightly* meaning-modifying transformations until it has a B_i whose prefix says what we want *and* it has a rank that is still large enough not to be mistaken for a marker. In the (rare) cases where this fails, we insert a new sentence whose meaning "fits in" with the rest of the text and that has these desired properties; the current prototype does not implement such sentence insertions. Slight meaning modifications are preferred to meaning-preserving transformations, because they improve resilience–the adversary does not know which sentences are watermark-carrying and would need to perform widespread meaning changes to damage the watermark, which would defeat the attacker's goal of somewhat preserving the text's overall meaning.

Finally, we note that the probability that (ii) is not satisfied by the new rank of s_i (in Step 2) is approximately α/n and hence is very small (it is the probability that the new rank is less than that of the top-ranked marker sentences). A similar comment holds for the probability of a newly inserted sentence having too small a rank (it is also α/n).

4.2 Watermark Reading

Anyone with the secret prime p can generate the B_i of every sentence s_i, hence its $\hat{B}_i$ and its rank in S. From there on, S is used as in the above watermark-insertion except that no modifications are made to any s_i: we simply read the watermark bits out of each sentence that comes after a marker.

4.3 Resilience

We now have enough information to quantify the effects of various attack actions by an adversary (we assume the general, TMR-based version of our scheme in this discussion). In what follows, we use the fact that the number of sentences used by the watermark is at most 2α (because there are α markers and α successors to markers, with possible overlap between the two sets–although such overlap is unlikely to occur if α/n is very small).

- **Attack 1:** A meaning-preserving transformation of a sentence of the text cannot damage the watermark; examples of such transformations are simple synonym substitutions, sentence restructurings from active to passive, translation to another natural language (e.g., from French to English).
- **Attack 2:** A meaning-modifying transformation of a sentence of the text has probability $\leq 3\alpha/n$ of damaging the watermark: $2\alpha/n$ because there are $t \leq 2\alpha$ markers and successors to markers, and another α/n because a change to one of the other $n - t$ sentences causes its rank to jump to the marker-range with that probability.
  Comment. The above is an upper bound because it assumes that if the adversary has selected a watermark-carrying sentence for modification, that modification will damage the watermark; in reality the modification has probability roughly $1 - 2^{-\beta}$ of damaging the watermark. Similar holds for the estimated success probabilities of the other attacks described below.
- **Attack 3:** The insertion of a new sentence in the text has probability $\leq 2\alpha/n$ of damaging the watermark. This can happen in two ways: (i) if the new sentence's rank is $\leq \alpha$ so it "displaces" a marker in rank; (ii) if the new sentence "separates" a marker from its watermark-carrying successor. The probability of (i) is α/n, that of (ii) is α/n, and therefore the probability of (i) or (ii) is no more than $2\alpha/n$.
- **Attack 4:** Moving a contiguous block of sentences (e.g., a paragraph, section, chapter, etc.) from one place of the text to another has probability $\leq 3\alpha/n$ of damaging the watermark. This can happen in three ways: (i) if the beginning of the block being moved is one of the α sentences that follow a marker; (ii) if the end of the block being moved is one of the α marker sentences; (iii) if the position into which the block is being moved "separates" a marker from its watermark-carrying successor. The probability of (i) is α/n, that of (ii) is α/n, that of (iii) is α/n, and therefore the probability of (i) or (ii) or (iii) is no more than $3\alpha/n$.

All of the above probabilities can be decreased at the cost of more time taken by the watermark-reading process, and a somewhat higher probability of a false positive (more on false positives in the next subsection). While we avoided cluttering the exposition with discussions of such enhancements to our scheme, it is instructive to consider one example of what we mean by this: the probability of a success of case (i) of Attack 3 can be decreased down to approximately $(\alpha/n)^k$ where k is a tunable small positive integer parameter; a larger k complicates watermark-reading by making it take $O(n + \alpha^k)$ time rather than $O(n)$ time, as follows. The watermark-insertion scheme is the same as before, but at watermark-reading time we look at $\alpha + k$ rather than at α "markers", and try all α-sized subsets of these $\alpha + k$ sentences as being "true" markers–one of these

$O(\alpha^k)$ possibilities will yield a meaningful watermark (assuming the watermark bitstring is recognizable as such when one sees it, e.g., because it is in a natural language, or is the hash of an expected text).

4.4 Other Properties

Other properties of our scheme are:

- Having the key p is all that is required for obtaining the watermark from a watermarked text; in particular, it does not require knowledge of the original (pre-watermark) version of the text, or knowledge of the watermark message.
- The probability of a "false positive", i.e., that the text spuriously contains a w-bit watermark, is 2^{-w}. Note that, remarkably, it does not depend on how long the text is (there is no dependence on n). Note, however, that a watermark message that is too short would result in a substantial probability of a false positive, and should therefore be artificially lengthened (for example, by prefixing it with some "dummy" string like *The watermark:*).
- Two holders of differently-watermarked versions of the same text could successfully perform an attack against their watermarks by comparing their copies for differences. This can be made more difficult (but not impossible) at watermark-insertion time by, e.g., making random modifications to a number of sentences that are not used by the watermark (so long as this does not cause the rank of one of these to become smaller than that of a "marker" sentence – if this low-probability event happens then we do not make that change and instead we make another change to this sentence or switch to another sentence).

5 Current Prototype and Planned Extensions

As stated earlier, the current implementation uses syntactic tree structures, whereas the final prototype will use TMR trees; note that much of the software will remain the same because it assumes a tree structure and does not care where it comes from. Writing TMR tree building tools is our next task, followed by implementation of more of the transformations described in Section 3. This will make our life easier in the following sense.

One problem with the syntactic-tree-based approach was that we were not guaranteed the availability of a transformation or chain of transformations which could generate for us a syntactic structure that would yield the required bit(s). Although this is a rare occurrence, we do have a solution for that contingency, and that is the insertion of a new, semantically insubstantial sentence which would have exactly the syntactic structure we need. The syntactic difficulty is thus resolved but a serious semantic challenge is posed: the sentence should fit somehow in between the sentences it is inserted. An interesting challenge from the NLP point of view, it requires for its resolution some considerable semantic information at least about the two bordering existing sentences of the text. We have not implemented this solution, and we are not sure we want to, and here is the reason for that.

If we are going to use TMR information, which is essential for the sentence insertion remedy, we would much rather switch from the syntactic-tree-based approach discussed at length in this paper to the TMR-tree-based approach, which is the direction we have been moving to anyway. In fact, the syntactic approach is seen by us only as a way to train and to fine-tune our mathematical and computational instrumentation, which will work with any tree. Here are some of our main reasons for moving rapidly towards the TMR-based approach:

- the TMR of a sentence is a much more complex tree;
- a larger variety of watermarking techniques may hence be deployed; and most importantly,
- the watermark can be embedded in the top region of the TMR tree, which corresponds to a much coarser-grain meaning, thus making the scheme more resilient to any substitutions: only a substantial change of meaning will endanger the watermark in a sentence, and would have to be applied to many sentences because the adversary does not know which sentences are watermark-carrying. But such wholesale meaning-changes would defeat the attacker's goal of somewhat preserving the text's overall meaning.

To give the reader a feel for this approach, the following URL
`http://www.cerias.purdue.edu/homes/wmnlt/demo/index.php`
contains links to sets of demonstration data. The data was not selected to reflect the full diversity of means at our disposal and was, in fact, deliberately limited to a couple of most visible transformations for meaning-preserving text modification. The samples show (i) text before the watermark is inserted in it, (ii) the (very similar) text after the watermark is inserted, and (iii) the (substantially different) text after the adversary has (unsuccessfully) attacked it by making modifications to sentences and by inserting new sentences.

References

1. Anderson, R. (ed.) 1996. Information Hiding. First International Workshop. Cambridge, UK, May/June 1996. Proceedings. Lecture Notes in Computer Science 1174
2. Aucsmith, D., J. Hartmanis, G. Goos, and J. Van Leeuwen (eds.) 1998. Information Hiding II: 2nd International Workshop, IH '98. Portland, Oregon, USA, April 1998. Proceedings. Lecture Notes in Computer Science 1525.
3. Petitcolas, F. A. P., R. J. Anderson, and M. G. Kuhn 1999. Information Hiding–A Survey. Proceedings of the IEEE 87(7), pp. 1062-1078. July 1999.
4. Pfitzmann, A. (ed.) 2000. Information Hiding. Third International Workshop, IH '99. Dresden, Germany, September/October 1999. Proceedings. Lecture Notes in Computer Science 1768.
5. Katzenbeisser, S., and F. A. P. Petitcolas (eds.) 2000. Information Hiding. Techniques for Steganography and Digital Watermarking.
6. N. F. Johnson, Z. Duric, and S. Jajodia (eds.) 2000. Information Hiding: Steganography and Watermarking - Attacks and Countermeasures. Advances in Information Security, Vol. 1.

7. Cox, I. J., J. Kilian, F. T. Leighton, T. Shamoon 1996. Secure spread spectrum watermarking for images, audio and video. International Conference on Image Processing, Vol. 3, pp. 243-246.
8. Cox, I. J., and M. L. Miller 1996. A review of watermarking and the importance of perceptual modeling. Proc. SPIE - Int. Soc. Opt. Eng., Vol. 3016, pp. 92-99.
9. Katzenbeisser, S. C. 2000. Principles of Steganography. In [5, pp. 17-41].
10. Brassil, J., N. F. Maxemchuk, and L. OǴorman 1994. Electronic Marking and Identification Technique to Discourage Document Copying. Proceedings of INFOCOM '94, pp. 1278-1287.
11. Maxemchuk, N. F. 1994. Electronic Document Distribution. AT&T Technical Journal, September/October 1994, pp. 73-80.
12. Low, S. H., N. F. Maxemchuk, and A. M. Lapone 1998. Document Identification for Copyright Protection Using Centroid Detection. IEEE Transcations on Communication 46(3), pp. 372-383.
13. Atallah, M. J., C. J. McDonough, V. Raskin, and S. Nirenburg 2000. Natural Language Processing for Information Assurance and Security: An Overview and Implementations. In: Preproceedings of the Workshop on New Paradigms in Information Security, Cork, Ireland, September 2000. To appear in: M. Shaeffer (ed.), NSPW '00: Proceedings of Workshop on New Paradigms in Information Security, Cork, Ireland, September 2000. ACM Publications, 2001.
14. Atallah, M. J., and S. S. Wagstaff 1996. Watermarking Data Using Quadratic Residues. Working Paper, Department of Computer Science, Purdue University.
15. Wayner, P. 1992. Mimic Functions. Cryptologia XVI(3), pp. 193-214.
16. Wayner, P. 1995. Strong Theoretical Steganography. Cryptologia XIX(3), 285-299.
17. Chapman, M., and G. Davida 1997. Hiding the Hidden: A Software System for Concealing Ciphertext as Innocuous Text. Proceedings of the International Conference on Information and Communication Security. Lecture Notes in Computer Sciences 1334, pp. 333-345.
18. Kutter, M., and F. A. P. Petitcolas 2000. Fair Evaluation Methods for Watermarking Systems. Journal of Electronic Imaging 9(4), pp. 445-455.
19. Petitcolas, F. A. P. 2000. Watermarking Scheme Evaluation–Algorithms Need Common Benchmarks. IEEE Signal Processing Magazine 17(5), pp. 58-64.
20. Nirenburg, S., and V. Raskin 2001. Principles of Ontological Semantics (forthcoming). Pre-publication draft,
`http://crl.nmsu.edu/Staff.pages/Technical/sergei/book/index-book.html`.
21. Mahesh, K. 1996. Ontology Development for Machine Translation: Ideology and Methodology. Memoranda in Computer and Cognitive Science, MCCS-96-292. Las Cruces, NM, New Mexico State University, Computing Research Laboratory.
22. Nirenburg, S., and V. Raskin 1987. The subworld concept lexicon and the lexicon management system. Computational Linguistics, 13(3-4), pp. 276-289.
23. Nirenburg, S., and V. Raskin 1996. Ten Choices for Lexical Semantics. Memoranda in Computer and Cognitive Science, MCCS-96-304. Las Cruces, NM, New Mexico State University, Computing Research Laboratory.
24. Viegas, E., and V. Raskin 1998. Computational Semantic Lexicon Acquisition: Methodology and Guidelines. Memoranda in Computer and Cognitive Science, MCCS-98-315. Las Cruces, NM, New Mexico State University, Computing Research Laboratory.
25. Onyshkevych, B., and S. Nirenburg 1995. A lexicon for knowledge-based MT. Machine Translation, 10(1-2), pp. 5-57.