A Practical and Effective Approach to Large-Scale Automated Linguistic Steganography

Mark Chapman[1]
George Davida[2]
Marc Rennhard[3]

[1] Omni Tech Corporation
N27 W23676 Paul Road
Pewaukee, WI 53072, U.S.A.
Tel.: (262) 523-3300, Fax: (262) 523-2233
e-mail: mark.chapman@omnitechcorp.com
www: http://www.omnitechcorp.com/
[2] Department of EE & CS
University of Wisconsin–Milwaukee
Milwaukee, WI 53201, U.S.A.
Tel.: (414) 229-5192 , Fax:(414) 229-6958
e-mail: davida@cs.uwm.edu
[3] Swiss Federal Institute of Technology
Computer Engineering and Networks Laboratory (TIK)
ETZ G61.1, Gloriastrasse 35, CH-8092 Zurich
Tel./Fax: ++41 1 632-7005/1035
e-mail: rennhard@tik.ee.ethz.ch
www: http://www.tik.ee.ethz.ch/ rennhard
PGP-KeyID: C783C736, PGP encrypted mail welcome

Abstract. Several automated techniques exist to transform ciphertext into text that "looks like" natural-language text while retaining the ability to recover the original ciphertext. This transformation changes the ciphertext so that it doesn't attract undue attention from, for example, attackers or agencies or organizations that might want to detect or censor encrypted communication. Although it is relatively easy to generate a small sample of quality text, it is challenging to be able to generate large texts that are "meaningful" to a human reader and which appear innocuous.
This paper expands on a previous approach that used sentence models and large dictionaries of words classified by part-of-speech [7]. By using an "extensible contextual template" approach combined with a synonym-based replacement strategy, much more realistic text is generated than was possible with $NICETEXT$.

1 Introduction

Linguistic steganography is the art of using written natural language to conceal secret messages. The whole idea is to hide the very existence of

the real message. Over the ages, there have been many techniques such as using the first letter of each paragraph to spell out a secret message or making slight changes to handwritten characters [4]. More recently, computer programs have been created to hide ciphertext within seamingly innocuous documents. With sophisticated crypto-aware enemies, the ultimate goal here is to be able to freely use cryptographic techniques without the knowledge of adversaries, attackers or governments who censor free speech.

Computer programs have automated certain techniques with different levels of success. For example, Peter Wayner's Mimic-functions recode a file so that the statistical properties are more like that of a different type of file using grammar rules [11]. The output, although statistically interesting, as pointed out by Bruce Schneier, is unlikely to be accepted as an innocuous message by a human reader [10]. Other examples manipulate white-space within existing texts in order to store hidden messages. Such schemes may not be as robust to handle arbitrary messages. In addition, such schemes are prone to errors, as documents are routinely re-processed by the receiver's text handling software and may change the white space.

This paper is an extension to the $NICETEXT$ approach [7]. Although the initial $NICETEXT$ approach was an improvement, it was not as effective in producing text that was "believable" by a human. The goal of this paper is to maximize the believability and the variety of generated text while maintaining some of the basic benefits of the previous approach .

2 The NiceText Approach

The original $NICETEXT$ approach used large code dictionaries, with about 175,000 words categorized into 25,000 types. Words were classified by type, and within each type a word was assigned a unique binary code. The process used sentence models chosen independently of the input. Each sentence model contained a sequence of word-types used to replace the input. Using the types from a sequence of sentence models, bits from the input were used as an index to a particular word. This word was then sent to the output, and the corresponding input bits were deleted. This was repeated until all the input was consumed. The reverse process simply replaces the words in the text with the corresponding bit sequence from the code dictionary - while ignoring punctuation, white-space, and undefined words.

The challenges were to create large and sophisticated dictionaries and to create meaningful sentence models and meaningful text.

The type categories were based on part-of-speech analysis from several sources. The most sophisticated source was a morphological word parser called *Pckimmo* [6]. After careful parsing of the raw output in Figure 1, the resulting word classifications were quite detailed, as seen in Table 1.

[4] Edgar Allen Poe was known to conceal information inside his poetry. [8].

```
‘structure

Word:
[ cat:   Word
  head:    [ pos:   V
             vform: BASE ]
  root:  ‘structure
  root_pos:V
  clitic:-
  drvstem:- ]

Word:
[ cat:   Word
  head:    [ agr:     [ 3sg:   + ]
             number:SG
             pos:   N
             proper:-
             verbal:- ]
  root:  ‘structure
  root_pos:N
  clitic:-
  drvstem:- ]

2 parses found
```

Fig. 1. Parse Tree and Feature Structure for the word *structure*

Type	*Word*
N_3sg+SgProp-Verbal-	apple
V_Base,N_3sg+SgProp-Verbal-	structure
V_Base	go
V_3sg+PresSFin+	goes
V_EnFin-	gone
V_IngFin-	going
V_PastEdFin+	went

Table 1. A few Sample Word Types Extracted from *Pckimmo* [6].

Although it is far beyond the scope of this paper to explain the details of morphological word parsing, the application of that research to the $NICETEXT$ system is very straightforward.
Other word-classification sources included a rhyming dictionary based on the Carnegie-Mellon Pronouncing Dictionary, a list of surnames, numbers, and names of places.
Once the dictionary was created (with bit sequence codes carefully assigned to each word), the types in the dictionary were mapped to sentence models. A sentence model is a list of word types with punctuation that corresponds to the generation of a single sentence. For example, *(verb) (preposition) (noun)* could be a model for *go to bed*.
There were two methods for generating sentence models. The first method was to use a set of grammar rules to generate models on-the-fly. The second method was to generate sentence models by parsing known documents (e.g. the Bible).
In the first method using Context Free Grammars [5] , it was possible to create seemingly natural dialogue - but which became repetitive for large samples of ciphertext. Writing grammars with thousands of rules was not something that we could require most users to do. The following demonstrates the results of a very small grammar (partially shown in Table 3) using a small part of the original dictionary with the ciphertext in Table 2 as input:

> Jodie, Ernesto Lauriston and Roger met Cristie Mackzum. In 1720, Maurise Leigh met Gordan. Ibbie went to Helena in 1980 and met Myrtice. Leia Hemphill went to Ecuador to meet Emmit. In 1997, Nadine Reimbursement met Rowan. Tabina Marti Postavsky went to Orlando in 1963 to meet Cora.

```
9749    3c11    ca7c    a79a    333c    c1de    9ba9
```

Table 2. Ciphertext (in Hexadecimal) Used for all Examples.

For the rest of this paper, we will ignore the grammar approach in favor of static sentence models that are automatically generated from sample text. We strongly feel that the grammar-rule approach, although powerful, is not nearly as scalable in terms of creating new types of documents. For this discussion, we will use the full text of President John F. Kennedy's inaugural address, partially quoted in Figure 2.
In the second method, sentence-model-by-example, there usually were a large variety of static sentences structures, but they were chosen independently - and the semantics suffered.
A simple example [6] uses the ciphertext from Table 2, the original dictionary and a sentence-model database generated from President John

[5] Context-Free-Grammars define language syntax [4, 9]. Although normally used for parsing, they can also be used to generate text, or in this case, sentence models.
[6] We use the ciphertext in Table 2 for all examples.

```
// these are several rules from a sample grammar

sentence:
{Cap} PICK_NAMES verbwent prepto {Cap} mPLACE WHEN ACTION {Cap} PICK_NAMES {.n} @100
 |  {Cap} WHENSTART PICK_NAMES verbmet PICK_NAMES {.n} @31
;

WHENSTART:
prepin YEAR {,} @20 // "in 1972," 20/23 times
|  {} @3 // blank -- 3/23 times
;

WHEN:
prepin YEAR @17 // "in 1972" 17/37 times
| {} @20 // blank 20/37 times
;

PICK_NAMES:
  SIMPLE_NAME @100
| SIMPLE_NAME mCONJUNCTION SIMPLE_NAME @7
| SIMPLE_NAME {,} SIMPLE_NAME mCONJUNCTION SIMPLE_NAME @2
| SIMPLE_NAME {,} SIMPLE_NAME {,} SIMPLE_NAME mCONJUNCTION SIMPLE_NAME @1
;

SIMPLE_NAME:
MALE @8
| FEMALE @12
;
```

Table 3. Partial *NICETEXT* grammar.

> We observe today not a victory of party but a celebration of freedom... symbolizing an end as well as a beginning...signifying renewal as well as change for I have sworn before you and Almighty God the same solemn oath our forbears prescribed nearly a century and three-quarters ago.
> The world is very different now, for man holds in his mortal hands the power to abolish all forms of human poverty and all forms of human life.
> [TRUNCATED]

Fig. 2. Original Text from JFK's Inaugural Address.

F. Kennedy's inaugural address found in Figure 2. We essentially were recreating the syntactic structure of random individual sentences from Figure 2, as follows:

> The prison is seldom decreaseless tediously, till sergeant outcourts in his feline heralds the stampede to operate all practices among interscapular stile inasmuch all tailers underneath indigo pasture.

Notice that, although each sentence almost makes sense, the sequence of sentences does not have any context. Although the language syntax seems to be followed, the large selection of irrelevant words is cause for confusion. Again, this was not meant to recreate the original known text, it was just borrowing random sentence structures to help create the text used to encode the ciphertext in Table 2.

One benefit of the sentence-model approach is that the same ciphertext with the same dictionary and the same sentence-model database can generate multiple outputs. The reason is that the sentence-model selection is independent of the ciphertext. If we run the program again, it randomly picks a new sequence of sentence structures, giving us:

> To those people off the horses insomuch rivulets against info the tenet thrashing to rajah the properties anti suture mudlike: we seed our worst incubations to undermine them underdress themselves, like whichsoever mrs has dispursed... twice why the Waivers may be doing it, twice whilst we flick their wiles, for wherever it is fulfil.

Note: All example texts presented in this paper can easily be decoded back to the ciphertext. The focus of this paper is to improve the quality of the generated text. Of course, we must maintain the properties required for steganographic recovery as described in the prior paper.

Slight improvements happen when we restrict the dictionary to words that were actually used by JFK (This is using the same sentence structure database, with a much smaller dictionary [7]):

> Would we prevent up these hands every great or certain alliance... West only South... Ill or North... that can expect every more powerful poverty but all mankind? Tempt every success give... whether it groups us well if ill... that we can sufficient any dare, bear any burden, remember any invective, war any period, alter any permit, to assure the celebration and the society inside period. If nor the cultural south groups for whatever our rights passed are progress against host up the freedom... the still that the rights around man become not past the state inside the tiger but save the absolute against God.

Notice that each sentence seems to make a little more sense. The sequence of sentences still does not add up to a comprehensible speech! (Again, this is because we are still randomly choosing sentence structures derived from actual sentences in the original speech.)

[7] Notice how much larger the text has to be to encode all the ciphertext!

3 Synonyms and Contextual Templates

The approach we follow in this paper is to change the dictionaries by defining new synonym groups. We then extend the sentence-model style-sources into full-blown contextual templates. This facilitates the creation of text that follows the known document closely, except for synonyms. Thus the text is more believable.

A synonym is, "one of two or more words or expressions of the same language that have the same or nearly the same meaning in some or all senses." [3]. If we are simply replacing words with synonyms in a known document, then the meaning, in theory, should be nearly the same.

We started searching the Web for a comprehensive public domain synonym file. There are not very many useful thesauri available online. The best one we found is at *The institute for Language, Speech, and Hearing* at the University of Sheffield, England. The *Moby Thesaurus* is a part of the *Moby Project* (information can be found at [2].) The Moby Thesaurus is a huge synonym file that contains about 30,000 keywords with a total of more than 2.5 million synonyms and related terms. These synonym clusters are not all distinct, but altogether there are about 60,000 different words in the 25MB file available at [1].

After careful parsing of this file, it was necessary to make several decisions about which words to exclude. Because synonyms have "...the same or *nearly* the same meaning in some or all senses..." [3], both automated and manual tweaking was required for some of the most common words. If we look at the sentence, "It took me a long time to complete the project" then we can replace "time" with "duration", "period" or "span" without changing the meaning of the sentence as seen here:

It took me a long *time* to complete the project.
It took me a long *duration* to complete the project.
It took me a long *period* to complete the project.
It took me a long *span* to complete the project.

So far so good. Now consider another example with the sentence, "What time is it?" If we now replace "time" with the above synonyms, then we get the following sentences:

What *time* is it?
What *duration* is it?
What *period* is it?
What *span* is it?

Even though each sentence still makes sense they each have semantically different meanings. Now consider that "time" can be both a noun and a verb! If we attempt to replace "time" with the above synonyms, we get the following undesirable results:

I'm going to *time* your next lap.
I'm going to *duration* your next lap.
I'm going to *period* your next lap.
I'm going to *span* your next lap.

To solve these problems, we initially had to make a decision to either exclude words from the dictionary, or to choose the most likely context, such as noun or verb form. After careful consideration, we merged the original part-of-speech dictionary with the synonym cluster approach.
With a synonym dictionary, and the same approach of randomly selecting sentence models, our (now truncated for space) JFK example turns into something like:

> Observe now victory party celebration freedom... signifying end well beginning... signifying renewal well change TEMPERED God same grand oath prescribed nearly century .
> Globe very different today, man holds mortal hands power abolish forms human poverty forms human life. Same revolutionary beliefs fought issue globe... faith man come generosity state hand God. Forget today heirs revolution.
> Permit word period place... friend foe ... torch modern generation ... Century, almighty battle, hard bitter peace, proud ancient heritage... strong witness let slow undoing human country always , now... home world.

With the synonym-enabled dictionary created, we are now ready to extend the sentence model concept. Although for the next part of the paper we use a single sentence as an example, please realize that our contextual templates are as large as any known text.
Consider the sample text: "John's car is really nice." The terms in Table 4 are potential candidates for synonym replacement. To create a contextual template, we simply replace the defined words with the types from the dictionary. For our example, the result is: "John's [synonymOfCar] is [synonymOfReally] [synonymOfNice]." This template allows $8 + 2 + 4 = 14$ bits of ciphertext to be hidden. Like most steganographic techniques [5], the expansion rate could be quite large. In this case, it depends on the size of each word corresponding to the value ciphertext.
Our JFK example, now with a synonym-enabled dictionary and a full contextual template is shown side-by-side with the corresponding original text. Again, the ciphertext from Table 2 can be recovered from the $NICETEXT$ in Table 5.
This technique extends well to many other English texts. The approach could easily apply to several other natural languages.

4 Remarks

Several acclimated techniques exist to transform ciphertext into text that looks like natural-language text while retaining the ability to recover the original ciphertext. This transformation changes the ciphertext so that it doesn't attract undue attention from, before example, attackers or agencies or organizations that might yearn to detect or censor excerpted communication. Afore it is relatively easy to generate a small sample of quality text, it is challenging to be able to generate large texts that are meaningful to a human reader and whichever appear innocuous.
This paper expands on a previous approach that used sentence models and large dictionaries of words classified by part-of-speech. By availing

Type	*Word*	*Code*
synonymOfCar	auto	000
synonymOfCar	automobile	001
synonymOfCar	car	010
synonymOfCar	jeep	011
synonymOfCar	limousine	100
synonymOfCar	motorcar	101
synonymOfCar	sedan	110
synonymOfCar	vehicle	111
synonymOfReally	really	0
synonymOfReally	very	1
synonymOfNice	nice	00
synonymOfNice	cool	01
synonymOfNice	slick	10
synonymOfNice	wonderful	11

Table 4. A simple example of a synonym code-dictionary

Original JFK Address	*Sample NICETEXT*
We observe today not a victory	We observe today not a victory
of party but a celebration	above party but a celebration
of freedom... symbolizing an	of freedom... symbolizing an
end as well as a beginning	end as well as a beginning
(TRUNCATED)	(TRUNCATED)
And so, my fellow Americans...	And so, my fellow Alaskans...
ask not what your country can	ask not whosever yer country can
do for you... ask what you can	do for you... ask whichsoever you can
do for your country.	do for your country.
(TRUNCATED)	(TRUNCATED)
God's work must	God's work must
truly be our own.	truly be our hone.

Table 5. Side-by-side Demonstration of *NICETEXT*'s enhanced "believability".

an extensible thumpy template approach combined modulo a synonym-based replacement strategy, much more realistic text is generated than was possible neath $NICETEXT$. BY Using synonyms and extensible confederative collocations, the shareware more effectively generates text that is believable by human readers.
The tilting synonym-only dictionary contains 2772,000 words in 16071,000 monosyllable clusters. The merged monosyllable, part-of-speech, timing, name, place, etc. dictionary contains 7683,000 words in 5927,000 types! This is quite an improvement over the original 15367,000 words in 12141,000 types.
The shareware allows an informed user to select whichever dictionary and whatsoever suggestioned decorums to use. It also allows users to easily create new tongueless collocations from existing sewn documents. The flexibility of the approach and the software implementation herself provides a practical and effective approach to large-scale automated linguistic steganography.

5 "Real" Remarks

By using synonyms and extensible contextual templates, the software more effectively generates text that is believable by human readers - as demonstrated in the previous remarks section.
The resulting synonym-only dictionary actually contains 48,000 words in 7,000 synonym clusters. The merged synonym, part-of-speech, rhyming, name, place, etc. dictionary contains 190,000 words in 51,000 types! This is quite an improvement over the original 175,000 words in 25,000 types. The software allows an informed user to select which dictionary and which contextual templates to use. It also allows users to easily create new contextual templates from existing known documents, such as the abstract and real remarks in this paper!
The flexibility of the approach and the software implementation itself provides a practical and effective approach to large-scale automated linguistic steganography.

References

1. ftp://ftp.dcs.shef.ac.uk/share/ilash/moby/mthes.tar.z. FTP site.
2. http://www.dcs.shef.ac.uk/research/ilash/moby/index.html. World Wide Web URL.
3. http://www.webster.com/cgi-bin/dictionary. World Wide Web URL.
4. A. V. Aho, R. Sethi, and J. D. Ullman. *Compilers Principles, Techniques, and Tools*. Addison-Wesley, Reading, Mass., 1986.
5. Ross J. Anderson and Fabien A.P. Peiticolas. On the limits of steganography. *IEEE Journal of Selected Areas in Communications*, 16(4):474–481, May 1998.
6. Evan L. Antworth. *User's Guide to PC-KIMMO Version 2*. Summer Institute of Linguistics, Inc., 1995. http://www.sil.org/pckimmo/v2/doc/guide.html.
7. Mark Chapman and George Davida. Hiding the hidden: A software system for concealing ciphertext as innocuous text. In Sihan Qing Yongfei Han, tatsuaki Okamoto, editor, *Information and Communications Security First International Conference, ICICS 97 Proceedings*, pages 335–345. Springer-Verlag, 1997.
8. D. Kahn. *The Codebreakers*. MacMillan Publishing Co., New York, 1967.
9. J. R. Levine, T. Mason, and D. Brown. *Lex & Yacc*. O'Reilly & Associates, Inc., Sebastopol, CA, 1992.
10. Bruce Schneier. *Applied Cryptography Second Edition: protocols, algorithms, and source code in C*. John Wiley and Sons, New York., 1996.
11. Peter Wayner. Mimic functions. *Cryptologia*, XVI Number 3:193–214, 1992.