*Journal of Information Hiding and Multimedia Signal Processing* ©2011 ISSN 2073-4212
**Ubiquitous International** Volume **2**, Number **2**, April **2011**

# Free from the Cover Text: A Human-generated Natural Language Approach to Text-based Steganography

Michael Grosvald and C. Orhan Orgun

Department of Linguistics
University of California, Davis
Davis, CA 95616
mgrosvald@ucdavis.edu, ocorgun@ucdavis.edu

Received June 2010; revised October 2010

Abstract. *A common pitfall of existing encryption procedures using lexical (text-based) steganography is the fact that the encrypted text may be recognized as such by someone who intercepts it. We introduce a new procedure which combines an automated algorithm with human input. The resulting texts are novel and therefore not searchable or otherwise easily recognized as encoding a hidden message.*
**Keywords:** steganography, lexical, text-based

1. **Introduction.** This paper introduces Neko, a new approach to linguistic (text-based) steganography that follows up on the work of Chand and Orgun [2] and Orgun and Chand [14], who propose a system based on word replacement called Lunabel. Other word-replacement steganography systems include [1], [3], [4], [5], [15], and [18]. These systems share a common strategy. They start with a piece of natural text, called the cover text. There is a list of words targeted for replacement. Each word replacement encodes a bit of information. Steganography proceeds by identifying replaceable words within the cover text and choosing the appropriate replacement word from the word lists. Compilation of word lists such that replacements with a word from the same list will retain the syntactic and semantic naturalness of the cover text is the main challenge such approaches face.

There are also systems of text-based steganography not based on word replacement. These systems use a variety of approaches, such as translation from one language to another [13], hiding information in errors [17], text summarization [10], lists or notes [9], [11]. There are, of course, non-text-based techniques as well, but we are not concerned with those here.

Vulnerabilities of such systems are discussed in detail in [16]. The major vulnerabilities are syntactic and semantic unnaturalness. This results from the fact that it is well nigh impossible to compile word lists such that replacement of one word in a list with another will leave the cover text unaffected in all cases. Our new method, Neko, seeks to avoid the unnaturalness problem by not using a cover text. Instead, it provides a sequence of words. Human input is then required to create a text that includes those words in the same order that Neko provided them. We show that this cover-text-free approach has the potential to overcome some weaknesses that Lunabel is subject to. In particular, if a cover text is recognizable even after substitutions are performed by Lunabel, someone who intercepts the encrypted message can compare the two versions of the text. Any words that differ

between the two documents will immediately be identifiable as replacements performed in the course of encryption. Neko avoids this potential pitfall by requiring human input in creating the hiding text, which guarantees that the encrypted text will be novel and therefore not searchable or recognizable. As it requires substantial input, Neko is not meant for mass steganography, but rather for small scale applications where linguistic naturalness is paramount.

Section 2 presents a summary of the relevant aspects of Lunabel. Section 3 discusses a minimal modification of Lunabel that leads to the first version of Neko and discusses its advantages and disadvantages with respect to Lunabel. Section 4 introduces a second version of Neko and discusses further advantages specific to this second version. In Section 5, we show short passages of encrypted text generated by Lunabel and Neko. Section 6 presents a statistical analysis of the vulnerability of Lunabel and Neko encryptions to bigram-based attacks. Section 7 concludes the paper.

## 2. **Summary of Lunabel.** Lunabel has several aspects, listed below:

(a) Word lists
(b) Conversion of plain text into a sequence of digits
(c) Replacement of cover text words

We discuss each aspect separately.

### 2.1. **Word lists.**

Any word replacement scheme must specify which words form a substitution class. A substitution class is a set of words such that a given word on the list can be substituted for any other without affecting the linguistic acceptability of the sentence that the word is part of. For example, the sentences *I had eggs for breakfast* and *I had pancakes for breakfast* suggest that the two words eggs and pancakes might be members of a substitution class. By contrast, **I had indicate for breakfast* ("*" marks unacceptable forms) shows that *eggs* and *indicate* are not members of a substitution class. This is of course because *eggs* and *indicate* are from different syntactic categories (also known as parts of speech), namely noun vs. verb. *Eggs* and *pancakes*, on the other hand, are both nouns. Sharing syntactic category is a minimal requirement for belonging in a substitution class. Other criteria are involved as well; the reader is referred to Orgun and Chand ([14]) for discussion. For reasons of programming efficiency and other considerations they discuss, Orgun and Chand use substitution classes of 16 words each. We follow them in this regard, though our system of steganography will work with any size of substitution classes (as will theirs). We present one of the substitution classes of Lunabel as an example:

(1) Substitution class example.
key, passkey, login, password, pincode, pin, passcode, authorization, permission, authentication, signal, ticket, indicator, passport, code, credential

In the style of writing that this particular word list is intended to use as cover text ("readme" files included in software packages), these words can almost always substitute with one another without affecting grammaticality. The number of substitution classes to be used is arbitrary. In principle, a single class will suffice for encryption purposes. However, using a fair number of substitution classes has advantages. The main advantage is that one thereby has a correspondingly large number of words available for encryption purposes. This will reduce the size of cover text needed to hide a plain text of a given length. In the following sections, we will show how these substitution classes are used in the Lunabel system.

2.2. **Conversion into digits.** Before word substitutions can be performed based on the substitution classes, the cover text needs to be converted into a sequence of digits. These digits will then determine which word from a given word list will be used in word substitutions. This section describes the conversion of the cover text into digits.

In Lunabel, conversion is based on ASCII codes. Naturally, any other character coding will work equally well (for example, there is no reason one could not use Unicode). The conversion is quite simple: each character of the plain text is replaced by its ASCII code, expressed as two hexadecimal digits (this is why word lists of 16 elements each were found convenient). The character "a", for example, converts into [1, 6]. The phrase portion *I had* converts into [9 2 0 6 8 6 1 6 4 2]. Once this conversion is performed, word substitutions can be carried out. This is the final step in text-based steganography.

2.3. **Substitution.** The main task in steganography is to hide a message in what appears to be an ordinary piece of text. In Lunabel, as in other replacement-based systems, this is done by taking a cover text and replacing certain words in it by other words. This section describes the procedure for performing these replacements.

The replacement system scans the cover text one word at a time, from its beginning. Words that do not appear in any word list (substitution class) are copied unchanged. Eventually, a word that appears on a substitution class will be encountered. When this happens, this word will be substituted for by another word from its word list. Which word? This is determined by the current digit in the converted plain text. This is described more explicitly below.

The first time a word from a substitution class is encountered, the substitution proceeds as follows (the general case will be discussed afterwards).

- Let $W$ stand for the first cover text word that belongs to a word list.
- Let $L$ stand for the word list that $W$ belongs to.
- Let $E_1, E_2, ...E_{16}$ be the elements of word list $L$ (one of these is $W$).
- Let $D$ stand for the first digit in the converted form of the plain text.

Word $W$ of the cover text is now replaced with the $D^{th}$ element of word list $L$ (that is, with $E_D$).

The general replacement scheme can be described as follows:

Assume that $i-1$ substitutions have already been performed (initially, $i = 1$). Represent the converted plain text (sequence of digits) as follows: $[D_1, D_2, ...D_N]$. Read one word ($W$) of the cover text.

(a) If $W$ is not an element of a word list, copy $W$ unchanged.
(b) If $W$ is an element of word list $L$ with elements $E_n$, then replace $W$ with $E_{D_i}$.
(c) If $i = N$ (the plain text has been fully encrypted), copy the rest of the cover text unchanged.
(d) If there are no words left in the cover text when $i < N$, issue an error message.
(e) If there are words left in the cover text and $i < N$, move to the next word in the cover text and go to step (a).

This method of steganography does not indicate the end of the plain text. In decryption, therefore, the recovered plain text will generally be followed by a series of random characters, corresponding to the hexadecimal digits of the word list elements in the unchanged portion of the cover text (step (c) above). A simple way to deal with this is to append a (any) end-of-message marker to the plain text prior to its conversion into digits.

2.4. **Encryption.** It may be noticed that we have described a mechanism that embeds a plain text directly in a modified cover text. No encryption is carried out beyond the act of hiding. Hiding and encryption can in fact be combined, as discussed in [14]. However,

we do not pursue this possibility. One reason for our choice is that our interest lies firmly in the domain of information hiding. Another reason is that, if desired, the plain text can be encrypted by any desired method (of any desired strength) prior to the hiding steps we present. The encrypted message can then be hidden by following the exact same steps. In a real life application, this may very well be the method of choice; by following this method, a second layer of security is achieved in case the fact that the two parties are exchanging secret messages is found out and the hidden message is revealed. Steganography has, in such a scenario, failed, but the message still remains encrypted.

3. **Losing the cover text.** Using a cover text and replacement scheme has certain disadvantages. First, we discuss those disadvantages. Then, we describe Neko, which avoids these by doing away with the cover text.

3.1. **Disadvantages of cover text use.** Cover text systems have at least three disadvantages. First, there is the issue of what text to use. If the cover text is an existing text (a section of a book, a file from some software installation, web content, etc.), then there is a risk that this may be recognized by someone who intercepts the steganographic message (e.g. by performing an Internet search on a passage from within the document). This will cause two problems. First, the fact that the cover text is recognizable but different from its original may immediately arouse suspicion and lead someone to start to analyze the message. This occurrence will already have defeated the purpose of steganography, which is to avoid detection of encrypted messages as such. Steganography instead aims to mask messages as innocuous text.

Second, by comparing the original cover text to its intercepted version, one can identify words that have been substituted. This provides the interceptor with information about not just the presently intercepted message, but the scheme of information hiding as well (at the very least, they will know that certain words are substitution class members).

These problems can be avoided by writing a novel cover text for each transmission. This of course requires human input. The problem with systems like Lunabel is that one has to trust the user to put in the required effort to create new cover texts. However, there is a risk that an individual user might decide to take a shortcut and use an existing text, or perhaps even use the same cover text repeatedly.

The second problem is that even carefully crafted substitution classes may not always give rise to natural text. This is because there might be specialized contexts in which the substitution does not work. For example, going back to our earlier forms *eggs* and *pancakes*, we can see that the sentence *Break two eggs* will not work when it is transformed into *Break two pancakes*.

Finally, it is desirable that words that are on the same word list have similar frequencies of occurrence. That way, word frequencies in the encrypted text will not be too different from natural text. Even then, n-gram frequencies will likely differ (e.g. see [6], [7], [8], [16]); for example, even if "up" and "down" have comparable frequencies, the bigrams (sequences of two words) "make up" and "make down" do not. The problem is, finding substitution classes is hard enough even without attention to token frequencies. It becomes even harder when one wants to take frequencies into account.

3.2. **Neko.** In the previous section, we argued that using an existing cover text is dangerous. Instead, relatively secure steganography demands a novel cover text for each message transmission. One obvious way to achieve that is to have a human compose a new cover text each time. Our system of steganography requires human input in creating a post-hoc cover text after the information hiding task has been carried out automatically. We will now show how this can be done.

We use the same conversion-to-digits system and word lists as Lunabel. In encryption, however, we do not use a cover text. Instead, we directly employ the word lists, as follows (the symbols used have the same meanings as before).

Assume $i-1$ digits from the converted plain text have already been processed.

(a) Read the $i^{th}$ digit of the converted plain text, $D_i$.
(b) Choose a random word list, $W$.
(c) Write element $L_{Di}$ of $W$ in a file.
(d) If the plain text has been exhausted, stop.
(e) Otherwise, increment $i$ and go to step (a).

Once this procedure is carried out, one has a sequence of words. The sequence might, by way of example, look something like:

(2)Neko-generated word list example.

passkey upload monitor...

The next step is for a human being to manually enter text around these words to create some natural looking text. For example, one might complete the word sequence in (2) as follows.

(3) Neko text completion example.

Obtain a passkey from your system administrator. Enter it into the appropriate box. Upload your files to the desired directory. When working with large files, you can turn off your monitor to save energy...

Care must be taken to make sure the additional words entered in this process do not belong to any of the word lists used in steps (a)–(e). Otherwise decryption will fail, due to additional (unwanted) elements being added to the numerical sequence representing the desired character string. This can easily be facilitated by using a text editor that automatically highlights word list elements when they are typed.

Obviously, this system is more labor-intensive than taking an existing cover text and automatically substituting words in it. However, we have already argued that the latter procedure is risky. Our system is not that much more demanding of human input than a relatively safe use of a replacement system would be. Both systems require human input (in replacement systems, for security, in Neko, by design). The only additional demand Neko places on a human being is the ability to write a text around a given sequence of words. While this does call for a modicum of creativity, we judge the task to be not at all excessively difficult.

4. **Freedom from substitution classes.** We discussed another weakness of word replacement schemes, namely the conflicting demands of good substitution classes and controlling for token frequencies within a word list.

Consider once again the mechanism for cover text creation in Neko. The cover text is created by human input only after the information hiding step is automatically performed. The word lists are used solely for the purpose of encoding the converted digits of the plain text. Therefore, they do not need to form a linguistic substitution class. This frees us to compile word lists based solely on frequency of occurrence. One simply needs word counts from a corpus of text (easily available; equally easy to compile). The next step is simply to collect 16 words that have similar frequencies and place them in a substitution class.

This allows a further refinement. Referring back to the Neko algorithm, we see that step (b) calls for a random word list. By weighting the word lists according to the token frequencies of their elements, we can assure that the words will appear in the cover text at about their proper frequencies.

This leads to one final advantage. Any word replacement scheme is vulnerable to n-gram analysis. But in our system, a human being composes the final text. It is reasonable to assume therefore that the resulting text will have appropriate collocations at reasonable frequencies.

In the next section, we illustrate typical results of Lunabel (word replacement in cover texts) and Neko (human-generated text surrounding a word list). In section 6, we perform a statistical analysis comparing bigram frequencies between Neko and Lunabel, relative to natural (cover) text.

## 5. Example texts.

The following three sub-sections give excerpts of a cover text as well as Lunabel and Neko texts. The Lunabel text differs from the cover text only where word list elements have been replaced in order to code the message. Such word list elements have been highlighted. The Neko text uses the same word list elements as the Lunabel text, but for this encryption method, a human author composed the surrounding text. It can be seen from these illustrative examples that the Lunabel method has produced text which is semantically odd in a number of places, whereas the Neko method enables one to avoid this problem.

### 5.1. Natural (Cover) text.

They reigned over the earth for more than 100 million **years** and suddenly, mysteriously disappeared. What caused the demise of this ubiquitous **group** of reptiles which included some of the largest animals to ever walk the planet? One of the great mysteries in science is the extinction of the dinosaurs at the end of the Mesozoic Era some 65 million **years** ago. Who (or more likely what) caused it is unknown and a **subject** of great debate. Dinosaurs appeared at the beginning of the Mesozoic Era and were the dominant **form** of **life** until the end of that era. They lived almost everywhere there was land including Antarctica. We can see their bones in the geological record. The lower stratum of rock contains the earliest and most primitive species of dinosaur, and the upper stratum contains the newer species. Then, suddenly, at a geological strata **line** called the Cretaceous-Tertiary Boundary (often referred to as the K-T Boundary), the dinosaurs disappear...

### 5.2. Lunabel text.

They reigned over the earth for more than 100 million **times** and suddenly, mysteriously disappeared. What caused the demise of this ubiquitous **universe** of reptiles which included some of the largest animals to ever walk the planet? One of the great mysteries in science is the extinction of the dinosaurs at the end of the Mesozoic Era some 65 million **things** ago. Who (or more likely what) caused it is unknown and a **characteristic** of great debate. Dinosaurs appeared at the beginning of the Mesozoic Era and were the dominant **sense** of **question** until the end of that era. They lived almost everywhere there was land including Antarctica. We can see their bones in the geological record. The lower stratum of rock contains the earliest and most primitive species of dinosaur, and the upper stratum contains the newer species. Then, suddenly, at a geological strata **context** called the Cretaceous-Tertiary Boundary (often referred to as the K-T Boundary), the dinosaurs disappear...

5.3. **Neko text.**

> One of the greatest scientific achievements of modern **times** is the great leap in our understanding of the large-scale temporal and spatial structure of the **universe**. Human beings are by nature very curious, and it has often been said that this curiosity is one of the **things** that defines us. This trait that is so **characteristic** of our species has allowed us, by extension, to gain a greater **sense** not only of where we live, but who we are as well. However, one key **question** that science has yet to answer is whether we are alone in this great cosmos. If we find that other beings have also attained consciousness and intelligence as we have, can we still consider ourselves unique? It is in the **context** of such a discussion that one is forced to examine possibilities which at first glance may make us uncomfortable, but which ultimately must be investigated if we are to continue to pursue such scientific endeavors...

6. **Statistics: Bigram analysis.** It has been noted in the literature that steganographic texts may be particularly vulnerable to n-gram analysis ([6], [7], [8], [16]). While it is relatively easy to match single word frequencies to those found in natural text, it is much more challenging to do the same for word pair (and n-tuplet) frequencies. Accordingly, we have chosen bigram frequencies as our criterion for comparison of natural, Lunabel, and Neko texts.

Since Neko is designed for low-volume application, a brute force count of frequencies is unlikely to yield statistically valid results. Therefore, we have decided to use an indirect method that compares the number of novel bigrams that occur in new texts. We compiled an ad-hoc corpus for this purpose, which includes roughly equal numbers of words in three types of texts: geological history, accounts of species extinctions, and free-form narratives by individuals. These text types match the genres of the Lunabel and Neko texts that we used to perform this statistical analysis. The corpus has a total size of approximately 100,000 words. This has proven large enough to provide sufficiently diverse bigrams, yet small enough that a new piece of text is likely to have a number of "novel bigrams"; that is, bigrams appearing in the new text but not in the corpus. Our analysis seeks to determine whether texts encrypted using Lunabel or Neko contain an unusually large or small number of novel bigrams relative to natural text.

For this analysis, three texts representing each of the three encryption styles (Cover text, as well as Lunabel and Neko; i.e. nine texts total) were prepared. The three cover texts were the ones used to create the three Lunabel encryptions using word lists as described above; the Neko texts also incorporated words taken from those same word lists. Of course, "Cover text" is not an encryption style per se but was included so that comparisons with a baseline condition would be possible. To evaluate a given text, we tallied "novel bigram counts" for each word appearing both in the corpus and in an encryption word list. That is, we first collected all of the bigrams in the text that were absent from the corpus, and which contained words belonging to one of the encryption word lists. For each such word, we then summed the number of cases in which the word appeared as a member of a novel bigram. This permits us to determine the relative likelihood that an encrypted text would contain a large or small number of novel bigrams, in comparison with unencrypted (natural) text. To determine whether such differences among encryption styles were significant, we performed a one-way analysis of variance (ANOVA) on the resulting dataset, with factor Encryption Style (Cover, Lunabel, or Neko). There was no significant effect of Encryption Style ($F(2, 700)=1.992$, $MSE=0.291$, $p>0.10$). Follow-up comparisons found that means were somewhat higher for the encrypted texts than for natural text (means [standard errors] for Cover, Lunabel and Neko respectively were: 0.215 [0.030], 0.295

[0.030], 0.263 [0.027]), but the only pairwise difference that was significant here was the Cover vs. Lunabel comparison ($p<0.05$).

These results indicate that Lunabel encryptions have a significant tendency toward higher novel bigrams counts than does natural text, while Neko does not. In particular, we find that Neko is not more vulnerable to bigram analysis than a purely algorithmic word replacement method.

7. **Conclusion.** Many linguistic steganography systems start with an existing cover text and then modify this cover text in order to hide a message in it. This modification gives rise to certain vulnerabilities. One type of vulnerability is that the modified text is identifiable as unnatural by human readers. Another, more subtle, vulnerability is that the statistical properties of the resulting text may be measurably different from those of natural human language. In this paper, we have proposed a linguistic steganography system that overcomes these problems. Instead of starting with a cover text and replacing words in it to convey information, our method starts with a list of words that encode a plain text. It then calls for a human being to compose a cover text including these words. Since the cover text is written by a human in its final form, it is by definition, from a linguistic point of view, natural language. As such, it does not appear suspicious to a human observer. Therefore, we conclude that this approach, in addition to comparing favorably to word replacement techniques vis a vis statistical analysis, solves the problem of "the human attack in linguistic steganography" ([14]).

## REFERENCES


[1] Bergmair, R. 2004. Towards linguistic steganography: A systematic investigation of approaches, systems, and issues. Tech. rep., University of Derby, August 2004.

[2] Chand, V. and C. O. Orgun. 2006. Exploiting linguistic features in lexical steganography: Design and proof-of-concept implementation. *Proceedings of the 39th Annual Hawaii International Conference on System Sciences*, vol. 6. IEEE: New York.

[3] Chapman, M. and G. Davida. 1997. Hiding the hidden: A software system for concealing ciphertext in innocuous text. *Proceedings of the International Conference on Information and Communications Security*, vol. LNCS 133, Beijing, China.

[4] Chapman, M. and G. Davida. 2002. Plausible deniability using automated linguistic stegonagraphy. *Proceedings of the International Conference on Infrastructure Security*, October 1-3 2002, Bristol, UK, 276-287.

[5] Chapman, M. et al., 2001. A practical and effective approach to large-scale automated linguistic steganography. *Proceedings of the information security conference* (ISC 01), Malaga, Spain, 156-165.

[6] Chen Zhi-li, Huang Liu-sheng, Yu Zhen-shan, Li Ling-jun, and Yang Wei. 2008a. A Statistical Algorithm for Linguistic Steganography Detection Based on Distribution of Words. *Third International Conference on Availability Reliability and Security*, no. 60773032.

[7] Chen Zhi-li, Huang Liu-sheng, Yu Zhen-shan, Zhao Xin-xin, and Zheng Xue-ling. 2008b. Effective Linguistic Steganography Detection. *CIT Workshops Australia*, no. 60773032.

[8] Chen, Z., L. Huang, Z. Yu, W. Yang, L. Li, X. Zheng, and X. Zhao. 2008c. Linguistic Steganography Detection Using Statistical Characteristics of Correlations between Words. *Information Hiding 10th International Workshop 2008, Santa Barbara CA USA May 19-21 2008; Revised Selected Papers*: 224.

[9] Desoky, A. et al. 2008. Auto-Summarization-Based Steganography, *Proceedings of the 5th IEEE International Conference on Innovations in Information Technology*, Al-Ain, UAE, December 2008.

[10] Desoky, A. 2009a. Listega: List-Based Steganography Methodology, *International Journal of Information Security*, Springer-Verlag, April 2009.

[11] A. Desoky, Notestega: Notes-Based Steganography Methodology, *Information Security Journal: A Global Perspective*, Vol. 18, No 4, pp. 178-193, January 2009.

[12] Desoky, A. 2010. NORMALS: Normal Linguistic Steganography Methodology. *Journal of Information Hiding and Multimedia Signal Processing*, 1(3), 145-171.

[13] C. Grothoff, et al. 2005. Translation-based steganography, *Technical Report CSD*, TR no. 05-009, Purdue University. (CERIAS Tech Report 2005-39)

[14] Orgun, C. O. and V. Chand. 2009. The Human Attack in Linguistic Steganography. M. Gupta and R. Sharman (Eds.), *Handbook of Research on Social and Organizational Liabilities in Information Security*, 380-397. New York: IGI Global. doi: 10.4018/978-1-60566-132-2.

[15] Shirali-Shahreza, M. et al., 2007. Text steganography in SMS. *International conference on convergence information technology*. Volume 21.23, 2260-2265.

[16] Taskiran, C. M., Topkara, U., Topkara, M. and E. Delp. 2006. Attacks on lexical natural language steganography systems. *Proceedings of the SPIE International Conference on Security, Steganography, and Watermarking of Multimedia Contents*.

[17] Topkara, Mercan, Umut Topkara and Mikhail J. Atallah. 2007. Information hiding through errors: A confusing approach, *Proceedings of the SPIE International Conference on Security, Steganography,and Watermarking of Multimedia Contents*, January 2007.

[18] Winstein, Keith. No date. The tyrannosaurus Lex system, available at: http://alumni.imsa.edu/ keithw/tlex/