Steganography

The art of hiding a message in plain sight

AMELIA ANDERSSON
and LINUS ENROTH HANSSON

KTH Computer Science
and Communication

Bachelor of Science Thesis
Stockholm, Sweden 2012

Steganography

The art of hiding a message in plain sight

A M E L I A A N D E R S S O N
a n d L I N U S E N R O T H H A N S S O N

DD143X, Bachelor's Thesis in Computer Science (15 ECTS credits)
Degree Progr. in Computer Science and Engineering 300 credits
Royal Institute of Technology year 2012
Supervisor at CSC was Johan Boye
Examiner was Mårten Björkman

URL: www.csc.kth.se/utbildning/kandidatexjobb/datateknik/2012/andersson_amelia_OCH_enroth_hansson_linus_K12004.pdf

Kungliga tekniska högskolan
Skolan för datavetenskap och kommunikation

KTH CSC
100 44 Stockholm

URL: www.kth.se/csc

Abstract

Information is sent back and forth between millions of people every hour, but you never know who might be listening. Mail-clients use encoding to protect the privacy of their users, but is it enough to ensure complete secrecy? We can never be completely sure that the information we store, send or receive hasn't been intercepted or altered.

This report will begin with a background of secret correspondences throughout history, it will then focus on the methods and implementations available today. Among the methods we find Context-Free Mimicry, the steganographic approach used to construct the implementation described in the latter part of this report. A presentation of the user experiment conducted to evaluate this implementation is also found in this report.

Sammanfattning

Mängder av information skickas fram och tillbaka mellan människor varje timma, men man vet aldrig vem som lyssnar. Mejlklienter använder sig av kryptering för att skydda sina användares privatliv, men är det tillräckligt för att garantera ogenomtränglig säkerhet? Vi kan aldrig vara riktigt säkra på att informationen vi lagrar, skickar eller tar emot inte har blivit tillgängliga för utomstående, avlyssnade eller ändrade.

Denna rapport börjar med att ge en bakgrund kring olika historiska metoder för hemliga korrespondenser, för att sedan gå vidare till metoder och implementationer som är tillgängliga idag. Bland dessa går att finna Context-Free Mimicry, den steganografiska metoden som användes till att konstruera den implementation som var del av detta projekt. Implementationen beskrivs i slutet av rapporten, tillsammans med det utvärderande användarexperiment som utfördes.

Statement of collaboration

During the scope of this project the authors of this essay have divided the work between them in a fairly equal way.
A. Andersson was in charge of the construction of the grammar and L. Hansson was in charge of the implementation. When it comes to writing the report, the authors have had responsibility over some chapters and a couple of chapters have been written by both authors. During the whole writing process, proofreading has been done by both the authors as well as external people.

Table of contents

1. Background

1.1 Introduction

The concept of concealing written information isn't, as we shall see shortly, a new one. Some methods that go back to ancient times are still used today. Despite this, the idea is all but outdated. Privacy is a subject that is very much debated in today's society, particularly in association with the technology that has been made available over the past couple of years. The Internet has been shown to be a whole new world in its own right and the boundaries of that world are constantly being redefined and altered, making it hard for users to keep track of what an online service legally can do and what it cannot. If a text or a picture is published on the Internet or stored without reasonable protection it can easily be copied once, twice or even thousands of times within a matter of seconds. Methods of protecting sensitive information is therefor a very current subject and one that should be explored.

In hiding information, there are two approaches one can take. The first, and surely more popular one, is called cryptology. The second approach is called steganography, which is the approach this report will focus on and base an implementation upon. The next part of this chapter will provide an introduction to these two approaches to prepare the reader for the problem definition of the project this report is based on.

1.2 Cryptology

In the early days of written communication, most people were illiterate, making advanced ciphers relatively unnecessary. Listed below are three examples of ciphers using two different techniques to encrypt a message. Two of them use substitution and the last one uses transposition.

Common for all these are that the receiver has to know what keyword or method was used to encrypt the message in order to be able to decrypt it. Also, all ciphertexts are gibberish, if someone else intercepts the message he or she will know that something has been done to the text. If the intention of the interceptor is to steal information he or she would most likely try to crack the code right away.

1.2.1 Caesar cipher

One of the most widely spread encoding techniques is the Caesar cipher, which is a kind of substitution cipher. When encrypting a message each plaintext letter is replaced with another letter a fixed number of steps further down the alphabet. In order to decrypt the message, one does exactly the same thing but in reverse, namely replace the letter with the letter the same amount of steps up the alphabet.[1]

A small example of a message encoded with a Caesar cipher of shift 3:

Plaintext: the quick brown fox jumps over the lazy dog

Ciphertext: WKH TXLFN EURZQ IRA MXPSV RYHU WKH ODCB GRJ

1.2.2 Vigenére cipher

A Vigenére cipher is an encoding method using a series of Caesar ciphers. To encrypt a message you need a matrix which consists of the alphabet written 26 times in different rows where each row is shifted one step from previous row, see image 1. Along with this one must have a keyword which is repeated until it has the same length as the message to encrypt, see the example below. To encrypt a message using the Vigenére cipher one matches each keyword letter / message letter pair with a specific cell in the matrix. The keyword letter specifies what row to look at and the message letter specifies what column to look at. [3]

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Image 1: The Vigenére Matrix

The same example as above but encoded with Vigenére cipher using the keyword “keyword”:

Plaintext: the quick brown fox jumps over the lazy dog

Keyword: key wordk eywor dke yword keyw ord keyw ord

Ciphertext: DLC MIZFU FPKKE IYB FIDSC STAF KKO PYVM URQ

1.2.3 Rail Fence Cipher

The Rail Fence Cipher is a kind of transposition cipher, where the ciphertext is a permutation of the plaintext. In a transposition cipher the positions of the characters (or groups of characters) are shifted according to a regular system, which creates a permutation of the plaintext. [17]

In order to encrypt a message using the Rail Fence Cipher one has to decide how many rows to use. The plaintext is then written downwards as many rows as decided, and when the bottom is reached one simply starts from the top again. Now the rearrangement of the characters are complete and the encoded message is read from left to right as usual.

For instance, the plaintext from above, using three rows, can be encoded like this:

Plaintext: the quick brown fox jumps over the lazy dog

Rail Fence:

t . . q . . c . . r . . n . . x . . m . . o . . r . . e . . z . . o .
. h . . u . . k . . o . . f . . j . . p . . v . . t . . l . . y . . g
. . e . . i . . b . . w . . o . . u . . s . . e . . h . . a . . d . .

Ciphertext: tqcrn xmore zohuk ofjpv tlyge ibwou sehad (ciphertext divided into blocks of five to help avoid errors).

1.3 Steganography

In contrast to cryptography, steganography does not reveal its wish to be kept secret from sender to receiver. While cryptography is solely concerned with concealing the contents of a message, steganography is concerned with concealing the existence of the message altogether.

One of the first known books in the field was Steganographiæ by Johannes Trithemius, written c.1499, but already by the 16th and 17th centuries there had arisen a large literature on steganography. Gaspar Schott explained in his book Schola Steganographica(1665) how to hide messages in music scores where each note could correspond to a letter in the alphabet. This, in theory, could enable two musicians to have a conversation with each other using nothing but their instruments.

Another still widely used and simple method within the field of steganography is the acrostic. An acrostic is a form of writing in which some recurring feature spells out a message. The recurring feature could for example be the first letter of a paragraph or the first syllable of the page. The writers Lewis Carroll and Edgar Allen Poe both wrote poems that were acrostics.[5]

Elizabeth it is in vain you say
"**L**ove not" — thou sayest it in so sweet a way:
In vain those words from thee or L. E. L.
Zantippe's talents had enforced so well:
Ah! if that language from thy heart arise,
Breathe it less gently forth — and veil thine eyes.
Endymion, recollect, when Luna tried
To cure his love — was cured of all beside —
His folly — pride — and passion — for he died.

- *An Acrostic* by Edgar Allan Poe [11]

In the example above, the first letter in each row constitutes the word ELIZABETH.

An example of more modern steganographic methods is audio steganography. When using steganography in audio files one can alter the binary sequence of the audio file to conceal the message one wishes to send. Another method used today is to conceal messages in digital images by altering the least significant bits in all color components.[16]

1.4 Problem definition

The aim of this project is to complete a study of different methods of hiding secret text in a cover text. The project will be divided into two parts. The first part includes an introduction to the different approaches to lexical steganography. Here the methods are described, compared to each other and discussed. Some current implementations to lexical steganography will also be introduced.

The second part of the project will be based on an attempted implementation by the authors. Our goal will be to create a cover text that does not raise suspicion if intercepted by a human. A user test will be conducted to evaluate the implementation, in different contexts and with different inputs. The user test will be presented and discussed in this report along with a description of the implementation.

2 Linguistic steganography

2.1 Methods

Acrostics, explained in section 1.3, is one example of hiding information in text. But, due to its repeatability, it is just as easily detected and cracked as it is created. Using the text's *language* – i.e., its choice of words, its sentence structure or its meaning as a tool for hiding text is far more challenging in both directions. This method of encrypting text is called *linguistic steganography.* One can approach this method in three ways, each one explained below.

2.1.1 Lexical steganography

The lexical approach to a steganographic implementation focuses on avoiding suspicion on the interpretation of each word in a text. This approach exploits synonymy. Words in natural language are linked together by the lexical relation of synonymy. The word *nice* can for example be replaced with *fine, great, decent* or *wonderful*, without significantly changing the sentence; I live in a *nice* house.

In lexical steganography a set of words that can be used interchangeably is called a synonymy set. These sets can be encoded in different ways. One way of encoding a synonymy set is of course to translate the order of the word into a bit string. The first word *nice* represents the string '01'. The second word *fine* represents the string '10' and so on. Using this code one can transmit a secret message through altering just a few words in a text. The substitution of words should be evenly distributed throughout the text to lower the risk of suspicion.[4]

The main problem with working with synonyms is of course multiple meanings of words. For example, *too* in the phrase *I'm happy for you too* does not have the same meaning as in *I'm too short to ride*. In the first phrase, *too* could be a synonym for *also* whereas replacing *too* with *also* in the second phrase would not work.

Another problem is the writer's personal choice of words and style of the text, which could vary greatly. If the steganogram is a formal letter, for example, the sudden appearance of a familiar or slang word in the text would be considered suspicious.

2.1.2 Context-free mimicry

Context-free mimicry, on the other hand, wants to ensure that the interpretation of a *set* of words does not raise suspicion, in the aspects of grammar and sentence structure. This translates to an essentially syntactic approach. To get the syntax of a sentence one can turn to a Context Free Grammar (CFG) as a model. CFG is a formal

grammar in which there are *nonterminal* symbols and *terminal symbols.* The nonterminal symbols represent one of the terminal symbols. This relationship can be illustrated as follows:

V → w

where V is a nonterminal symbol and w is either a set of terminal symbols, nonterminal symbols, or empty. [6] To illustrate CFG further in the field of language we have the following simple example:

S → A	N	V
A → a	the	
N → baby	cat	dog
V → is sleeping	is eating	is dying

As we can see, the string S is made up of three sets of words in this order: an article, a noun and an action. Here the A denotes the article, N denotes the noun and V denotes the verb. From this we can get several different combinations of sentences, e.g. a cat is eating, the baby is dying.

One way to exploit this is to let every word the sender chooses for each of the steps represent a part of a bitstring. In the above example *a*, *baby* and *is sleeping* would be represented by zeros and the full sentence would be represented by the bit string '*000*'. If a sender and receiver decide on a model of sentence structure(s) all they need to decode the text is the grammar and the text. [4]

2.1.3 Ontological

While context-free mimicry has no regard whatsoever for semantics, the ontological approach is all about retaining the *meaning* of the text throughout the process, or rather making sure each set of words are composed in such a way that their interpretation does not raise suspicion. Instead of replacing single words this technique replaces phrases with equivalent meaning. One can consider the following sentences to have equivalent meanings:

- Isak is listening to music.
- The music is being listened to by Isak
- Isak lyssnar på musik

As in the result of the mimicry example, the above sentences can be broken down into different classes - there is an actor, an action and a noun. But in constructing phrases with equal meanings one has to ask the following questions:

- What is the grammatical mood of the set of words? Are they an expression of fact, desire, command, etc.?
- What constraints do we have in the aspect of transitivity? What relationship do we have between the words? Is Isak just listening or is he listening to something?
- What mood do we want to use? Active or passive?

Going through these questions we can make different choices of phrase structure without affecting our intended meaning. The phrase *Isak is listening to music* can very well through the use of a passive mood instead of an active mood be changed into *The music is being listened to by Isak,* or, regarding the transitivity, be broken down into two separate phrases: Isak is listening. The music is being listened to. [4]

2.2 Current implementations

There exist some implementations today working with each of these three approaches. There is the *Tyrannosaurus Lex* implementation, which substitutes synonyms in a similar way to the example given in section 2.1.1.

There are also implementations of mimicry systems, the most notable of these created by Peter Wayner who has published a book on the subject. A sample of these implementations can be found on the book's website [7]. The sample uses a grammar mimicking spam.

2.2.1 Tyrannosaurus Lex (Winstein)

The Tyrannosaurus Lex system is an open-source synonym substitution system designed by Keith Winstein. It relies on a synonym lexicon offering interchangeable sets of words. When encoding, a word is looked up in the lexicon and if a set of synonyms exists the set of words are interpreted as mixed radix digits. Consider the following example, taken from [4]:

In this simple example, the lexicon contained the following synonym sets: {bastioned(0), fortified(1)} {furthermore(0), moreover(1)} {elector(0), voter(1)} and {iii(0), three(1)}, and used these to hide the bitstring (“1101”).

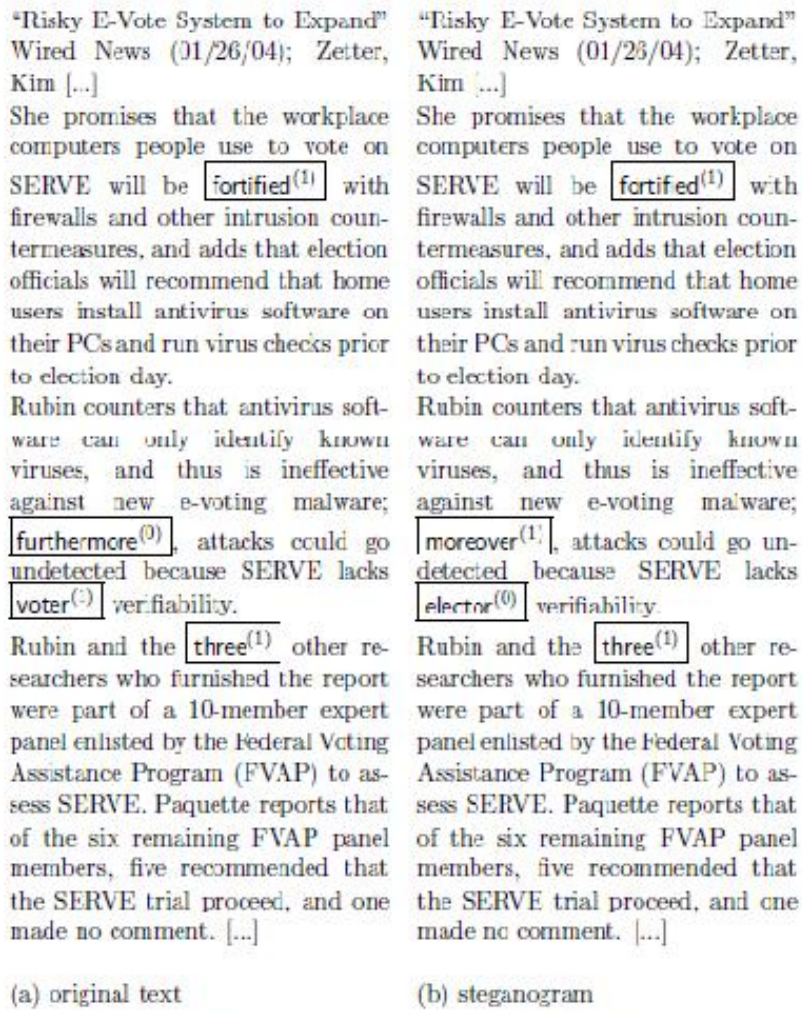

Image 2: To the left we have the original text, to the right we have the covertext hiding the bitstring 1101 using synomynsets

2.2.2 Mimicry (Wayner)

Wayner's system mimics the real structure of natural language. It relies on probabilistic context-free grammar (PCFG), which is just like CFG (explained briefly in section 2.1.2), only the structure of the sentence is based on probabilities. Let us take a look at a more complex example than the one given in section 2.1.2, taken from [4]. Consider the following production grammar:

S -> AB (0.25) 00
S -> AC (0.25) 01
S -> DC (0.25) 10
S -> DB (0.25) 11

A -> Good Golly, (0.25) 00
A -> Whoa, (0.25) 01
A -> Wow, (0.25) 10
A -> Zounds, (0.25) 11

B -> loving E (0.5) 0
B -> a winter's night E (0.25) 10
B -> friendship E (0.125) 110
B -> snuggling E (0.125) 111

C -> panthers F (0.25) 00
C -> pterodactyls F (0.25) 01
C -> Gila monsters F (0.25) 10
C -> serpents F (0.25) 11

D -> Hmmm, (0.5) 0
D -> Well, (0.25) 10
D -> I'm not sure about that, but(0.25) 11

E -> is better than no hair at all.(0.25) 00
E -> is a word for kittens (0.25) 01
E -> is better than pickles for lunch.(0.25) 10
E -> shouldn't be overestimated.(0.25) 11

F -> shouldn't be left unattended with kittens (.25) 10
F -> aren't such bad pets in the scheme of things (.5) 0
F -> are the meanest part of an end. (.25) 11

To encode the bitstring 11101011 using this grammar do the following:

1. Consider the four choices of production, the first two bits of the bitstring, "11", lead us directly to the fourth choice S -> DB.
2. We now have "101011" left with the instructions to expand D and B, starting with D. The bits "10" leads us to D's second choice - ["Well,"].
3. The bits left to encode are "1011", and the phrase we have so far is "Well, " with the instruction to expand B. The "10" leads us to add ["a winter's night" + E] to our phrase.
4. The phrase we now have is "Well, a winter's night" with the bits "11" left to encode and an instruction to expand E. The bits lead us to E's fourth choice ["shouldn't be overestimated."] which also terminates the sentence.

This results in the phrase "Well, a winter's night shouldn't be overestimated.", which could, in an irrelevant context, be considered ridiculous by humans but a computer would find it hard to see anything suspicious about it. To fool humans one would need very general, yet varied sentence structures.

2.2.3 NICETEXT (Chapman)

Chapman's system NICETEXT is similar to Mimicry since it uses a purely symbolic model of linguistic similarity. NICETEXT is also similar to Tyrannosaurus Lex since it relies on interchangeability sets and that it does not rely on a given innocuous text. One difference between Tyrannosaurus Lex and NICETEXT is that the interchangeability sets have to have a cardinality which is a power of two in NICETEXT.

NICETEXT combines these two techniques and creates, supposedly, innocuous text using a style template originating from either a grammar or a sample text. Using the grammar

S --> NP VP
NP -> Det N
NP -> N
VP -> V NP

where N is a list of nouns, V is a list of verbs and Det is a list of determiners, one can, for instance, derive the sentence-model

Det N V Det N

as in "The cat chases the mouse".

2.3 Discussion of methods

Considering our ambition to create an implementation with one of the three approaches mentioned, we needed firstly to determine which of the three would actually be feasible to implement well within the time given to complete this project. The ontological approach seemed to us too complex to implement with a satisfactory degree of functionality. One would have to make sure the implementation preserves the meaning of each and every phrase when encoding, meaning one would have to construct a grammar utilizing both synonyms and grammatical moods. Relying on only synonyms, i.e. taking the lexical approach, would be difficult enough. For the covertext to be convincing one would have to have a large text to use for sparse encoding. At the same time one would have to make sure all synonyms in each set would work in the contexts they would be encoded into.

When constructing a smaller implementation, as we were obliged to in this case, context-free mimicry is by far the easiest option. With a relatively small grammar, one can create acceptable covertexts originating from plaintexts of different lengths. One advantage with this method is that it is not dependent on an already existing text, as in the lexical approach. The grammar gives all text necessary to create the covertext.

3. Implementation

Although a decent implementation of the mimicry approach to linguistic steganography is known to be effective in fooling computers, we wanted to create a cover text that would be able to get past the scrutiny of the human eye as well.

As mentioned in the background of this report, different forms of poems have been used throughout modern history to send secret messages. This makes sense as poetry is a platform in which one can work and bend the rules of grammar and common sense through the excuse of artistic freedom. We decided to recycle this idea for our implementation and constructed the program which we in this report will henceforth refer to as MimicVerse.

MimicVerse was constructed with the help of the Swedish poem “Åsen” [8] written by the Nobel prize winner Erik Axel Karlfeldt. “Åsen” fits our idea of an implementation well as almost each and every line could be separated into two separated phrases that, at least in rhythm, were remarkably close to being interchangeable with the others. From the poem, a grammar (see Appendix [A]) was constructed that would enable MimicVerse to encode 10 bits (or two characters, see further explanation below) for each line in the steganographic poem. The grammar consists of several hashmaps, one for each list of words, where both the key and value are strings. The key is equivalent to the binary sequence corresponding to the value. When encrypting, MimicVerse divides the plaintext into pairs of characters and translates each pair to one line of covertext.

To encode and decode the entire Swedish alphabet plus space (30 different characters) we needed five bits for each character. 10 bits per line would thus allow two characters to be encoded into every line. In order to convert a plaintext into a binary string, we used a predefined function containing 30 different binary sequences, all of length 5. With 5 binary digits, 32 possible unique combinations can be used. We only needed 30 combinations to cover the Swedish alphabet plus space. Table 1 shows the corresponding binary sequence for each character.

In order for MimicVerse to work, we need an even number of characters when the encoding begins. Since that is not always the case, we used the last two combinations, namely '11110' and '11111', to make sure that we always have an even number of characters in the plaintext. In the case where the plaintext is of uneven length, we append '11111' to the binary string, and where the plaintext is of even length we append '11110 11111' to the binary string. With this addition, we create either a complete row or half a row in the end of the covertext that we have control over. We know how the encoded message will end and we can take that into account when we decrypt the message again. In the case of an even number of characters in the plaintext, we append the line

“min barndomsvän, där skall hon försvinna”

Character	Binary Sequence	Character	Binary Sequence
space	00000	o	01111
a	00001	p	10000
b	00010	q	10001
c	00011	r	10010
d	00100	s	10011
e	00101	t	10100
f	00110	u	10101
g	00111	v	10110
h	01000	w	10111
i	01001	x	11000
j	01010	y	11001
k	01011	z	11010
l	01100	å	11011
m	01101	ä	11100
n	01110	ö	11101

Table 1: All characters with their corresponding binary sequence.

The encoding works as follows:

1. Read a plaintext from a file.
2. Convert the text to a binary sequence using the method above.
3. If the plaintext was of uneven length, append '11111' to the binary sequence. If the plaintext was of even length, append '1111011111' to the binary sequence.
4. Encrypt 10 binary digits, equivalent to two characters, at a time using the grammar from Appendix A.
5. Remove a trailing carriage return and save covertext to file.

The decryption works as follows:

1. Read a covertext from a file.
2. Go through list A from the grammar and check if the covertext starts with any of the strings stored there. If not, do the same with list B from the grammar.
3. When a match is found, remove that string from the covertext, add the corresponding key to the binary sequence and continue to either list C/D or list E/F depending on which list the match is found in.
4. Go through the list I and find the matching line ending, remove it from covertext and add the corresponding key to the binary sequence.
5. If the covertext is not empty, go back to step 2.
6. Convert the binary sequence to characters using the method described above.

For example the Swedish word KÖTTBULLAR would result in 5 lines of steganographic verse plus the last line that we append where each pair of letters in KÖTTBULLAR would result in a line.

4. User experiment

The experiment consisted of encrypting a message between 20-120 characters and letting a user try to guess which of four texts presented to him or her was the covertext to that message. The three texts used as comparisons were either original Swedish poems or poems written in another languages translated into Swedish using Google Translate. See Appendix C for a complete list of all poems used in the experiment.

The users were chosen at random, and were of different age and background, though all fluent in Swedish. We only permitted one test per user, as our grammar is very specific and would most likely have been recognized if a user had seen it before.

The hypothesis was that our implementation would create good poems for inputs of length 30 or less and acceptable poems for input of lengths between 31 and 64. With input of greater length the belief was that the implementation would create a poem with too much repetitiveness to be accepted by the users.

4.1 Method

Firstly, an input string of restricted length was acquired from the user and encoded using the MimicVerse. The resulting text was printed and shown to the user along with three other texts, which were chosen alternatively with the following two models in mind:

> **Model 1:** All poems were original and at least one of the poems were translated using Google Translate. The translation were made either to Swedish from another language or from Swedish to English and then back to Swedish.
>
> **Model 2:** All three poems were unmodified Swedish poems.

All poems used as comparisons to the MimicVerse texts were not modified in any way apart from elimination of all blank rows so as to make them compact and more easily comparable to the MimicVerse. The poems were translated using Google Translate and taken as they were without modification (except the removal of blank rows) before being presented to the user. They were, however, chosen with care to give our implementation a fair chance. Rhyming Swedish poems, for instance, were avoided as they would most likely prove too big a challenge. We made sure that the poems were translated properly, without containing too many obviously untranslated words. The poems were also chosen according to length, as we wanted to test the implementation with different input lengths we wanted to have a variety of lengths among the comparison poems to choose from.
The users were instructed to locate the cover text and if they did not chose MimicVerse's text on their first attempt, to continue with the other poems until they

found the right one. By doing this we wanted to see if there were any difference between the two models regarding the amount of guesses the users needed before finding the cover text. Our hypothesis was that the users would be less easily fooled when comparing the cover text to original Swedish poems.

4.2 Result

The results of the user experiment are presented in the tables below. The tables depict how many of the users chose the MimicVerse-poem (the "correct answer") as their first choice and how many chose one of the other poems for different lengths of the input.

Result from Model 1, comparing the cover text with translated poems.

Number of characters	Chose MimicVerse	Chose other	Total
20 – 30	0	3	3
31 – 64	1	1	2
> 64	1	1	2

Table 2: The results using Model 1

Result from Model 2, comparing the cover text with unmodified Swedish poems.

Number of characters	Chose MimicVerse	Chose other	Total
20 – 30	1	3	4
31 – 64	1	1	2
> 64	0	2	2

Table 3: The results using Model 2

Total result.

Number of characters	Chose MimicVerse	Chose other	Total
20 – 30	1	6	7
31 – 64	2	2	4
> 64	1	3	4

Table 4: The total result from the user study.

As we can see, the MimicVerse implementation did very well for small inputs. The short inputs gave a good result, only 1 out of 7 users found our cover text on their first attempt. The result from Model 2 with input greater than 64 characters, does not comply with our hypothesis since neither of the two users found the cover text. The users did, in fact, find the cover text on their third and fourth try respectively.

5. Discussion

5.1 Remarks on Implementation

The aim of MimicVerse implementation and the user experiment was to investigate if context-free mimicry is an effective method for hiding information from humans. The implementation is by no means optimized to its full potential. The grammar was small and its size increased the risk of repetitiveness for large inputs during testing. Presenting the covertext in the form of a poem, however, made the repetitiveness more acceptable. During our research we did not come across any modern method hiding information in mock poetry, and we felt it was an ideal way to try to adapt a method so commonly used in the past to the modern age.

The grammar was, as mentioned, taken from an already existing poem that suited our idea of an implementation. One could just as well, and perhaps with better end results, create a grammar from scratch. Using an existing poem to form a context-free grammar was easy and convenient but is by no means encouraged for future implementations as it poses a high safety risk. The users could easily have found the original poem through an ordinary search engine and jumped to the conclusion that what they were reading was a covertext. The creation of MimicVerse and the user experiment was only meant to explore the possibility of constructing a covertext looking like a poem, it was not meant to be bullet-proof.

5.2 Remarks on User Experiment Method

We instructed all users to locate the covertext and if failing to chose our text on their first attempt, to continue with the other poems until they found the right one. By doing this we sought to reach a conclusion regarding a difference between presenting the covertext among authentic Swedish texts versus translated texts. From the results we could see no such difference. Our hypothesis was that the covertext would be more easily identified among native Swedish texts. Perhaps the results would have been different had we had a greater library of poems to present to our users as comparison-texts. For our study we used nine Swedish poems and nine translated poems.

One comment we received in more than half of the user tests was that the user had only guessed which of the texts presented could be the covertext, as he or she didn't know what to look for. Many users thought all poems were coherent, the rows seemed to fit together. But when the user studied all poems more carefully they could, sometimes, see a more regular pattern in our text compared to the others. From this we conclude that the covertext in most cases blended in nicely with the comparison-texts at first glance.

As a whole, we felt we constructed an acceptable method for our user experiment. Although all the texts presented to the user in the experiment except the covertexts

were consciously chosen by us to make it hard for the user, they were all genuine and can easily be found on the Internet. The translated poems were relevant to the study as web-translators are so frequently used in today's society. If one wanted to know the meaning of a poem written in a language one didn't understand one would use an online translator.

5.3 Remarks on Result

The results of the user experiment should be viewed as small indicators of MimicVerse's suspicion-rate depending on the size of the input. The study was small and is definitely not a reliable source for retrieving good statistics. A larger study, involving a significantly larger library of poems and number of users, is needed for this.

6. References

[1] S. Singh. The Black Chamber. http://www.simonsingh.net/The_Black_Chamber/caesar.html. Viewed 2012-03-07

[2] http://en.wikipedia.org/wiki/Atbash. Viewed 2012-03-07

[3] R. Morelli. http://www.cs.trincoll.edu/~crypto/ historical/vigenere.html, 2010-04-26. Viewed 2012-03-03-07

[4] R. Bergmair. *Towards Linguistic Steganography: A Systematic Investigation of Approaches, Systems and Issues*. http://richard.bergmair.eu/pub/towlingsteg-rep-inoff-b5.pdf. Downloaded 2012-03-07

[5] F. Petitcolas, R. Anderson & M. Kuhn. *Information Hiding – A Survey*. http://gray-world.net/it/papers/petitcolas99information.pdf. Downloaded 2012-03-07

[6] http://en.wikipedia.org/wiki/Context-free_grammar, 2012-02-27. Viewed 2012-03-07

[7] P. Wayner. *Mimicry Applet*. http://wayner.org/texts/mimic/, 1997-08-20. Viewed 2012-03-07

[8] http://www.diktkonst.se. Viewed 2012-04-02

[9] http://www.sv.dikt.org. Viewed 2012-04-02

[10] http://reumatiker.se/tidningar/nr603/030630.pdf. Viewed 2012-04-02

[11] http://www.poemhunter.com. Viewed 2012-04-03

[12] http://www.metrolyrics.com/el-dia-feliz-que-esta-llegando-lyrics-silvio-rodriguez.html. Viewed 2012-04-03

[13] http://www.short-love-poems.net/spanish-love-poems.html. Viewed 2012-04-03

[14] http://www.frmusique.ru/texts/g/gainsbourg_serge/gainsbourg.htm. Viewed 2012-04-03

[15] http://svea.elte.hu/skandin/digilib/froding_ett.pdf. Downloaded 2012-04-03

[16] http://en.wikipedia.org/wiki/Steganography. Viewed 2012-04-09

[17] S. Singh. *The Black Chamber.* http://www.simonsingh.net/The_Black_Chamber/railfencecipher.html. Viewed 2012-04-04

Appendix A, Grammar

S → ACI 00
S → ADI 01
S → BEI 10
S → BFI 11

A → ”knappt spörjs här liv, ” 000
A → ”det är maj, ” 001
A → ”jag hastar ej, ” 010
A → ”den flacka hed, ” 011
A → ”jag vandrar sjungande, ” 100
A → ”för alla fångna känslor ” 101
A → ”ödemark, ” 110
A → ”jag älskar stigarna, ” 111

B → ”den nakna, ” 000
B → ”rör sig en kvinnlig skepnad, ” 001
B → ”där nejden syns, ” 010
B → ”där ljung och timjan blomma, ” 011
B → ”en skara får gnager, ” 100
B → ”där stigen stupar, ” 101
B → ”min barndomsvän, ” 110
B → ”en ensam fur tronar, ” 111

C → ”i talln en kråka gungar” 00
C → ”där luft och dager flöda” 01
C → ”jag vandrar över åsen” 10
C → ”mitt hjärta väntar” 11

D → ”drömsignal i biåsen” 00
D → ”genom dalens vallmor” 01
D → ”mäktigt jubel mig betager” 10
D → ”en ljusröd sol sig tecknar” 11

E → ”du har min kärlek vunnit” 00
E → ”vän att skåda” 01
E → ”jag vill ej henne hinna” 10
E → ”den starke sångarn” 11

F → ”som en hatt med breda skyggen” 00
F → ”i mager mylla” 01
F → ”här bland dessa höjder” 10
F → ”där skall hon försvinna” 11

I → ”.\nJag ser, ” 000
I → ”.\n” 001
I → ”!\n” 010
I → ”;\n” 011
I → ”\n” 100
I → ”,\n” 101
I → ”\nSe! ” 110
I → ”\nO, ” 111

Appendix B, Example from our implementation

Plaintext: The quick brown fox jumps over the lazy dog

Covertext:

En skara får gnager, vän att skåda.
Jag ser, för alla fångna känslor i talln en kråka gungar.
Jag ser, rör sig en kvinnlig skepnad, jag vill ej henne hinna,
det är maj, drömsignal i biåsen;
den flacka hed, drömsignal i biåsen.
Jag ser, jag hastar ej, jag vandrar över åsen!
Jag älskar stigarna, mäktigt jubel mig betager
O, ödemark, drömsignal i biåsen.
Jag ser, ödemark, där luft och dager flöda
O, den nakna, som en hatt med breda skyggen.
Jag ser, jag hastar ej, mäktigt jubel mig betager,
för alla fångna känslor mäktigt jubel mig betager.
Jag ser, där ljung och timjan blomma, du har min kärlek vunnit.
Jag ser, jag älskar stigarna, mäktigt jubel mig betager
Se! för alla fångna känslor jag vandrar över åsen!
Knappt spörjs här liv, jag vandrar över åsen
knappt spörjs här liv, drömsignal i biåsen,
knappt spörjs här liv, där luft och dager flöda
det är maj, mitt hjärta väntar!
Rör sig en kvinnlig skepnad, som en hatt med breda skyggen.
Jag ser, jag vandrar sjungande, där luft och dager flöda
O, jag älskar stigarna, där luft och dager flöda
O, knappt spörjs här liv, mitt hjärta väntar

Appendix C, List of poems

Swedish poems

- Ett Gammalt Bergtroll, Gutav Fröding [15]
- Jagad [8]
- Kanske blir det en dikt ändå, Göran Hansson [9]
- Någonstans, Inga Juhlin [10]
- Olles förbund med makterna, Dan Andersson [8]
- Paris (hon som log och han som dog), Tomas Brink [8]
- Runt September, Göran Hansson [9]
- Vår, Gustav Fröding [8]

Translated poems

- A Valentine, Lewis Carroll [11]
- Comme un boomerang, Serge Gainsbourg [14]
- El día qué está llegando, Silvio Rodriguez [12]
- Je suis venu te dire que je m'en vais, Serge Gainsbourg [14]
- La dulce boca [13]
- LEDA, Rubén Darío [13]
- The sick rose, William blake [11]
- Fit the third (from Hunting of the Snark), Lewis Carroll [11]