Lexical Natural Language Steganography Systems with Human Interaction

Brecht Wyseur, Karel Wouters, Bart Preneel
K.U. Leuven/IBBT – ESAT-SCD/COSIC
Kasteelpark Arenberg 10
3001 Heverlee, Belgium
{brecht.wyseur,karel.wouters,bart.preneel}ATesat.kuleuven.be

Abstract: This paper describes an implementation for linguistic steganography based on word substitution over an IRC channel. A typical problem with linguistic steganography is that it is difficult to pay attention to the semantic cohesion of the result. If automated, sentences can be ripped out of context or appear unnatural. Therefore, we propose a solution that involves human interaction with the steganographic engine. This results in sentences that fit perfectly into the context of the IRC conversation. To provide a continuous flow of parts of the message to be transferred, it is encrypted with a stream cipher while it is embedded. A challenging aspect of this work is the generation of the word substitution table based on a session (stego) key. This is because an acceptable number of synonyms must be available for each word such that the interacting user is able to choose synonyms that produce a credible sentence that fits into the context. As we are dealing with an IRC channel though, spelling errors and abbreviations can be introduced to generate more alternatives. We also indicate how our implementation is related to other steganographic systems, such as automated linguistic steganography systems.

Keywords: Linguistic steganography, IRC, human interaction, information hiding

1. Introduction

Steganography is the ancient art and science of hiding information by embedding messages within other, seemingly innocent-looking messages. Already in the 16th century, a vast volume of literature existed on steganography and methods to encode information. Steganographia (Trithemus, 1606), the work of Johannes Trithemius (1462-1516), is a nice example of the use of steganography: at first sight, it appears to be about black magic and this is why it was listed on the Index Librorum Prohibitorum (”List of Prohibited Books”) of the Catholic Church. Given the decryption key, it shows to be concerned with cryptography and steganography.

A classical example of the use of steganography is illustrated by Simmons (1984) in his “Prisoners Problem”, in which Alice and Bob are locked up in separate cells in jail and wish to communicate about their escape plans, without alarming the warden Wendy (the adversary), who is monitoring their communications. This illustrates that in contrast to cryptography, where the meaning of the message is obscured, steganography tries to hide the mere fact that a certain message is communicated.

We refer to a legitimate object in a communication as a cover object, while a message with embedded information is called a stego object. The stegosystem creates stego objects with embedded messages, using a stego key, such that the operation cannot be inverted trivially. In a secure stegosystem, stego objects are indistinguishable from cover objects. We can distinguish three dimensions in a stegosystem:
Payload Capacity: the ratio of hidden information to cover information
Robustness: the ability of the system to resist against changes in the cover object
Imperceptibility: the potential of the generated stego objects to remain indistinguishable from other objects in the same category

Well-known contemporary examples of cover objects are audio, images and video. These are available in large quantities on the Internet, which makes it harder for an adversary to detect hidden communications. The first stegosystems in this area were often based on LSB (Least Significant Bit) substitution, which transforms a cover object into a perceptually similar stego object.

The problem with this approach is that statistical variances in such changes can be computed automatically and this makes it hard to design a good steganographic algorithm (Chandramouli and Memon, 2001).

Another type of cover object has received less attention in the digital world: the natural language. Contrary to multimedia cover objects, cover texts can be changed heavily by word substitution, without changing the meaning and thus preserving the innocence of the cover text. Moreover, automated analysis of natural languages remains a very hard problem in computer science. This makes automated attacks more difficult.

The remainder of this paper is organized as follows: In the next section, we describe linguistic steganography in more detail. In section 3, we present our approach of using linguistic steganography over an IRC channel. We conclude by indicating how our idea can be elaborated and which improvements could be made.

2. Linguistic steganography

Linguistic steganography is situated within the area of text steganography, which refers to hiding information by using text as the cover object. We can identify three different approaches for using cover text: format-based, random and statistical generation, and linguistic steganography. The security of stegosystems based on these techniques is evaluated to the imperceptibility of the stego text. This relies on the abilities of Wendy, the adversary, to analyse the syntax and semantics of text.

Format based steganography uses modifications to the physical formatting of text, e.g., insertion of spaces or non-displayed characters, font resizing, deliberate misspelling, etc. The resulting stego text can easily be distinguished from innocuous text using syntax checking software or by verification by a human.
Random and statistical generation techniques, using for example grammars that produce texts in a certain natural language, increases the requirements for a machine if it is to be used as an adversary, but with the ever rising computational power of adversaries and the availability of advanced grammar and spell checking software, these constructions also tend to fail. These systems are not secure against human adversaries.

This stimulated further research in techniques that go beyond syntax modification. A type of cover objects that recently gained attention is the set of natural languages, including the context in which sentences appear. The use of such cover objects result in a new type of steganography, referred to as linguistic steganography. These stegosystems are aimed at embedding secrets, using the semantic interpretation which a user assigns to text. To protect against human adversaries and machines powerful enough to check complex linguistic structures, e.g. advanced grammar and spell checking functionality, we want the stego text to be indistinguishable from the cover text. In linguistic steganographic systems, the original cover text is not given which implies that identifying stego text within a set of cover objects boils down to semantic verification of the stego text, which is a hard problem for machines.
In his thesis, Bergmair (2004) presented a construction for linguistic steganography. The encoding of data into text is done by substituting words by one of their synonyms. This can be detected only by an human who is able to identify substituted words within the context of the text. This ability is referred to as word-sense disambiguation. For a machine, word-sense disambiguation is a computationally hard problem, that is studied extensively for decades. Bergmair's original idea is further extended by Bergmair and Katzenbeisser (2004), indicating the limits of a machine compared to a human, using Human Interactive Proofs (HIPs). Furthermore, Bergmair (2004) elaborated his idea further in another direction by using Huffman codes to protect against statistical analysis of the stego text. The main issue is that a bitstring to be embedded into stego text is in general uniformly distributed. Since the usage of words in a natural language is not uniformly distributed, the mapping bitstrings to words should be balanced to avoid simple statistical attacks.
Basing the assignment of bitstrings to synonyms on Huffman codes, tends to move the probabilistic use of words in stego text towards the usage frequency of words in the natural language of the cover text.

This paper is inspired on the ideas presented by Bergmair and Katzenbeisser, but with a different approach. First of all, we do not consider the adversary Wendy to be a machine, but leave the option of Wendy being an intelligent user open. Therefore, synonyms cannot be used in the same way as in the Bergmair system, as they can be detected by Wendy as well. Instead, we use synonyms from a public thesaurus that fit into the context of the cover text. This is done by prompting the user with a choice of synonyms, that can replace selected words. The user should select a synonym that fits into the context of the conversation, such that the conversation keeps its innocence. This is a task that is very difficult for a machine. Each word in a certain subset of the thesaurus represents information and the difference between Bob (intended receiver) and Wendy, is that Bob is able make the link between synonyms and information, while Wendy is not (she is even not able to detect whether or not there is information). The main idea of this construction and how to assign code to words is described in the next section.

3. Linguistic steganography over an IRC channel

3.1 IRC

For our specific implementation of linguistic steganography, we decided to use IRC as a communications channel. The advantage of such a channel is that two parties, Alice and Bob do not need to communicate directly with each other in the public part of the channel. Alice can communicate with other people in public, while Bob is monitoring the conversation. Other advantages of IRC channels are that they are numerous, making the set of cover texts to be investigated enormous, and that IRC can happen anonymously, e.g., over the Tor network (Tor: anonymity online, 2007).

3.2 Setting

If Alice wants to send a message to Bob using our system, we assume the following:
Alice and Bob know in advance which channel to join and which nickname they will choose
They know each other's public key

A typical use of the system is illustrated in Figure 1 and would go as follows: Alice and Bob log into the same chat channel. Alice enters the secret message M into a local queue, and indicates that she wants it to be sent to Bob. Bob indicates that he wants to receive a message from Alice. In the background, a secret key K is agreed upon, using the Diffie-Hellman key agreement protocol. This secret key K is used to generate a word substitution table, based on a thesaurus of which Alice and Bob share a copy. The secret key K is also used to derive a session key Sk, used to encrypt the message M. This encryption is done using a stream cipher (RC4) such that Bob can start decrypting immediately, upon receipt of the very first byte. Then, Alice starts chatting on the channel. Each time she sends a message (cover text), a pop-up suggests some word replacements, based upon the word substitution table, such that part of the message M is encoded and broadcasted as stego text in the IRC channel. This is where human interaction is necessary: automating this would require solving the problem of word-sense disambiguation. Bob intercepts the stego text, decodes the words with the substitution table and then decrypts the transmitted message using RC4 with the session key Sk.

Our approach is related to content-aware steganography, presented by Bergmair and Katzenbeisser (2006). In their approach, information can be hidden from an automated observer (Wendy) because such an observer cannot solve the complex Artificial Intelligence problem of word-sense disambiguation, as mentioned before. In our scheme, the user solves the problem of word-sense disambiguation for the stego system, such that word replacements are done without violating the context in which they occur. This way, the resulting stego text will be indistinguishable from cover texts, even for human observers.

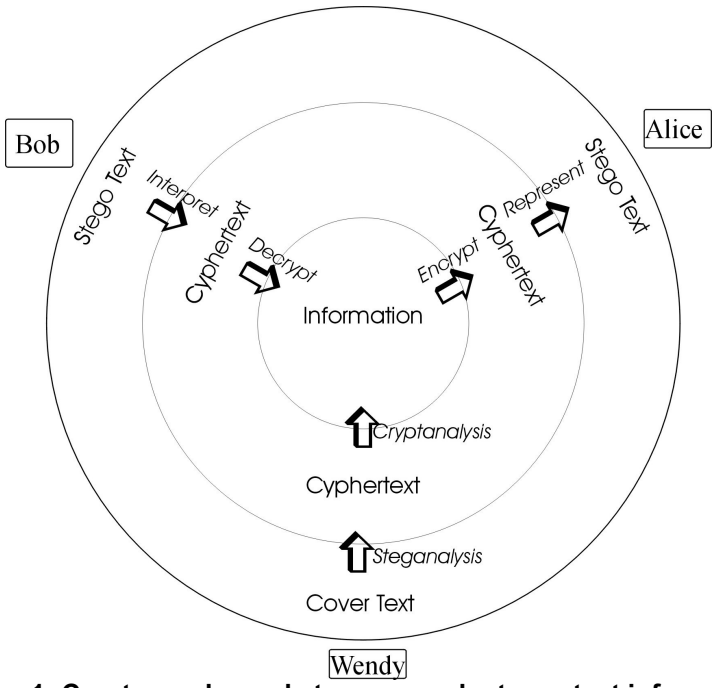

Figure 1: Cryptography and steganography to protect information

3.3 Word substitution table

In our stegosystem, single bits are uniquely assigned to keywords. This means that each time a word which represents '1' is used by Alice in the IRC channel, a '1' will be decoded by Bob. Keywords are selected from a thesaurus. For the purpose of this project, we used the English thesaurus, included in OpenOffice. The selection of keywords and the bits they represent is done as follows:
First, all words with a small number of synonyms are filtered out of the thesaurus, to ensure that enough synonyms can be offered to the user.
From the session key, established through the Diffie-Hellman protocol, a large random key stream S is generated.
Then, S is used to assign bit values to words; a value is represented by two bits in S. Bit values can be '0' (00 in S), '1' (11) or NULL (10 or 01). NULL indicates that this word represents no bit; it is assigned to increase the number of alternatives that can be presented to the user. Note that a word can occur multiple times in the wordlist, in several contexts, if it is very generic. These words are the first ones to be assigned a bit value.
After the assignment of bits to words, a deterministic algorithm is executed to make the distribution of ('0', '1', NULL) better, again to avoid presenting the user with a limited set of synonyms that would change the context of the sentence.

The exact algorithm to determine the word substitution table is still work in progress. Variations can be made, based upon the order of assignment and the deterministic "correction" algorithm. We developed a strategy to test how good a given word substitution table is, based on the number of alternatives of a word that represent a certain bit. In the ('0', '1', NULL)-case, the ideal situation is that each value is represented by a different partition of the synonyms of a given word, following a (25%-25%-50%)-distribution (see Figure 2): 25% of the synonyms represent '0', 25% of the synonyms represent '1' and 50% of the synonyms represent 'NULL'. By trying to converge to that situation, we make sure that the user has enough options to select a synonym that fits into the give context.

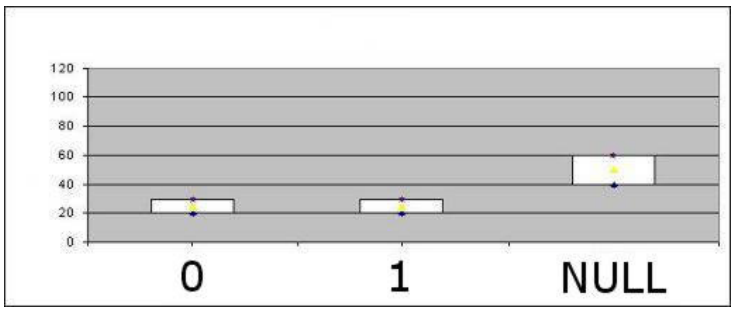

Figure 2: Ideal distribution of representations

We have already developed a prototype algorithm which successfully generates a fairly good substitution table (boxplot of the distribution in Figure 3), but more tests need to be conducted.

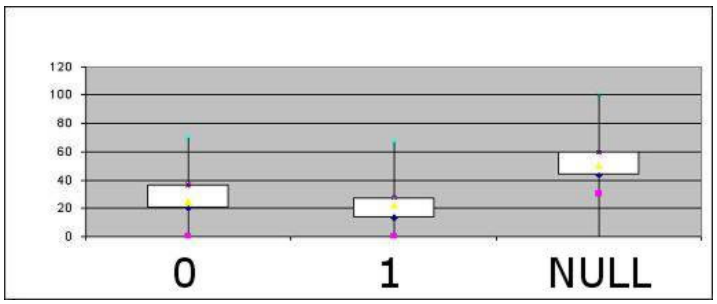

Figure 3: Converging towards the ideal distribution

The distribution shown in Figure 3 is obtained as follows: first, we assign a representation to each word in our list, based on the bitstream S, as discussed above. Then, the correction algorithm goes over the table: this deterministic algorithm makes sure that the percentage of synonyms with a NULL representation is above 30% for any given word. The main problem with this step is that a given word can occur multiple times as a synonym. That means that any change in the representation has forward and backward effects. An example of this can be seen in Figure 4: If the representation of 'big' is changed in the synonym list of 'large', it will have effects on the (earlier) synonym list of 'big' itself, and the (later) synonym list of 'great'.

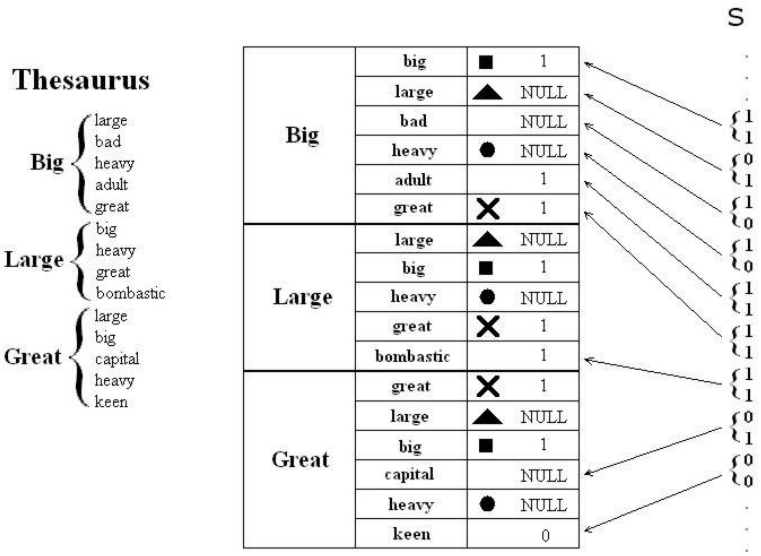

Figure 4: dependencies between synonym sets

The main point in which our approach differs from the one mentioned by Bergmair (2004), is that instead of unbalancing the synonyms list based on Huffman codes, we unbalance the synonyms

list using an association with NULL which Alice should almost always be able to choose. Preliminary results also seem to indicate that the random assignment of bitstrings (in our case '0', '1' and NULL) to *sets* of words instead of single specific words also provides a larger set of possible word substitution tables to be generated. If d is the average number of synonyms for each word and N is the number of words that can be selected for substitution, a first rough estimation of the number of tables is

$$\left(\binom{d}{\alpha} \cdot \left(\left\lfloor \frac{d-\alpha}{2} \right\rfloor \right) \right)^{N} ; \alpha = \left\lfloor \frac{d}{2} \right\rfloor .$$

This number is based on a simplified thesaurus, in which overlapping as illustrated in Figure 4 does not occur. For our tests on the OpenOffice thesaurus, d=7 and N=9732, after infrequent words and words with few synonyms were discarded. This results in 210^N possible tables. If a Huffman-based approach with a depth of 3 bits is applied, then 4 possible sets of code-mappings can be assigned to each synonym set, resulting in 4^N possible word substitution tables. More accurate estimations for the number of possible tables should be done, but preliminary results show that the number of possible tables in our case remains well above the number in the case of Huffman encoding. In both cases, the number of possible tables is a magnitude higher than the number of possible session keys. Therefore, a brute-force attack on the word substitution table is less efficient than a brute-force attack on the session key.
The disadvantage of our approach is the lower payload capacity. Per substituted word, on average 1/2 bits are transferred, while this is 7/4 in the case of a Huffman-based approach of a depth of 3 bits.

3.4 Limitations

There are also some limitations in using IRC with a linguistic substitution stegosystem, in this setting:

- Low payload capacity: as mentioned above, the amount of cover text needed per data bit is rather high, so a lengthy IRC session with a high amount of messages is necessary to communicate a rather small message.
- Statistical analysis may reveal that certain words are not in the default vocabulary, used by a certain person.
- By selecting a synonym, some grammatical errors can be generated. A simple example of this is when "(a) guitar" is replaced by "(a) instrument". Some of these errors can be avoided by technical means or by an extra interaction by the user.

Because of the low payload capacity, the system we propose is only meant to transfer short messages.

4. Implementation

The objective of our project was to build a IRC module for linguistic steganography. We studied several IRC clients, and X-Chat (X-Chat multiplatform chat program, 2007) was chosen, because it is multi-platform, open source, it has a GUI (using GTK+) and has well-documented plugin capabilities.
The steganographic core was written in C and is loaded using a plugin for X-Chat. The Diffie-Hellman key agreement, hash functions for the key expansion of K into S, and the RC4 stream cipher are provided by the OpenSSL cryptographic library (The OpenSSL project, 2007). Figure 5 and 6 give an impression of how the plugin interacts with the user when sending and receiving hidden information.

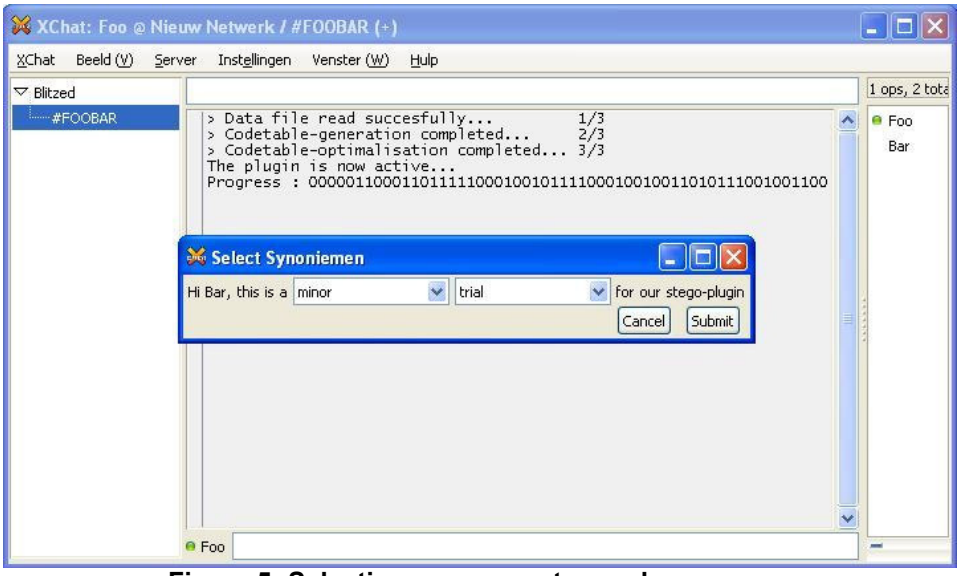

Figure 5: Selecting synonyms to send a message

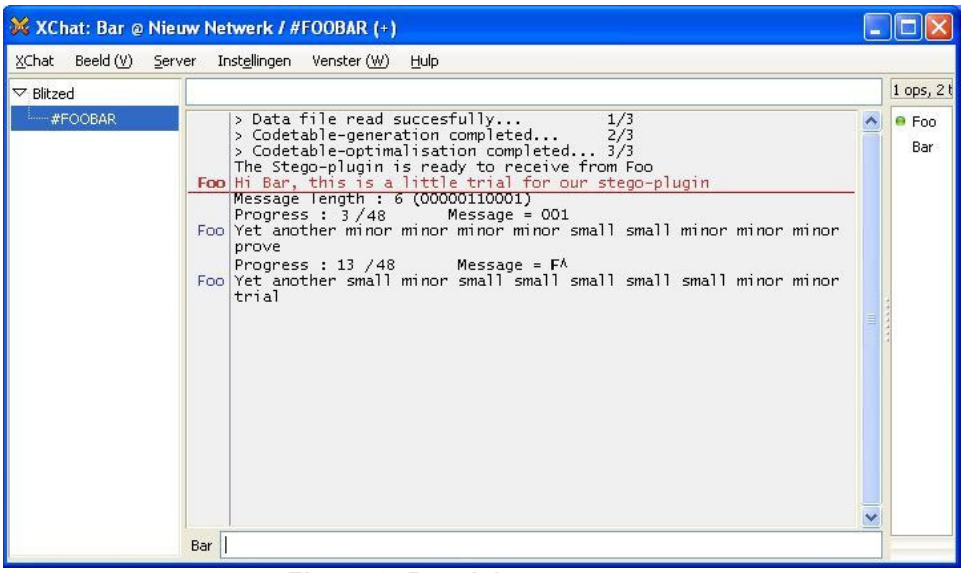

Figure 6: Receiving a message

5. Conclusion and future work

Linguistic steganography is one of the more advanced methods to hide information in text. We used a simple substitution system with human interaction to make sure that the context of sentences is not changed significantly. Doing this in an automated fashion would require the automation of word-sense disambiguation, a challenging problem in Artificial Intelligence. If partial solutions for this problem would exist, they could be employed to semi-automate the substitution process.

Other improvements and possible directions for future research include:

A rigorous statistical analysis should be done on some test subjects, to see if their word-choice differs significantly from the test subjects, chatting without the plugin. Given that two subjects could chat regularly without using the plugin, a word frequency list could be tailored specifically to their interaction, and taken into account when generating the word substitution table. Furthermore, it should be investigated if the addition of IRC abbreviations and spelling mistakes weakens the system. The security of this system also depends on the structure of thesauri. This should be

investigated, to allow for a better estimation of the number of tables that can be generated from them.
To protect against an active attacker, the stegosystem could be made more robust by adding an error-correcting code after encryption.

6. Acknowledgements

This work was partially financed by a Ph.D. grant of the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen). It was also co-funded by the IBBT (Interdisciplinary Institute for BroadBand Technology), a research institute founded by the Flemish Government in 2004. The authors would also like to thank Alexander Alderweireldt and Tim Thaens for the implementation of the presented stego system as an X-chat plugin.

References

Bergmair, R. “Towards linguistic steganography: A systematic investigation of approaches, systems, and issues”. final year thesis, April 2004. handed in in partial fulfillment of the degree requirements for the degree “B.Sc. (Hons.) in Computer Studies” to the University of Derby.

Bergmair R., Katzenbeisser S. “Towards human interactive proofs in the text-domain”. In Kan Zhang and Yuliang Zheng, editors, Proceedings of the 7th Information Security Conference, volume 3225 of Lecture Notes in Computer Science, pages 257–267. Springer Verlag, September 2004.

Bergmair R., Katzenbeisser S. “Content-aware steganography: About lazy prisoners and narrow-minded wardens”. In Proceedings of the 8th Information Hiding Workshop, Lecture Notes in Computer Science. Springer Verlag, 2006.

Chandramouli, R., Memon, N.D. “Analysis of LSB based image steganography techniques”. In Proceedings of the 2001 International Conference on Image Processing (ICIP 2001), pages 1019-1022.

Simmons, G.J. “The prisonners’ problem and the subliminal channel”. In D. Chaum, editor, Advances in Cryptology - CRYPTO’83, Lecture Notes in Computer Science, pages 51–67. Plenum Press, 1984.

Trithemius, J. Steganographia. Frankfurt, 1606.

The OpenSSL project. http://www.openssl.org, accessed March 13, 2007.

Tor: anonymity online. http://tor.eff.org/, accessed March 13, 2007.

X-Chat multiplatform chat program. http://www.xchat.org/, accessed March 13, 2007.