Exploiting Redundancy in Natural Language to Penetrate Bayesian Spam Filters

Christoph Karlberger, Günther Bayler, Christopher Kruegel, and Engin Kirda
Secure Systems Lab
Technical University Vienna
{christoph,gmb,chris,ek}@seclab.tuwien.ac.at

Abstract

Today's attacks against Bayesian spam filters attempt to keep the content of spam mails visible to humans, but obscured to filters. A common technique is to fool filters by appending additional words to a spam mail. Because these words appear very rarely in spam mails, filters are inclined to classify the mail as legitimate.

The idea we present in this paper leverages the fact that natural language typically contains synonyms. Synonyms are different words that describe similar terms and concepts. Such words often have significantly different spam probabilities. Thus, an attacker might be able to penetrate Bayesian filters by replacing suspicious words by innocuous terms with the same meaning. A precondition for the success of such an attack is that Bayesian spam filters of different users assign similar spam probabilities to similar tokens. We first examine whether this precondition is met; afterwards, we measure the effectivity of an automated substitution attack by creating a test set of spam messages that are tested against SpamAssassin, DSPAM, and Gmail.

1 Introduction

The purpose of a spam filter is to decide whether an incoming message is legitimate (i.e., ham) or unsolicited (i.e., spam). There are many different types of filter systems, including:

Word lists: Simple and complex lists of words that are known to be associated with spam.

Black lists and white lists: These lists contain known IP addresses of spam and non-spam senders.

Message digests: These systems summarize mails into pseudo-unique values. Repeated sightings of the same digest is symptomatic of a spam mail.

Probabilistic systems: Systems such as Bayesian filters are used to learn word frequencies that are associated with both spam and non-spam messages [11].

Since Bayesian filters do not have a fixed set of rules to classify incoming messages, they have to be trained with known spam and ham messages before they are able to classify messages. The training of a Bayesian spam filter occurs in three steps: first, each message is stripped of any transfer encodings. The decoded message is then split into single tokens, which are the words that make up the message. Last, for each token, a record in the token database is updated that maintains two counts: the number of spam messages and the number of ham messages in which that token has been observed so far. Besides that, the token database also keeps track of the total number of spam and ham messages that have been used to train the Bayesian spam filter.

Once a Bayesian spam filter has created a token database, messages can be analyzed. Analogous to the training phase, the message is first decoded and split into single tokens. For each token, a spam probability is calculated based on the number of spam and ham messages that have contained this token as well as the total number of spam and ham messages that have been used to train the Bayesian spam filter. The following formula is frequently used for this calculation:

$$P_{spam}(token) = \frac{\frac{n_{spam}(token)}{n_{spam}}}{\frac{n_{spam}(token)}{n_{spam}} + \frac{n_{ham}(token)}{n_{ham}}} \qquad (1)$$

In this formula, n_{spam} and n_{ham} are the total numbers of spam and ham tokens, whereas $n_{spam}(token)$ and $n_{ham}(token)$ denote how many times a token appeared in a spam or ham mail, respectively. Note that there are alternative ways to calculate this probability; an overview can be found in [22]. Next, Bayes theorem is used to calculate the spam probability of the whole message by combining the spam probabilities of the single tokens. Finally, the message is classified as ham or spam, typically by comparing its combined spam probability to a pre-defined threshold.

2 Existing Attacks

The goal of attacks against Bayesian spam filters is to let spam mails be identified as ham mails. Currently existing attacks aim to achieve this by adding words to the spam mails. The objective is that these additional words are used in the classification of the mail in the same way as the original words, thereby tampering with the classification process and reducing the overall spam probability.

When the additional words are randomly chosen from a larger set of words, for example, a dictionary, this is called random word attack ("word salad"). The objective is that the spam probabilities of the words added to the spam message should compensate for the original tokens' high spam probabilities in the calculation of the whole message's combined spam probability. There is some controversy about the effectiveness of such an attack: Several authors have found random word attacks ineffective against Bayesian spam filters [9, 13, 22], because many random dictionary words are infrequently used in legitimate messages and, therefore, tend to have either neutral or high spam probabilities for Bayesian spam filters. An improvement of the random word attack is to add words to the spam mail that are often used in legitimate messages. This is called common word attack. The idea is that often-used words should have lower spam probabilities than randomly-chosen words for Bayesian filters, thus being better suited for an attack. The number of a additional words that are needed for this attack to work varies between 50 [20] and 1,000 [13]. Finally, Lowd and Meek improved the common word attack by adding words that are common in the language but uncommon in spam [13]. They called their attack frequency ratio attack. Lowd and Meek calculated that about 150 frequency-ratio words suffice to make a spam message unrecognizable.

Another common approach to circumvent Bayesian spam filters is to overlay the text of a message on images that are embedded in HTML. The content of the mail is visible to the user, but it is unrecognizable to text-based filters, which usually ignore images in their analysis of the message [1].

3 Our Approach: Word Substitution

Most attacks described previously have one thing in common: they add words to spam mails. From the spammer's point of view, the disadvantage of adding words to spam messages is that blocks of additional words are indicators of spam, and algorithms that are able to detect these additional words, such as Zdziarski's Bayesian Noise Reduction algorithm [21], foil attacks.

In this paper, we explore an alternative approach: instead of adding known good words to compensate for the bad words in the spam mail, one could exploit redundancies in the language and substitute words with a high spam probability by synonyms with a lower spam probability. The idea of a computer-aided substitution attack was first hinted at by Bowers [2]. Bowers showed that by manually replacing suspicious words, the spam probability of a message can be lowered. However, a completely manual substitution process is clearly impractical for an attacker. In this work, we investigate the feasibility of an automated substitution attack and evaluate its success against three spam filters.

4 Precondition for a Substitution Attack

For a successful substitution attack, it is necessary that Bayesian spam filters at different sites (and for different users) judge words sufficiently similarly. Otherwise, the attacker would not know which words are considered suspicious by the victims' spam filters and, therefore, should be substituted. In addition, it would be unknown which synonyms could be used for the substitution, since it would be equally unknown which words receive a low or neutral spam probability by the victims' Bayesian spam filters.

The spam probability of a word is determined (a) by the number of appearances of this word in spam mails, (b) the number of appearances of this word in ham mails, (c) the total number of spam mails, and (d) the total number of ham mails the Bayesian filter has classified. If the spam mails and the ham mails of users of Bayesian spam filters are sufficiently similar, then it is reasonable to assume that words are classified similarly enough for a substitution attack to work.

Are the mails used for training Bayesian spam filters of different users the same? This is clearly not the case. But many spam filters are set up for more than one user; that means, they use broader samples of spam and ham mail for training. Even more important, however, is that we can assume that many users receive very similar spam mails. After all, the idea of spam is the wide dissemination of particular messages. Thus, it is reasonable to assume that many users receive similar spam messages, and as a result, their filters assign high spam probabilities to the same (or very similar) sets of words.

Another aspect to consider in this regard is how fast messages, in particular spam messages, mutate. When message content changes too quickly, the classification of the words in the messages would change too. As a result, the effectiveness of the attack decreases, because the adversary does not know which words to replace, and which synonyms to choose. In order to determine whether spam messages change slowly enough to allow a substitution attack, we examined three different spam archives. We extracted messages received in the year 2006, divided them by the month they were received, and created lists of the most frequently used tokens for each month. We then measured the overlap of these lists by comparing them. The goal was to determine how many of the 100 most frequently used tokens of one month appear among the 100 most frequently used tokens of another month. The results in Table 1 show that the majority of the 100 most frequently used tokens in one month's spam messages appear among the top 100 tokens of another month's spam messages.

Manual inspection of the most frequently used tokens showed that the lower the rank of a token in this list is, the less is the difference of that token's and the next frequently used token's number of appearances. Some tokens that are at the end of the list of the 100 most frequently used tokens of one month do not appear on another month's top 100 list and, therefore, lower the overlap. However, many of these tokens are not completely missing in the spam corpus of that other month, but are only a little bit too infrequent to appear in the list of the 100 most frequently used tokens. If that border case tokens would count too, the overlap would be higher; to be able to estimate the overlap including the border case tokens, we also measured how many of the 100 most frequently used tokens of a month appear among the 200 most frequently used tokens of another month. The results for this type of comparison is shown in the right half of Table 1.

Our results demonstrate that many of the terms used in spam messages do not change over the course of a year, which is a certain indication that the spam probabilities Bayesian filters assign to these terms do not change too much either. These findings are confirmed by related studies: Sullivan [18] examined "almost 2,500 spam messages sampled from 8 different domains over a period of 2.5 years" and found that spam is relatively time-stable. Pu and Webb [16] studied the evolution of spam by examining "over 1.4 million spam messages that were collected from SpamArchive between January 2003 and January 2006." They focused their study on a trend analysis of spam construction techniques, and found that these changes occur slowly over the course of several months, confirming Sullivan's claim.

5 Substitution Attack

As mentioned previously, the goal of the substitution attack is to reduce the overall spam score of a mail by automatically replacing words with high spam probability by words with low spam probability. This is done in several steps:

1. All words with a very high spam probability are identified.
2. For every such word, a thesaurus is queried to find a set of words with similar meaning, but with a lower spam probability.
3. If a set of suitable synonyms is found, the spam word is replaced with one of the possible candidates.

Identify words with high spam probability. Words that raise the spam probability of a message need to be automatically replaced by words that have a lower spam probability. To this end, the spam probability of each word in a message has to be determined. For this, we query the Bayes token database of a spam filter. More precisely, we trained SpamAssassin with about 18,000 spam mails from Bruce Guenter's Spam Archive [10] and about 12,000 ham mails from the SpamAssassin and Enron ham archives [6] to prepare SpamAssassin's Bayes filter with a large and comprehensive training corpus. Then, for each word of a message, SpamAssassin was consulted to derive the spam probability for this word. We chose the SpamAssassin mail filter [17] for this task because it is widely used and achieves good results in spam detection.

Based on the Bayesian spam probabilities for each word, the decision is made whether this word needs to be replaced. To this end, a *substitution threshold* is defined. If a word with a spam probability higher than that threshold is found, it is replaced with a synonym. In the following Section 6, we show results of experiments using different values for the substitution threshold.

Finding words with similar meaning. If a word with a spam probability above the threshold is found, WordNet [15] is queried for alternatives. WordNet is a lexical database that groups verbs, nouns, adjectives, and adverbs into sets of cognitive synonyms (called *synsets*). That is, each synset represents a concept, and it contains a set of words with a sense that names this concept. For example, the word "car" is classified as noun and is contained in five synsets, where each of these sets represents a different meaning of the word "car." One of these sets is described as "a motor vehicle with four wheels" and contains other words such as "automobile" or "motorcar." Another synset is described as a "cabin for transporting people", containing the word "elevator car." For every synset, WordNet provides links to *hypernym synsets*, which are sets of words whose meaning

Spam Archive	Messages examined	Overlap 100/100(%)			Overlap 100/200(%)		
		min.	max.	avg.	min.	max.	avg.
Bruce Guenter's SPAM Archive [10]	127,120	68	91	81.2	76	100	94.1
SpamArchive.org [3]	334,477	65	95	80.0	78	100	94.6
TLIQuest Spam Archives [19]	52,799	63	93	79.2	80	100	94.2

Table 1: Overlap of most frequently used tokens in three different spam archives for 2006 (see Section 4).

encompasses that of other words. That is, a hypernym is more generic than a given word. For example, "motor vehicle" is a hypernym of "car" [15].

Whenever WordNet is queried for alternatives for a particular word, the tool not only requires the search word itself, but also additional information that describes the role of this word in the sentence (such as whether this word is a noun, a verb, or an adjective). The reason is that WordNet distinguishes between synsets for verbs, nouns, adjectives and adverbs, and it is necessary to specify what kind of synsets the result should contain. For example, if a word that is used in the query is a noun, it would not make sense to look up synsets of verbs. Obviously, many words possess more than a single role. For example, the word "jump" can either be used as verb or as noun. The natural language processing (NLP) tool that we use to perform the recognition of the roles of words in a sentence is contained in the LingPipe NLP package [12]. This tool relies on the context of a word to discover its role in a sentence and assigns a tag to each word that describes its part-of-speech role [4].

Once the role of a word is discovered, WordNet can provide all synsets that contain the search word in its proper role. In addition, WordNet is also queried for all direct hypernym synsets of these sets, because hypernyms can also act as synonyms and, therefore, expand the search space for suitable replacement words. Unfortunately, the role of a word is not sufficient to select the proper synset. The reason is that one must select the synset that contains those words that are semantically closest to the original term. As mentioned above, the noun "car" could be replace by the term "automobile", but also by the word "cabin." To choose the synset that contains words with a semantics that is closest to the original term, SenseLearner [14], a "word sense disambiguation" tool, is employed. This tool analyzes the mail text together with the previously calculated part-of-speech tags to determine the synset that is semantically closest to the original search word.

Replacing words. When SenseLearner is successful in determining a single synset, only words from this synset are considered as candidates for substitution. Otherwise, all synsets returned by WordNet are considered (although the substitution is less likely to be accurate).

The easiest strategy is to select that word among the candidates whose spam probability is the lowest. Another strategy is to randomly choose a word from the resulting synset(s). The latter approach aims to create diversity in the substitution process for a large set of mails. If a word is always replaced by the same word, the spam probability of this word would rise every time the mail is classified as spam. Variability in the substitution could slow down this process. Thus, we can select between *minimum* or *random* as replacement strategies.

Additional obfuscation. In the case that no word with a spam probability lower than that of the original word is found and the spam probability is very high, it is possible to exchange a single letter of the word with another character that resembles this letter (e.g. "i" with "1", "a" with "@"). This is an implementation of a trick from John Graham-Cumming's site "The Spammer's Compendium" [8] to conceal words from spam filters. Another threshold, called the *exchange threshold*, has to be defined that specifies for which words (and their corresponding spam probabilities) this obfuscation process should be invoked.

6 Evaluation

We evaluated our substitution attack against three popular mail filters: SpamAssassin 3.1.4 [17], DSPAM 3.8.0 [5], and Gmail [7]. For our experiments, we randomly chose 100 spam messages from Bruce Guenter's SPAM archive [10] for the month of May 2007.

In a first step, the header lines of each mail were removed, except for the subject line. This was done for two reasons. First, the header (except for the subject line) is not altered in a substitution attack, because it contains no words that can be replaced by synonyms. Second, retaining the header of the original spam mail would influence the result, because certain lines of the header could cause a spam filter to classify a message differently. In the next step, each HTML mail was stripped of its markup tags and converted into plain text. Then, we corrected manually words that were extended with additional characters or were altered in similar ways to escape spam filter detection (e.g., Hou=se – House). Finally, the resulting messages were processed by our prototype that implements the proposed substitution attack. In total, five different test sets were created, using different settings for the substitution and exchange thresholds as well as different substitution policies (as described in section 5):

Test Set A: no substitution, original header removed.

Test Set B: substitution threshold: 60%; exchange threshold: 95%; minimum replacement strategy.

Test Set C: substitution threshold: 60%; exchange threshold: 100%; minimum replacement strategy.

Test Set D: substitution threshold: 60%; exchange threshold: 100%; random replacement strategy.

Test Set E: substitution threshold: 80%; exchange threshold: 100%; minimum replacement strategy.

A threshold of 100% means that no character exchange or word substitution is performed. Test set A consists of the original messages for which no substitution was performed. Test sets B, C and D use an aggressive substitution policy, whereas test set E aims to preserve more of the original text. Test set C and D use the same threshold settings, but apply a different replacement strategy. The difference between test set B and C is that certain words in test set B are obfuscated.

Results. Using our five test sets, SpamAssassin and DSPAM were locally run to classify all mails in each set. In addition, all mails were sent to a newly created Gmail account to determine which of them Gmail would recognize as spam. SpamAssassin was used with its default configuration (where the threshold for classifying a spam is 5). However, note that we disabled SpamAssassin's ability to learn new spam tokens from analyzed mails. This was done to prevent changes in the results that depend on the order in which the tests were executed. Furthermore, SpamAssassin was not allowed to add network addresses to its whitelist. DSPAM was used in its standard configuration, with the exception that it was not allowed to use whitelists as well. Whitelisting is disabled to ensure that filters would never incorrectly let a mail pass as ham without first invoking the Bayesian analysis.

The results of the experiments are listed in Table 2. For each tested spam filter, the numbers show the mails that are incorrectly classified as ham (i.e., the mails that successfully penetrated the filter). At a first glance, the effectiveness of the substitution attack does not seem to be significant, especially for SpamAssassin and Gmail. Closer examination of the results, however, revealed that the overall effectiveness of the attack is limited because SpamAssassin use Bayesian analysis only as one component in their classification process. For example, SpamAssassin uses "block lists" that contain URLs that are associated with spam mails. In our test set, many mails do contain such links, and in some cases, a mail received more than 10 points for a single URL. In this case, the spam threshold of 5 was immediately exceeded, and the mail is tagged as spam regardless of the result that the Bayesian classifier delivers.

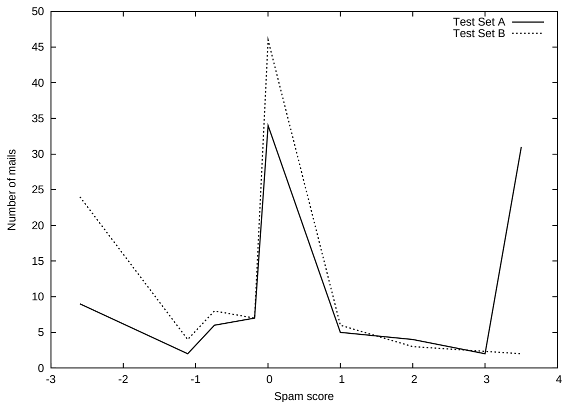

Figure 1: SpamAssassin: Bayesian spam scores.

To gain a deeper understanding of the effect of our attack on the Bayesian classifier of SpamAssassin, we examined the Bayesian spam score that is computed by SpamAssassin for the mails before (test set A) and after the most effective substitution attack (test set B). The results are shown in Figure 1. Note that the spam scores that are assigned to a mail by SpamAssassin are fixed values that range from -2.599 to 3.5. A negative score means that the content of the mail is regarded as ham, whereas a positive score implies that the mail is spam. Values around 0 are neutral that leave the classification of the mail to other mechanisms. In the figure, it can be seen that for the original test set A, only 10% of all mails had the lowest score of -2.599, while 30% received the highest spam score of 3.5. After the substitution attack (with test set B), 25% of all mails achieved a score of -2.599, while only 2% received 3.5 points. Also, the number of mails that were assigned a neutral spam score increased. This clearly shows the significant effect of the substitution attack on the Bayesian classification.

This claim is further confirmed when analyzing the results for DSPAM shown in Table 2. DSPAM is much more dependent on the results derived by the Bayesian filter when detecting spam, and thus, the number of spam mails that passed the filter could be more than doubled after the substitution process. To pass filters such as SpamAssassin (and probably also Gmail), the attacker also has to take into account other factors besides the

Mail set	Substitution threshold	Exchange threshold	Replacement strategy	Mails not recognized as spam by		
				SpamAssassin 3.1.4	DSPAM 3.8.0	Gmail
Test Set A	100%	100%	-	25	25	15
Test Set B	60%	95%	minimum	32	56	23
Test Set C	60%	100%	minimum	31	53	23
Test Set D	60%	100%	random	32	54	21
Test Set E	80%	100%	minimum	30	46	22

Table 2: Number of test spam messages not recognized by filters.

content (i.e., text) of his mail. For example, by frequently changing the URLs that point to the spammer's sites (or by hosting these sites on compromised machines), one could evade SpamAssassin's block list. In this case, the substitution attack is only one building block of a successful attack.

Number of possible substitutions. In addition to evaluating the effectiveness of a substitution attack, we also assessed the number of different versions that can be created from a single spam mail. For this, we analyzed the number of words for which substitution was attempted, as well as the number of possible synonyms for each word. When a substitution threshold of 60% was used, the system attempted to replace on average 36 words per mail. For these, an average of 1.92 synonyms were available, and in 23% of the cases, not a single synonym could be found. For a substitution threshold of 80%, 19 substitution attempts were made on average, with 1.65 available synonyms (and no synonym in 29% of the cases). Using a random replacement strategy, we also found that there are on average 992 variations of one mail.

Limitations. The substitution attack is effective in reducing the spam score calculated by Bayesian filters. However, the attack also has occasional problems.

One issue is that it is not always possible to find suitable synonyms for particular words. This is especially relevant for brand names and proper names such as "Viagra." In this case, one has to resort to obfuscation by replacing certain characters. Unfortunately for the attacker, spam filters are quite robust to simple character substitution. This can be observed when one compares the results for test set B (with obfuscation) with test set C (without obfuscation) in Table 2. Also, newly created words can be learned by spam filters, which counters the obfuscation or even raises the spam score of a mail [22]. Another problem for automated substitution are spelling errors in spam mails, which make it impossible to find the misspelled words in the thesaurus.

Another issue is that automated word substitutions are not always perfect. Natural language processing is a difficult task, and our tools are not always able to identify the correct role or semantics of a word. For example, WordNet yields "nexus" as replacement for "link." Other examples are "locomote" for "go" or "stymie" for "embarrass." We have invested significant effort to select precise replacements, but, unsurprisingly, the system fails sometimes. Moreover, the bad grammar used in many spam mails makes correct semantic analysis even more challenging. To mitigate this limitation, one could consider a setup in which the substitution system produces different versions of a particular spam mail that all have low spam probabilities. Then, a human can pick those alternatives that sound reasonable, and use only those for spamming. An example for a mail before and after word substitution is shown in Appendix A.

7 Conclusion

Spam mails are a serious concern to and a major annoyance for many Internet users. Bayesian spam filters are an important element in the fight against spam mail, and such filters are now integrated into popular mail clients such as Mozilla Thunderbird or Microsoft Outlook. Obviously, spammers have been working on adapting their techniques to bypass Bayesian filters. For example, a common technique for disguising spam is appending additional words to mails, with the hope of reducing the calculated spam probability. The effectiveness of such evasion efforts, however, varies, and Bayesian filters are increasingly becoming resistant.

In this paper, we present a novel, automated technique to penetrate Bayesian spam filters by replacing words with high spam probability with synonyms that have a lower spam probability. Our technique attacks the core idea behind Bayesian filters, which identify spam by assigning spam probability values to individual words. Our experiments demonstrate that automated substitution attacks are feasible in practice, and that Bayesian filters are vulnerable. Hence, it is important for service providers and mail clients to make use of a combination of techniques to fight spam such as URL-blocking, blacklisting, and header analysis.

Acknowledgments

This work was supported by the Austrian Science Foundation (FWF) under grant P18157, the FIT-IT project Pathfinder, and the Secure Business Austria competence center.

References

[1] ARADHYE, H. B., MYERS, G. K., AND HERSON, J. A. Image analysis for efficient categorization of image-based spam e-mail. In *Eighth International Conference on Document Analysis and Recognition* (2005).

[2] BOWERS, J. Bayes Attack Report. http://web.archive.org/web/20050206210806/www.jerf.org/writings/bayesReport.html,, February 2003.

[3] CipherTrust SpamArchive. ftp://mirrors.blueyonder.co.uk/sites/ftp.spamarchive.org/pub/archives/submit/.

[4] CUTTING, D., KUPIEC, J., AND PEDERSEN, J. A practical part-of-speech tagger. In *Third Conference on Applied Natural Language Processing* (1992), Xerox Palo Alto Research Center.

[5] The DSPAM Project. http://dspam.nuclearelephant.com/.

[6] Enron Email Dataset. http://www.cs.cmu.edu/~enron/.

[7] GOOGLE. Gmail. http://mail.google.com/.

[8] GRAHAM-CUMMING, J. The spammers' compendium. http://www.jgc.org/tsc.html.

[9] GRAHAM-CUMMING, J. How to beat an adaptive spam filter. In *MIT Spam Conference* (2004).

[10] GUENTER, B. Bruce Guenter's SPAM Archive. http://www.untroubled.org/spam/.

[11] KRAWETZ, N. Anti-Spam Solutions and Security. http://www.securityfocus.com/infocus/1763, 2004.

[12] LingPipe 2.4.0. http://www.alias-i.com/lingpipe/.

[13] LOWD, D., AND MEEK, C. Good word attacks on statistical spam filters. In *Conference on Email and Anti-Spam* (2005).

[14] MIHALCEA, R. Senselearner. http://lit.csci.unt.edu/~senselearner/.

[15] PRINCTON. Wordnet 2.1. http://wordnet.princeton.edu/, 2006.

[16] PU, C., AND WEBB, S. Observed trends in spam construction techniques: A case study of spam evolution. In *Third Conference on Email and Anti-Spam (CEAS)* (2006), p. 104.

[17] SpamAssassin. http://spamassassin.apache.org/.

[18] SULLIVAN, T. The more things change: Volatility and stability in spam features. In *MIT Spam Conference* (2004).

[19] THORYK, R. Tliquest spam archives. http://web.archive.org/web/20051104234750/http://www.tliquest.net/spam/archive/.

[20] WITTEL, G., AND WU, F. Attacking statistical spam filters. In *First Conference on Email and Anti-Spam (CEAS)* (July 2004).

[21] ZDZIARSKI, J. Bayesian noise reduction: Contextual symmetry logic utilizing pattern consistency analysis. In *MIT Spam Conference* (2005).

[22] ZDZIARSKI, J. *Ending Spam: Bayesian Content Filtering and the Art of Statistical Language Classification.* No Starch Press, 2005.

Appendix A: Example Substitution

This example shows the content of a mail before and after the substitution process. It can be seen that most words are substituted by a reasonable replacement, although using passing for loss is suboptimal.

Original Text

Subject: Take twice as long to eat half as much

I know it is the HOODIA that has made me **lose** weight. **Now** I am so confident I think I will try to do it a few more times and see where it gets me. I love the fact that I am getting weight **loss** results without any bad side effects like the other **products** that have stimulants in them. So I just had to write and give you my **testimonial** to say I am happy I **gained** my body back and since losing weight, I am **ready** to become more active and attractive than I have ever been. Thanks So Much,
Patricia Strate - Currently 137 **lbs**

Order online securely from our **website**
http://lk-hood.com
(A sample is **available** at no cost to you)
pls click the remove link at our **website**, and enter your id there

Text with spam words substituted

Subject: Take twice as long to eat half as much

I know it is the HOODIA that has made me **drop off** weight. **Instantly** I am so confident I think I will try to do it a few more times and see where it gets me. I love the fact that I am getting weight **passing** results without any bad side effects like the other **merchandises** that have stimulants in them. So I just had to write and give you my **testimony** to say I am happy I **derived** my body back and since losing weight, I am **quick** to become more active and attractive than I have ever been. Thanks So Much,
Patricia Strate - Currently 137 **pounds**

Order online securely from our **internet site**
http://lk-hood.com
(A sample is **usable** at no cost to you)
pls click the remove link at our **internet site**, and enter your id there