Die approbierte Originalversion dieser Diplom-/Masterarbeit ist an der Hauptbibliothek der Technischen Universität Wien aufgestellt (http://www.ub.tuwien.ac.at).

The approved original version of this diploma or master thesis is available at the main library of the Vienna University of Technology (http://www.ub.tuwien.ac.at/englweb/).

TECHNISCHE UNIVERSITÄT WIEN

VIENNA UNIVERSITY OF TECHNOLOGY

Diplomarbeit

Penetrating Bayesian Spam Filters Using Redundancy in Natural Language

Ausgeführt am

Institut für Rechnergestützte Automation
Arbeitsgruppe Automatisierungssysteme
der Technischen Universität Wien

unter der Anleitung von

Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Wolfgang Kastner
und
Privatdozent Dipl.-Ing. Dr.techn. Christopher Krügel
und
Privatdozent Dipl.-Ing. Dr.techn. Engin Kirda
als verantwortlich mitwirkenden Universitätsassistenten

durch

Günther Bayler
Am Wiesenweg 17
2403 Scharndorf

16. Oktober 2007 _____

Abstract

Today's attacks against Bayesian spam filters attempt to keep the content of spam emails visible to humans, but obscured to filters, or they attempt to fool the filters with additional good words appended to the spam. Another conceivable approach is to substitute suspicious words in spam emails with innocent words to make them appear as legitimate emails (i.e., ham emails). A precondition for the success of such an attack is that Bayesian spam filters of different users assign similar spam probabilities to similar tokens. In this thesis, it is examined whether this precondition is met; afterwards, the effectivity of a substitution attack is measured by creating a test set of spam messages that are classified by three different spam filters.

Zusammenfassung

Heutzutage übliche Attacken gegen Bayessche Spamfilter verwenden meistens eine von zwei verschiedenen Methoden: entweder wird der Inhalt von Spam-E-Mails so verändert, dass er zwar von Menschen, nicht aber von Spamfiltern gelesen werden kann, oder aber es werden unverdächtig aussehende Wörter zum Spam-E-Mail hinzugefügt, um die Spamfilter zu täuschen. Eine andere denkbare Variante, um Bayessche Spamfilter zu umgehen, ist, für Spamfilter verdächtige Wörter in Spam-E-Mails durch Synonyme dieser Wörter zu ersetzen, die für Bayessche Spamfilter unverdächtig sind, um zu erreichen, dass derart manipulierte Spam-E-Mails vom Filter nicht erkannt werden. Eine Voraussetzung für eine Attacke dieser Art ist, dass Bayessche Spamfilter von verschiedenen Usern gleichen Wörtern ähnliche Spam-Wahrscheinlichkeiten zuweisen. In dieser Diplomarbeit wird zuerst untersucht, ob diese Voraussetzung erfüllt ist. Danach wird die Effektivität einer Substitutionsattacke untersucht, indem 100 Spam-E-Mails, bei denen verdächtige Wörter automatisch durch unverdächtige Synonyme ersetzt wurden, durch drei verschiedene Spamfilter klassifiziert werden.

Contents

1 Introduction

1.1 Motivation

There are several reasons why I have chosen to write my diploma thesis about a topic out of the field of spam: Spam is a problem almost every email user is confronted with and annoyed by. The currently estimated percentage of spam in the total volume of email lies between 73.5 and 89.2 percent[1] [1, 2, 3], which is enormous in respect to the thus necessary oversizing of Internet connections and email servers. Another aspect of the spam problem is that most spam is completely useless: The *response rate*, that is, the percentage of spam receiver's that are interested in the advertised product and reply to a spam email in the total number of receivers of a spam email, is estimated between 0.0075 and 5 percent[2] [4, 5], depending on the product or service advertised. This implies that at least 95 percent of spam is sent to no avail. This is partly a general problem of advertising and unavoidable[3], but nevertheless many spam emails could be avoided but are not because this would mean an additional effort for their senders: since the cost for sending a single spam message is very low, for a spam sender it is more cost effective to send spam indiscriminately to as many receivers as possible than to go over the receiver address list to sift out receivers that are not interested in the advertised product, thereby achieving a higher response rate.
There are many different approaches against spam that aim to increase the costs for the spam senders in different ways, but there is currently no perfect approach. In the past, as soon as an efficient approach was invented and widely deployed, the senders of spam messages adapted their method of operation to this new conditions, so that the initially efficient anti-spam approaches lost their efficiency and had to be adapted by their inventors (a phenomenon called *arms race* in the anti-spam community). All of this makes spam an interesting field for research.

[1] Note that these figures are not scientifically assured, but estimations from different, potentially unreliable sources. There are no scientifically assured figures about the percentage of spam in the total volume of email since for an accurate estimation, it is necessary to filter large amounts of email, which is de facto possible only by large email (security) providers

[2] Also these figures are not scientifically assured. Without illegally sending unsolicited email, it is hard to estimate the response rate of spam.

[3] "Half the money I spend on advertising is wasted; the trouble is I don't know which half" – John Wanamaker, (attributed), US department store merchant (1838 - 1922) [6]

1.2 Problem Definition

1.2.1 The Network of Spam

The basic functionality of spam operations can be expressed in several discrete steps:

1. The spammer sends spam messages to the receivers.
2. The spam messages are transmitted to the receivers via the Internet.
3. The spam messages are received by the individual receivers.
4. Some receivers respond to the spammer because of the spam message.

First, the spam sender sends spam messages to many individual receivers, which is depicted by the thick arrow in figure 1.1. How does the sender know the email addresses of the receivers? He has to acquire them first, which is illustrated by the dot-dashed arrows in figure 1.1 and explained in section 1.2.2. Next, the spam emails are transmitted to the receivers via the Internet. Since, nowadays, the majority of email is spam, a significant part of the Internet's transmission capacity is wasted in this way. The transmission ends with the reception of the spam messages by the individual receivers. If the receiver uses a spam filter, the spam is either separated from the ham, for example, by moving it into a separate folder, or it is marked as spam so that the user can easily filter out marked messages for himself. Some users behave in a way that the spammer intends, that is, they respond to the spam message, usually by ordering an advertised product via a website (whose URI is given in the spam message) or via phone. Spam has a very low response rate, that is, of all the receivers of a spam message, only a few respond to it. The responses to the spam messages are illustrated in figure 1.1 with the thin arrow, that completes the *main cycle* in the *network of spam*.

1.2.2 Acquiring Email Addresses

There are different ways to acquire email addresses as targets for spam operations. Among the most important ones, there are: *(a) Harvesting* them from one of the services of the Internet, such as the World Wide Web (WWW). In this approach, the spammer uses a program called *harvester* to search the web for email address. A harvester is a certain type of a *web crawler* that starts its operations by scanning an arbitrary web page for email addresses and hyperlinks. The email addresses are stored in a list; this list is the harvester's output. The hyperlinks are used by the harvester to get to other web pages that can be scanned for email addresses and further hyperlinks in the same way as the starting page. Not only the web can be automatically searched for email addresses, but also other services of the Internet, such as the Usenet, are potential targets for harvesting operations. In the case of the Usenet, the harvester is a *Usenet bot* that is specialized to filtering email addresses out of Usenet posts.

Another way for a spammer to acquire email addresses is *(b)* by *buying* them from other Internet criminals, such as botnet operators or other spammers, an approach that is hinted to by the existence of advertisements for email address lists in spam emails. Since many spammers operate botnets, another possibility for them to acquire email addresses is *(c) stealing* them from the hijacked computers the botnet consists of. To do this, the spammer can either look specifically for address books of popular email clients, or scan all of the bot's disks for email addresses. It is also possible to trick users into entering their email addresses on *(d) fraudulent websites* to acquire email addresses for spam operations. Thinkable ways to do this are with greeting card sites or sites, where users can sign petitions. Another possible way for spammers to acquire email addresses is the *(e) dictionary attack*, where the spammer uses a list of persons names, for example, a phone book, to generate a list of possible email addresses. For the spammer, it does not matter if many of these generated email addresses do not work, because such addresses are skipped quickly when sending spam (see also section 3.2.6).

1.2.3 Specific Problems caused by Spam

There are several problems that arise because of the existence of spam. From an economic point of view, the main problem with spam is that it imposes costs to its receivers. The receiver has to pay for *(a)* the *transmission of spam* via the Internet, for *(b)* the *setup and maintenance of a spam filter*, if one is used. Furthermore, there are costs for *(c) deleting spam messages*, if there is no spam filter, and for *(d) deleting false negatives*, that is, spam emails that were not recognized by the filter.
Since spam filters do not work perfectly, there are not only false negatives, but also *false positives*, that is, ham emails that were erroneously classified as spam. For this reason, there are also costs for *(e) dealing with false positives*. These costs are usually larger than the costs for deleting false negatives: while it is relatively unproblematic to delete single spam emails that slip through a spam filter, false positives are often overlooked in the mass of filtered spam, which can lead to missed ham emails. Depending on the content of a missed email, the extent of the costs for a false positive differs: it can be anywhere between negligible (for emails such as daily horoscopes or the TV program) to threatening the existence of the receiver (for emails containing of orders, Invitations to Tender, or Calls for Papers).[4]
Besides the costs imposed by spam, many people perceive spam simply as *annoyance*. This is caused in part by the futility of spam: since spam usually is not focussed on specific target groups, most of the receivers are not interested in buying the products advertised in it, but instead, are bored by receiving large amounts of similar looking, but uninteresting emails. Some spam emails advertise pornography, which additionally is perceived as offending by many people.
The underlying cause for all these problems is that the amount of spam received by individual email users is too big. There are email accounts that are completely unusable due to the

[4] On the other hand, false positives are the prerequisite for the nowadays widely used excuse of claiming to not have received an email, because it was erroneously filtered as spam...

large amount of spam sent to them, which does not make sense for the spammers either, since nobody will read their spam on such accounts. If the total amount of spam sent was lower, the costs of spam would be lower too: eventually, there would not be the need for spam filters, and thus, the costs for dealing with false classifications would vanish. Maybe spam would not be perceived as such a big annoyance as it is today, because single advertisements that are focussed to target groups can be more interesting than annoying for their receivers, as it is the case with postal bulk mail, where the amount of mailings is manageable and bulk mail senders usually respect requests from receivers to be deleted from their address lists.

1.2.4 Weak Points in the Network of Spam

The spam problem can be attacked at several weak points in the network of spam. Every point of attack has its advantages and disadvantages, but there is no *Achilles' heel* in the network of spam, which could stop spam entirely.
The first section in the network of spam is to send spam messages to the receivers. Approaches against spam that aim at this point have the advantage that spam, that is never sent, needs neither Internet transmission capacity nor disk space at email servers. The disadvantage is that this is the point where the attacker has the least influence, since it is situated rather in the spammer's sphere of influence than in the attacker's.
The second section is the transmission of the spam messages via the Internet. Since a basic principle of the Internet is that transmissions do not always use the same path, approaches against spam aimed at this point are highly unpromising.
In the network of spam, the third section is the reception of the spam messages by the individual receivers. Since spam email is in principal not different from ham email, anti-spam approaches aimed at this point have to have the ability to discriminate between spam messages and ham messages, and are, therefore, called *filtering approaches*. The advantage of filtering approaches is, that the point of attack lies completely in the sphere of influence of the attacker, who has, thus, the freedom to filter his mail in any thinkable way. The disadvantage is, that when spam is filtered when it enters the receiver-side, the costs for its transmission and storage on an email server have already incurred and, thus, cannot be reduced with filtering approaches.
The main cycle in network of spam is completed with the responses of some receivers to the spam senders. These responses can be different, depending on the type of spam that is sent: for most spam messages, the response intended by the spammer is that the receivers should visit a website or call a phone number to buy the advertised product. Other spam messages try to lure the receiver into buying a certain share or voting for a certain political party. For both types of these spam messages, the sender does not need a website or phone number to achieve his goals. The advantage of attacks against this point in the network of spam is, that they can cause the most harm to the spammer since they are aimed at that point that was the reason for the spammer to send his messages in the first place, in contrast to anti-spam approaches such as the filtering approach, which is relatively unproblematic for spammers since there will

always be email users that do not use efficient filters. On the other hand, the problem with such attacks is that, since there are so many different types of spam, there is no single approach that works against all of them.
The *auxiliary cycle* in the network of spam starts with the publication of the email addresses from potential receivers of spam. This can happen in different ways: There are ways that are avoidable by the email user, such as, putting the email address in plain text on a web page or using it in Usenet posts. But there are also other ways that the email user cannot avoid, for example, the stealing of email addresses of hijacked computers, or dictionary attacks. For this reason, approaches against spam aimed at this point cannot stop spammers from acquiring email addresses completely. Nevertheless, email users should try to avoid the careless publication of their email addresses wherever it is possible.
The second section of the auxiliary cycle in the network of spam consist of the actual harvesting of the email addresses from different services of the Internet. As with the first section, since there are so many different ways to acquire email addresses for spamming operations, also approaches against this point cannot stop spammers completely. But for the individual email user, such approaches can be very effective, since if one users email address is protected from harvesting, spammers usually avoid that address in favor of other, unprotected addresses (see sections 3.4.5 and 3.4.4).

1.3 Goal

The goal of this thesis is to provide a basis for a further improvement of Bayesian spam filters. This is done by examining, whether current Bayesian filters can be penetrated by spam emails in which suspicious words are automatically replaced with synonyms of these words, that have the same meaning but are not considered suspicious by the filters. This implies that Bayesian spam filters of different users assign similar spam probabilities to similar tokens; the first subgoal of this thesis is to examine, whether this precondition is met. Afterwards, three different spam filters are used to examine how effective substitution attacks are, which is the second subgoal of this thesis.

1.4 Challenges of the Field

The field of spam is challenging especially for two reasons. First, there is the *constant evolution* of new ways spam senders try to circumvent current anti-spam methods. Every time a new anti-spam approach is devised, that is able to reduce the amount of spam received significantly, this new approach works only as long as spammers need to adapt their methods of operation to the changed environment. This arms race is limited only by the creativity of its opponents, that is, the spammers and the anti-spam community.
The other challenging aspect of spam is that, from a scientific point of view, it cannot be associated with exactly one discipline, but it is a field of application for several, different

disciplines; that is, it is an *interdisciplinary problem*. For example, Bayesian spam filtering, which is only one aspect in the field of spam, is a field of application for *information science* (more specifically, for *document classification*), which is in itself an interdisciplinary science with connections to *statistics*, *artificial intelligence*, *computational linguistics*, and its subfields *natural language processing* and *machine learning*. But this is only one aspect in the field of spam: there are also other aspects of this problem that fall into the social sciences, that is, disciplines such as *jurisprudence*, *sociology*, and *economics*. Considering the motivation of email users to respond to spam, even *psychology* could be counted to the sciences that have a connection to the spam problem.
While the above mentioned points are challenging for people working in the field of spam, at the same time they make the spam problem interesting: the arms race makes sure that there are continually new, exciting anti-spam approaches to study, while the interdisciplinarity gives incentives to view the spam problem from other, unfamiliar perspectives.

1.5 Organization of this Thesis

1.6 Terminology

Spam. General term used to describe unsolicited messages. Mostly used for email spam, but also for unsolicited messages over other communication channels and for automated ways of manipulating search engines. Examples for spam that is not email spam are: *(a)* Usenet Spam, *(b)* Spam over Instant Messaging (SPIM), *(c)* Search Engine Spam (Spamdexing), *(d)* Spam over Internet Telephony (SPIT), and *(e)* Spam over Mobile Phone (SPOM). Not to be confused with SPAM, which is a registered trademark for a canned meat product of Hormel Foods, LLC [7].

Ham. In the context of spam filtering, this is the opposite of spam, that is, all messages that are not unsolicited.

Bot (Web Robot, Internet Bot). A computer that is used to execute different tasks automatically over the Internet [8]. Usually a bot is not in the property of the botnet operator, but it is hijacked, that is, it is a computer connected to the Internet that is infected with malicious software so that it can be remote controlled. Usually the real owner of a hijacked computer does not know that his computer is hijacked. See *botnet*.

Botnet. A collection of several bots. The number of bots that are combined into a botnet varies from a few to several ten thousands. In the context of spam, botnets are used for different tasks, such as *(a)* as senders for spam, or *(b)* for collecting email addresses from the World Wide Web, Usenet, or from the local disks of the bot, or *(c)* for carrying out *Distributed Denial of Service*-attacks against other spammers or against anti-spam operations (see section 3.4.3).

CAPTCHA. An abbreviation for *Completely Automated Public Turing test to tell Computers and Humans Apart*. This term describes a computer program that can automatically generate different tests that "most humans can pass, but current computer programs can't pass." [9] CAPTCHAs are often used on websites of free email providers to prevent automated registrations for email accounts, since this was exploited in the past from spam senders to acquire new email address from which they could send spam. Examples for CAPTCHAs are distorted texts that the users have to read or simple arithmetical problems the users have to solve.

Denial of Service (DoS). A type of an attack against a computer that provides a service, such as a web server. The aim of a DoS-attack is, that the attacked computer's service is no longer available. This can be achieved for example by continuously sending malicious requests to the attacked computer, which is then more or less occupied with answering this requests, thus being unavailable for regular clients.

Distributed Denial of Service (DDoS). A more effective type of a DoS-attack. A DDoS-attack is carried out from several individual computers simultaneously, thus achieving a higher load of the attacked computer. In addition to this, DDoS-attacks are harder to repel than DoS-attacks from single computers.

DNS. An abbreviation for *Domain Name System*, the system that resolves names to IP addresses in the Internet.

DNS Blacklist (DNSBL). A blacklist that can be queried via DNS.

False Negative (FN). Spam that was not recognized by a filter, that is, erroneously classified as ham.

False Negative Percentage (FNP). In the context of spam filtering, this term denotes the proportion of spam emails that are not recognized as spam by the filter. The currently best spam filters have FNPs that are below 1 % [10].

False Positive (FP). In the context of spam filtering, this term describes ham that was erroneously classified as spam by a filter.

GUI. An abbreviation for *Graphical User Interface*, that is, a type of an user interface to a computer that does not work primarily with typed commands, but with graphical representations, such as icons and windows.

HTML. An abbreviation for *Hypertext Markup Language*, that is, the predominant page description language in the Internet.

ISP. An abbreviation for *Internet service provider*, that is, the organization that connects private users and companies with the Internet. Usually, not only the connection to the

Internet, but also other services, such as .hosting web pages and mail servers, are provided by an ISP.

Mail User Agent (MUA). A computer program for sending email to and receiving email from mail servers. Popular examples for MUAs are *Eudora*, *Evolution*, *Lotus Notes*, *Microsoft Outlook*, *Mozilla Thunderbird*, *Pegasus*, and *Pine*.

Mail Transfer Agent (MTA). A synonym for *mail server*.

SMTP. An abbreviation for *Simple Message Transfer Protocol*. The protocol that is used for transferring emails between computers on the Internet.

Token. In the context of spam filtering, an atomic element of an email with a separate spam probability. Single words are the most important tokens for spam filters, but there are also other types of tokens, such as tokens that represent word combinations.

Unsolicited Bulk Email (UBE). Email sent in bulk to many recipients at once, without their prior consent. Can have commercial content, but there is also non-commercial UBE, such as phishing emails, chain letters, or bulk emails from political parties or advocacy organizations.

Unsolicited Commercial Email (UCE). Email with commercial content, sent without the recipient's prior consent. Not necessarily sent in bulk.

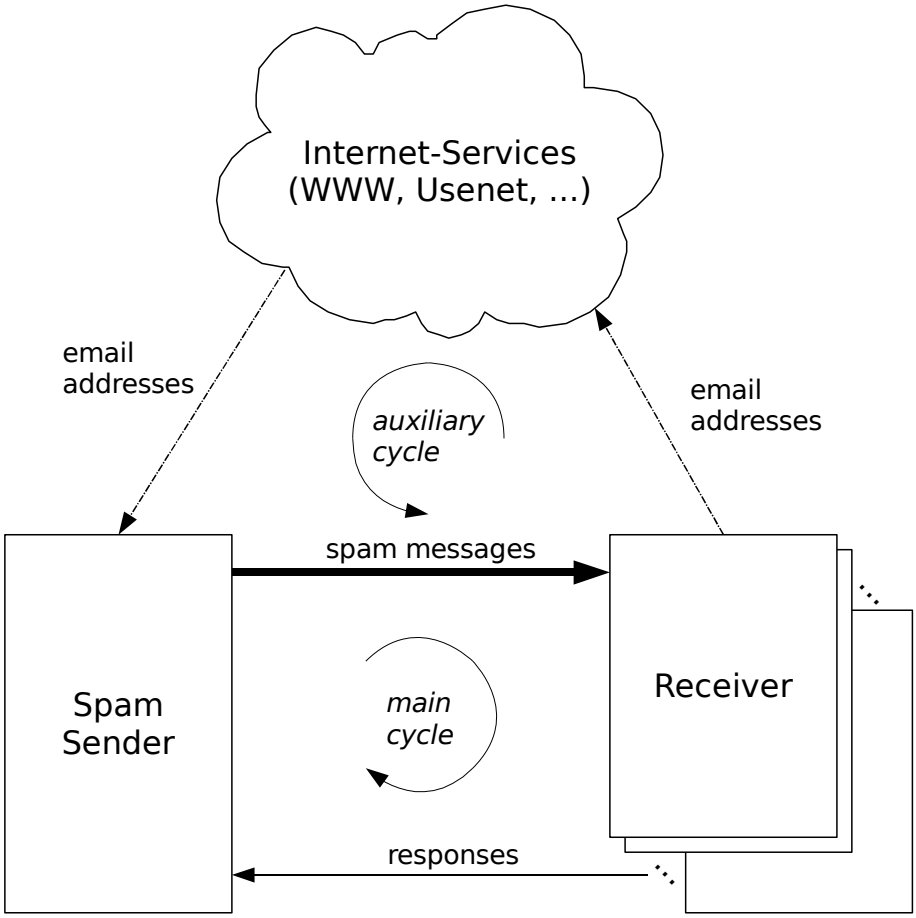
Uniform Resource Identifier (URI). A character string used for identifying a resource. If a URI can be additionally used for locating a resource, it is called *Uniform Resource Locator (URL)* [11].

Usenet. One of the services of the Internet, that enables users to post messages to and discuss with other users in theme-focused forums, called *Newsgroups*. Relevant in respect to spam for two reasons: *(a)* Newsgroups are a common target for spam, and *(b)* Spam senders acquire receiver email addresses among others from the Usenet.

Usenet Bot. A special type of a *bot* that operates on the *Usenet*. Usenet bots are usually used for automatically downloading pictures from the Usenet or similar tasks. In the context of spam, Usenet bots are relevant because they are used by spammers to harvest email addresses from the Usenet.

World Wide Web (WWW). One of the services of the Internet, that consists basically of hyperlinked documents. Nowadays often simply called *the web*.

Figure 1.1: The Network of Spam



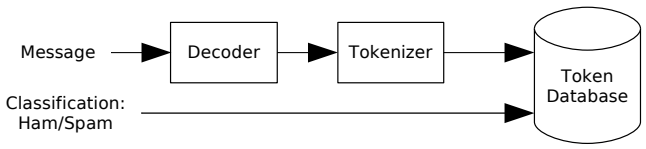
2 Bayesian Spam Filtering

2.1 Description of Bayesian Spam Filtering

The purpose of a spam filter is to decide whether an incoming message is legitimate (i.e., ham) or unsolicited (i.e., spam). One type of spam filter uses rules that specify suspicious properties of emails that indicate spam. Spam filters of this type are called *heuristic spam filters*. In contrast to heuristic spam filters, *Bayesian filters* do not have a fixed set of rules to classify incoming messages. This has an important implication: Bayesian spam filters have to be *trained* with known spam and ham messages before they are able to classify messages.

2.1.1 Training

Figure 2.1: Block Diagram: Bayesian Spam Filter - Training

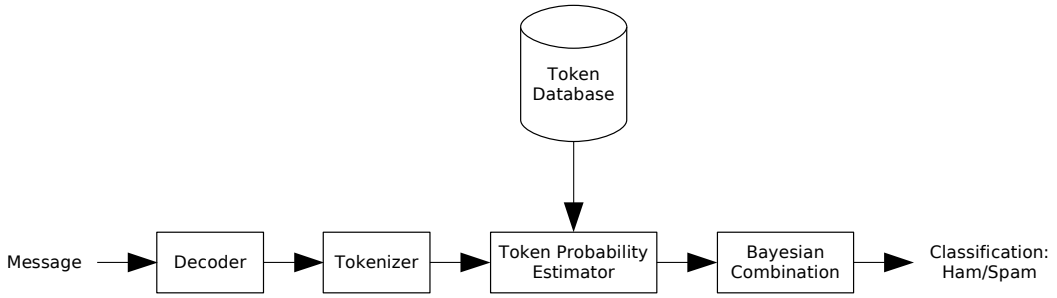


The training of a Bayesian spam filter is depicted in Figure 2.1: first, each message is stripped of any transfer encodings, such as quoted-printable or base64 encodings. The decoded message is then split into single tokens, which are mainly the words of which the message consists. Last, for each token T, a record in the token database is updated that contains two counts: the number of spam messages and the number of ham messages in which that token has been observed up to now ($n_S(T)$, $n_{\bar{S}}(T)$). Besides that, the token database also keeps track of the total number of spam and ham messages (n_S and $n_{\bar{S}}$) that have been used to train the Bayesian spam filter.

2.1.2 Classification

Figure 2.2 shows how a Bayesian spam filter classifies a message. Analogous to the training phase, first, the message is decoded and split into single tokens. For each token T, a spam

Figure 2.2: Block Diagram: Bayesian Spam Filter - Classification



probability is calculated from the number of spam and ham messages that have contained this token and the total number of spam and ham messages that have been used to train the Bayesian spam filter.
The *prior probability* of an email being a spam email is calculated from the proportion of spam emails in all emails used for training the filter:

$$P(S) = \frac{n_S}{n_S + n_{\bar{S}}} \tag{2.1}$$

The *token spam probability*, that is, the conditional probability that an email is spam if it contains a certain token T, is calculated as follows:

$$P(S|T) = \frac{\frac{n_S(T)}{n_S}}{\frac{n_S(T)}{n_S} + \frac{n_{\bar{S}}(T)}{n_{\bar{S}}}} \tag{2.2}$$

Note that in some spam filters alternative ways are used to calculate this probability; an overview can be found in [12, pp. 69–71].
Next, Bayes' theorem and the law of total probability are used to calculate the *posterior probability* of an email being spam by combining the spam probabilities of n tokens from the email. Some filters use all tokens for this step, while others use *feature pruning algorithms*. In the latter case, for example, only the n most interesting[1] tokens are taken into account, which speeds up the calculation.

$$P(S|T_1 \wedge \ldots \wedge T_n) = \frac{P(T_1 \wedge \ldots \wedge T_n|S)P(S)}{P(T_1 \wedge \ldots \wedge T_n)} =$$

[1]The *interestingness* of a token is defined as the absolute value of the difference between the token's spam probability and 0.5. [13]

$$= \frac{P(T_1 \wedge \ldots \wedge T_n|S)P(S)}{P(S)P(T_1 \wedge \ldots \wedge T_n|S) + P(\bar{S})P(T_1 \wedge \ldots \wedge T_n|\bar{S})} \tag{2.3}$$

In Bayesian spam filters, it is assumed that:

$$P(T_a) \perp\!\!\!\perp P(T_b) \quad \forall\, a \neq b \tag{2.4}$$

That is, all token occurrences are conditionally independent. This is called the *Naive Bayes assumption*. It is clearly not true that the tokens of an email occur indepent of each other, hence the "naive" in the name of this assumption, but, in spite of that, the classification algorithm works. More details about the Naive Bayes assumption and Bayesian classification in general can be found in the machine learning and computational linguistics literature, for example in [14, pp. 177–184] and [15, p. 237].
From assumption 2.4 follows:

$$P(T_1 \wedge \ldots \wedge T_n|S) = P(T_1|S) \cdot \ldots \cdot P(T_n|S) \tag{2.5}$$

$$P(T_1 \wedge \ldots \wedge T_n|\bar{S}) = P(T_1|\bar{S}) \cdot \ldots \cdot P(T_n|\bar{S}) \tag{2.6}$$

Insertion of 2.5 and 2.6 in 2.3 and further application of Bayes' theorem leads to

$$\begin{aligned} &P(S|T_1 \wedge \ldots \wedge T_n) = \\ &= \frac{P(T_1|S) \cdot \ldots \cdot P(T_n|S)P(S)}{P(S)P(T_1|S) \cdot \ldots \cdot P(T_n|S) + P(\bar{S})P(T_1|\bar{S}) \cdot \ldots \cdot P(T_n|\bar{S})} \\ &= \frac{P(S|T_1)\frac{P(T_1)}{P(S)} \cdot \ldots \cdot P(S|T_n)\frac{P(T_n)}{P(S)}P(S)}{P(S)P(S|T_1)\frac{P(T_1)}{P(S)} \cdot \ldots \cdot P(S|T_n)\frac{P(T_n)}{P(S)} + P(\bar{S})P(\bar{S}|T_1)\frac{P(T_1)}{P(\bar{S})} \cdot \ldots \cdot P(\bar{S}|T_n)\frac{P(T_n)}{P(\bar{S})}} \\ &= \frac{P(S|T_1) \cdot \ldots \cdot P(S|T_n)P(S)^{1-n}}{P(S|T_1) \cdot \ldots \cdot P(S|T_n)P(S)^{1-n} + P(\bar{S}|T_1) \cdot \ldots \cdot P(\bar{S}|T_n)P(\bar{S})^{1-n}} \end{aligned} \tag{2.7}$$

As Peters [16] points out, in some Bayesian spam filters, in particular those inspired by Graham's article "A Plan for Spam" [13], it is assumed that $n_S = n_{\bar{S}}$, that is, the emails used for training the filter consist in equal shares of spam and ham emails. In this case, $P(S) = 0.5$ and $P(\bar{S}) = 0.5$, which simplifies equation 2.7 further:

$$\begin{aligned} &P(S|T_1 \wedge \ldots \wedge T_n) = \\ &= \frac{P(S|T_1) \cdot \ldots \cdot P(S|T_n)0.5^{1-n}}{P(S|T_1) \cdot \ldots \cdot P(S|T_n)0.5^{1-n} + P(\bar{S}|T_1) \cdot \ldots \cdot P(\bar{S}|T_n)0.5^{1-n}} \\ &= \frac{P(S|T_1) \cdot \ldots \cdot P(S|T_n)}{P(S|T_1) \cdot \ldots \cdot P(S|T_n) + P(\bar{S}|T_1) \cdot \ldots \cdot P(\bar{S}|T_n)} \end{aligned} \tag{2.8}$$

Finally, the message is classified as ham or spam, depending on its posterior spam probability. Note that there are different ways to combine the token spam probabilities into a posterior spam probability of the whole email. An overview of these can be found in [12, pp. 74–80]. An alternative way as shown here was given by Robinson [17].

2.2 History of Bayesian Spam Filtering

The technology behind spam filtering is older than the spam problem, and even older than email itself: In 1961, Maron wrote a paper about automatically classifying documents [18]. He described an algorithm that assigns a document to one out of several categories based on the occurrence of *clue words*, that is, words that indicate that a document belongs to a certain category. Maron assumed the conditional independence of different clue words to facilitate the calculations, similar to the Naive Bayes assumption used in Bayesian spam filters. He concluded that "It is feasible to have a computing machine read a document and to decide automatically the subject category to which the item in question belongs."

One of the first email facilities was developed four years later, in 1965, for the Compatible Time-Sharing System (CTSS), a timesharing operating system for the IBM 7094 mainframe computer, at MIT [19]. Back then, computers usually were not connected with each other. This changed with the implementation of the *ARPANET*, the predecessor of the Internet, which started in 1969 [20]. In 1971, Tomlinson extended a single-computer email program so that messages could be sent to users of other computers, if the computers were connected over the ARPANET. These messages were the first *network emails* [21]. Unsolicited messages were at that time not a problem, although only four years later in 1975, Postel wrote RFC 706 "On the junk mail problem" [22]. It describes a blacklisting approach (see section 3.2.1) against unwanted email from malfunctioning hosts.

According to Templeton [23], possibly the *first spam email* was sent in 1978 by a marketing representative of DEC to 593 recipients. It contained an advertisement for the presentation of a new computer system (see Appendix A). Many recipients were annoyed by this email, and since it violated the usage policies of the ARPANET, the Defense Communications Agency, which is the organization that operated the ARPANET, filed a complaint against the sender of this spam email. This happened three years before the creation of SMTP (Simple Message Transfer Protocol) by Postel [24] and five years before the introduction of the Domain Name System (DNS) by Mockapetris [25, 26].

With the exponential growth of the Internet in the 1990s, also the spam problem growed exponentially. In 1996, *ifile*, the first spam filter that used the Naive Bayes algorithm was developed by Rennie [27, 28]. In 1998, an important paper about Bayesian spam filtering was written by Sahami et al. [29]. In spite of this paper, Bayesian filters were not very popular until 2002, when Graham published *A Plan for Spam* [13], which was discussed heavily on Slashdot [30]. After this publication, several Bayesian spam filters were developed, such as *SpamProbe* by Burton [31], *SpamBayes* by Peters et al. [32], *Bogofilter* by Raymond [33], *POPFile* by Graham-Cumming [34], and *DSPAM* by Zdziarski [35].

3 Bayesian Spam Filtering in the Context of Other Anti-Spam Approaches

Bayesian spam filtering is only one possibility to counter the spam problem. To be able to assess its advantages and disadvantages, it is necessary to compare it with other approaches against spam. Up to now, there is no single solution against spam that works perfectly; each has its own advantages and disadvantages.
In this chapter, the approaches against spam are categorized according to its working points. There are some anti-spam approaches that focus on the origin of spam. These are called *sender-side approaches* and have the advantage that spam, that is stopped before it is ever sent, cannot cause useless traffic and allocate email server disk space unnecessarily. *Receiver-side approaches* focus on the endpoints of spam. Compared with sender-side approaches, this has the advantage that the receiver has total control over his side, while the sender-side lies out of his sphere of influence. *Combined approaches* take place on the sender- and on the receiver-side. Approaches that do not focus directly on spam are subsumed under *other approaches*.

3.1 Sender-Side Approaches

3.1.1 Legal Approach

The spam problem arose because with the advent of the Internet it was possible to send emails worldwide without having to pay postage for each message. This situation was unprecedented, thus the then existing laws were not applicable to unsolicited email messages. New laws had to be created; important examples are the *CAN-SPAM Act of 2003*, which governs the use of unsolicited commercial email messages in the United States of America, and the European Union's *Electronic Communications Privacy Directive of 2002*.

Example 1: CAN-SPAM Act of 2003

The CAN-SPAM Act of 2003 [36] allows unsolicited commercial emails, if they comply to certain rules. The most important requirements are that the emails contain *(a)* a label identifying the message as advertisement, *(b)* opt-out instructions, *(c)* the sender's legitimate postal address, *(d)* valid header lines (including a valid subject line), and *(e)* a warning label if the email contains sexually oriented material. The CAN-SPAM Act prohibits specifically *address*

harvesting, that is, automatically searching web sites for email addresses, and *dictionary attacks*, that is, automatically generating receiver email addresses from names, letters, or numbers.
While the CAN-SPAM Act of 2003 is often criticized for being to mild, a bigger problem is that it is ignored by most spammers. MX Logic, an email and web security services provider, reports an average rate of compliance with the CAN-SPAM Act of 2003 of 0.45% of all spam emails for 2006 [37].

Example 2: Electronic Communications Privacy Directive of 2002

The Electronic Communications Privacy Directive of 2002 [38] is stricter than the CAN-SPAM Act of 2003. The most important rules are that *(a)* prior explicit consent of the recipients of unsolicited commercial messages is needed ("opt-in"), except a business relationship between the sender and the receiver exists already, *(b)* every direct marketing message has to contain opt-out instructions to give the recipient the opportunity to refuse further messages, and *(c)* the use of false identities or false addresses is prohibited.

Problems

The main problem of the legal approach is that it is only seldom possible to execute the anti-spam laws and punish the senders of spam email. Much spam comes from abroad or it seems to come from abroad because the spam senders use servers in foreign countries, which makes prosecution of them nearly impossible for the legal authorities. Spam is global phenomenon that is hard to fight with the legal systems which are responsible only for single countries.

Related Work

An overview of current spam laws can be found on Sorkin's website [39]. The European Union's anti-spam legislation is discussed by Lugaresi [40]. Moustakas, Ranganathan, and Duquenoy [41] compare the United States' and the European Union's spam laws.

3.1.2 Opt-Out

The idea of *opt-out* is that the receiver of unsolicited messages has to contact the sender and inform him that no more messages are desired in the future. While being necessary and desirable in electronic mailing lists, where users have to opt in to receive messages upfront, this method does not work with spam messages, because many spammers do not respect the wish of the user to get no more emails in the future, but, in contrary, take the opt-out-request as indication of the validity of the email address and send even more spam afterwards. In spite of this problem, opt-out is proposed in the CAN-SPAM Act of 2003 (see section 3.1.1).

3.1.3 Robinson Lists

The idea of *Robinson lists* is based on the opt-out principle. In the classical opt-out approach, each sender of bulk messages maintains a list of receivers who do not want to receive bulk messages. Every time before a bulk message is sent, receivers who have opted out from receiving bulk messages are removed from the list of target addresses, so that the message is sent only to receivers who have not opted out. This is cumbersome for the receivers, since they have to contact each sender of bulk messages independently to deposit their wish of not getting bulk messages.
A better alternative are Robinson lists, that is, opt-out lists that are kept commonly for all senders of bulk messages. For postal mail, these lists are typically maintained by the national direct marketing associations. Although this approach works for postal mail, it does not work for email for several reasons: Postal bulk mail is targeted to a certain geographical area that encompasses at most the area of a nation. This is useful for the sender, since far-distant receivers are not likely to respond to marketing messages anyway, and the postage increases with the distance to the receiver. While the former is equally true for email, the latter is not: there is neither postage for email to far-distant receivers nor for email to near ones. Senders of unsolicited emails have, therefore, little financial incentive to target their messages to a certain geographical area (which, besides, would be more difficult for email than for postal mail, since email addresses that end in a generic top-level domain cannot be allocated to a geographical position easily). Thus, bulk emails can potentially reach mailboxes from all over the world. While the sender of postal bulk mail has to consider the Robinson lists of his local direct marketing associations only, for email bulk messages the sender would have to take all Robinson lists worldwide into account. This is too complex and costly compared with the alternative of ignoring Robinson lists and trying to avoid complaints by disguising the sender address of the emails.
Alternatively, a global Robinson list is thinkable. But apart from the technical challenge a global Robinson list would pose, an open question is who should be responsible for and bear the costs of maintaining such a list. And besides that, it is not clear upfront whether spammers would respect a global Robinson list and use it for cleaning their mailing lists, or abuse it to find out valid email addresses.

3.1.4 Port 25 Blocking

Port 25 is the standard TCP port of SMTP. Some Internet service providers (ISPs) block connections from their customers via port 25 to all servers except their own email servers. This has the effect that sending email directly from a email server on a client, thus avoiding the ISP's email server, is impossible; instead, the ISP's email server has to be used. The advantage of this approach is that spambots, that is, virus- or worm-infected computers in the Internet that are abused for sending spam, are sabotaged by this measure. The disadvantage is that users are totally dependent from their ISP's email server in this approach. They cannot

run their own email server, because this would require connections to port 25. Since many users do not have this need, for an ISP a sensible compromise could be to block outgoing port 25 traffic from their customers per default and activate it only on request. Important examples for ISPs that use port 25 blocking are: AT&T, Microsoft Network (MSN), EarthLink, and Verizon.

3.2 Receiver-Side Approaches

3.2.1 Blacklisting

One of the earliest approaches against spam was *blacklisting*, that is, maintaining a list of unwanted senders. As early as in 1975, Postel described a blacklisting approach against spam in RFC 706 "On the junk mail problem" [22] (see also section 2.2).

Email Address Blacklisting

In the beginning of the spam problem, senders of spam emails could be identified by their email addresses. To avoid receiving spam, it was sufficient to maintain a blacklist, that is, a list of known spam senders, and to discard all emails whose sender address was on that blacklist. In reaction to this, spam senders proceeded to using forged email addresses, which is possible very easily because the sender email address is not validated in SMTP. To evade email address blacklists, senders of spam emails generate new forged sender addresses for each spam run.

IP Address Blacklisting

Forging the IP address of the computer that sends spam is not as trivial as forging the sender email address. For this reason, it is better to use the IP address to identify senders of spam. There are several IP address blacklists; important examples are:

- Spamhaus Block List (SBL) [42]
- Spam Prevention Early Warning System (SPEWS) [43]
- Spam and Open Relay Blocking System (SORBS) [44]
- Not Just Another BlackList (NJABL) [45]
- SpamCop Blocking List (SCBL) [46]

Most IP address blacklists can be queried via DNS and, thus, are called *DNS blacklists* or, abbreviated, *DNSBLs*.

The problem with this approach is that nowadays, spam is usually not sent from individual email servers, but mostly via *botnets*, that is, computers that are infected by malicious software that allows them to be remote controlled via the Internet. Botnets can consist of up to several ten thousand infected computers, of which many can have dynamically assigned IP addresses. For this reason, it is increasingly difficult to keep IP address blacklists up to date.

URI Blacklisting

The idea of *Uniform Resource Identifier (URI) blacklisting* is to detect spam via suspicious domain names. These can be found either in URIs in the spam email itself, or in the host part of the message's envelope sender address. Important examples of URI blacklists are:

- Spam URI Realtime Blocklists [47]
- Realtime URI Blacklist [48]
- Abusive Hosts Blocking List [49]
- RFC-Ignorant.Org [50]

URI blacklists are also called *right-hand side blacklists* (RHSBLs), because the domain name is in the host part of the message's envelope sender address, which is on the right-hand side of the email address.

3.2.2 Whitelisting

A *whitelist* is a list of allowed email senders. In a whitelisting approach, an email is delivered to the recipient only if its sender address is on the whitelist. All spam emails are blocked in this approach, unless the sender addresses of a spam email is on the whitelist. False positives can occur only if the sender address of a spam email is forged with an email address that is on the whitelist, which can happen by accident or intentionally.
Filtering spam with whitelists has a major disadvantage: it is impossible to get in contact via email with somebody who uses whitelisting, if the sender is not on the whitelist. For this reason, a pure whitelisting approach is unemployable for receivers that have to be reachable for unknown senders, such as helpdesk technicians, journalists, and scientists.
A useful application for a pure whitelisting-only approach is filtering children's email, where parents want to have full control over the email contacts of their children. Besides this, whitelisting is a useful extension of other filtering mechanisms that allows messages from known ham-senders to bypass other filters and, therefore, reduce the risk of false-positives.
Big email providers sometimes use *paid whitelists*. If large-scale bulk email senders want to make sure that their email is delivered to users of these email providers, they have to pay a fee to be put on the whitelist. Only commercial senders of bulk messages that are not unsolicited

are accepted for paid whitelists. This is ensured through an accreditation process each applicant firm has to undergo, where each firm's history is checked. If there are indications that an applicant firm has sent unsolicited bulk email (UBE) in the past, its application is refused.
Important examples for email providers that use paid whitelists are AOL and Yahoo!, which both use *CertifiedEmail* from Goodmail Systems [51], as well as MSN, Hotmail, and Roadrunner, which use *Sender Score Certified* from Return Path [52].

3.2.3 Greylisting

A email server that uses *greylisting* checks, what a SMTP-client that connects to it does when it encounters a temporary error from the server. The desired behavior that is defined in SMTP, more specifically in RFC 2821 [53, section 4.5.4.1], is that a client should repeatedly retry sending the email after certain timeout periods. Most regular SMTP-clients obey these rules, in contrast to SMTP-clients that are used for sending bulk email, which usually do not retry if they cannot submit their spam email immediately. The reason for this non-RFC-conform behavior is that for a spam sender, the costs for retrying to send an email (which consist mainly of keeping a list of email servers that are temporary not available) are not justified by the gain of this behavior: instead of investing time and memory into a certain recipient on the address list to ensure the delivery of the spam email, it is more efficient to skip such recipients and send spam to other, easier reachable recipients instead.
A disadvantage of this approach is that emails from unknown senders are delayed, regardless of whether the email is spam or not. To avoid delaying all emails, greylisting is usually combined with whitelisting (see section 3.2.2): if a SMTP-client retries to send email in response to a temporary error, it is automatically added to a whitelist. If, later, the same client connects to the server again, it is recognized and its emails are delivered without an additional delay.
One could argue that spam senders could use RFC-compliant SMTP-clients to avoid being filtered by greylisting. This is principally true, but greylisting is not entirely useless in this case. Indeed, the spam emails cannot be filtered with greylisting if spammers use RFC-compliant retry strategies, but the delivery of spam emails is delayed. Eventually, the spam sender's IP address is put on a blacklist (see section 3.2.1) in the time between the first and the second sending attempt; if IP blacklists are checked before finally delivering the email, spam emails can be filtered anyway. It is also thinkable that the spam sender does not get around to retrying to send spam to servers that are temporary not available, because, for example, the ISP of the computer that is used for sending spam cut that computer's Internet connection because of complaints before a second attempt to send the spam.

3.2.4 Hash-Based Methods / Collaborative Filtering

The nowadays used *hash-based methods* are implemented as client-server systems, where the servers are interconnected to create a bigger, distributed system. They work as follows: For each incoming email, the client generates an unique identifier, usually a hash sum. To find

out, whether or not that email is spam, the client sends a query containing this hash sum to the server, which has a database of hash sums of different spam emails. According to the information in the database, the server answers the client's query, which can then deliver or block the email.

Hash-Based Filters without User Input

One variant of this concept is, that the client sends hash sums of all received messages to the server. The server counts the number of queries it receives for each hash sum. If the number of queries for a certain hash sum is above a threshold, it can be assumed that the corresponding message is sent in bulk. This variant has the disadvantage that it cannot differentiate between legitimate and unsolicited bulk messages. For this reason, it is necessary to combine it with a whitelisting approach, where, for example, email from mailing lists are excepted from the spam filtering process. An important implementation of this variant is the *Distributed Checksum Clearinghouse (DCC)* [54].

Hash-Based Filters with User Input (Collaborative Filtering)

Another variant of hash-based methods is an application of *collaborative filtering* [55]: the clients send only hash sums of known spam emails to the server. In this variant, the hash sum database of the server contains only hash sums of spam emails; if the hash sum of an unknown email is in the database, the email is spam. The advantage of this variant is, that legitimate bulk emails are not detected erroneously as spam. A disadvantage is, that it implicitly relies on the assumption that all users are consistent in their classification of spam and ham, which is not necessarily true: one user's spam can be another user's ham. Another disadvantage is, that the system is susceptible to intentionally false spam reports on legitimate bulk emails by malicious clients, but this can be mitigated by the application of reputation systems. Important implementations of this variant of hash-based methods are *Vipul's Razor* [56], its commercial derivative *Cloudmark* [57] and *Pyzor* [58].

Attacks against Hash-Based Filters

Spam senders try to avoid being filtered by hash-based methods by randomly making small changes to their emails, so that not all copies of a spam email are identical. Thus, different hash sums will be created for the different versions of a spam email by hash-based methods. This attacks are countered by implementations of hash-based methods that are insensitive to small alterations of messages. This can be achieved, for example, by using feature selection algorithms, which is discussed in more detail by Kołcz, Chowdhury and Alspector [59].

3.2.5 Heuristic Filters

Heuristic filters try to identify spam by heuristics, that is, common-sense rules that describe different characteristics of spam emails. The more rules apply for an email, the higher is the probability that this email is spam. Typical examples for rules are *(a)* checks for phrases that occur often in spam (such as "One hundred percent guaranteed", "Full refund", or "Click here"), *(b)* checks for common spam obfuscation techniques (such as altering words by replacing certain characters with numbers, or different HTML- and plain text parts in an email), and *(c)* checks for unusual headers (such as non-matching timestamps in `Date:`- and `Received`-lines).
Since heuristic filters are widely used, spammers try to avoid being filtered by them by creating spam messages that do not trigger tests used in heuristic filters. There is even a bulk email tool that contains a copy of an open-source heuristic filter, so that creators of spam emails can optimize them to not trigger its tests [60,61]. Because of this, the rules of heuristic filters have to be updated often to avoid an increase of the filter's false negative percentage (FNP), which is a certain disadvantage of heuristic filters.
An advantage of heuristic filters compared with Bayesian filters is, that they can filter email without having to be trained first, since the rules used by them are static. All heuristic filters are equipped with a basic set of rules that enables them to filter email immediately after installation.
An important implementation of a heuristic filter is *SpamAssassin* [62]. SpamAssassin is not a purely heuristic filter, but rather a combination of many different anti-spam techniques. Besides its heuristics, it implements, for example, *(a)* a Bayesian filter (see Chapter 2), *(b)* blacklisting (see section 3.2.1), *(c)* whitelisting (see section 3.2.2), *(d)* the validation of tokens from computation approach schemes, such as *Hashcash* (see section 3.3.2), and *(e)* email sender address authentication schemes, such as SPF and DKIM (see section 3.4.1).

3.2.6 SMTP Tarpits

In computer security, *TCP tarpits* were originally invented to slow down portscans by computer worms [63]. The idea is that malicious connection attempts to unused IP addresses are detected and answered by a computer that monitors a subnet, the answers being intentionally delayed. This computer respective the program that answers connection attempts to unused IP addresses is called *tarpit*. The computer worm cannot differentiate between real answers from existing servers and forged answers from the tarpit, and, therefore, wastes time sending packets to non existing servers and waiting for answers. This reduces its spreading efficiency.
SMTP tarpits implement this concept for email servers. Connections from malicious SMTP clients are artificially slowed down with the intention to waste as much resources (time, memory, outgoing ports) of spammers as possible. To differentiate malicious from benign SMTP clients, different methods are used: some implementations use greylisting (see section 3.2.3), others are combined with heuristic filters (see section 3.2.5), and some tarpits do not have to

make this differentiation because they are set up exclusively for spam emails.
In contrast to greylisting, whose main purpose is to filter spam, the original intention of SMTP tarpits is to harm spammers. This can be seen in the implementation of the delay in the two concepts: while in the greylisting approach the delay of the connection is implemented by returning a temporary failure after the `RCPT TO:`-command (after which the client can terminate the connection and make connections to other servers), SMTP tarpits slow down their answers to to the email sender so that they cannot terminate the connection, but have to stay connected until every part of the answer is transmitted. The answers are delayed either on the transport layer or on the application layer.
Delaying answers on the transport layer, which is also known as *TCP damping*, is carried out by delaying TCP acknowledge packets or forging network congestion. Li, Pu, and Ahamad [64] examined this variant of the SMTP tarpit concept and found that it can harm spammers if enough receivers make use of it. Furthermore, they point out that "TCP damping gives incentive to spammers to develop ways to clean up their lists rather than blindly increasing their size".
Answers on the application layer (for email, that is the SMTP layer) can be delayed for example by inserting artificial pauses between each line of multiline replies to SMTP commands. Eggendorfer [65] examined the efficiency of SMTP tarpits, among others of those that make use of delays on SMTP level, and found that they mostly do not accomplish their goal of harming spam senders. The reason for this is that bulk email software is usually capable of multithreading and, thus, can send spam to other receivers while being delayed by a tarpit. Further on, many senders of bulk email evade tarpits by terminating the connection very quickly, instead of implementing timeouts (for example, the minimal timeouts suggested in RFC 2821 [53, section 4.5.3.2]). If this kind of evasion is applied, spam senders do not wait for the answer of the tarpit, and, thus, are not harmed by the tarpit. For this reason, SMTP tarpits set up solely for harming spammers do not fulfill their purpose. On the other hand, spam email of SMTP-clients that terminate their connection to the server very quickly is not delivered. For this reason, SMTP tarpits installed additionally to email servers are useful for filtering spam, in a similar way as greylisting is.
Examples for SMTP tarpits are *(a)* Lamb's *TarProxy* [66], *(b)* Donnerhacke's *Teergrube* [67], *(c)* Grosse's *SMTarPit* [68], *(d)* Rehbein's *Teergrube* [69], and *(e)* Mailchannels' *Traffic Control* [70]. A tarpit feature can also be found in *(f)* Microsoft's SMTP servers [71].

3.3 Combined Approaches

3.3.1 Email Postage

The idea of *email postage* (also known as *sender pays*) is to accept emails only if the sender paid a certain postage upfront, as it is usual in traditional postal mail since the implementation of the postage stamp in 1840. If every message costed the sender a certain, little amount of money, sending messages in bulk would be prohibitively expensive.

As proof that the postage was paid by the sender, each message carries a *stamp* in form of an unique token in the header. These stamps are issued by a central authority, which is also responsible for ensuring that each stamp is used only once.
There are several problems with this approach. Implementing a system for issuing and validating stamps is not trivial due to the large volume of email sent. Such a system would have to be bigger than the credit card system, as Levine has shown [72]. If such a system existed, it would be a "an appealing target for a denial-of-service attack", as Abadi et al. have found [73]. Besides, email postage would be a problem for electronic mailing lists, as Templeton points out [74], because mailing list traffic is in principle indistinguishable from traffic caused by spam. But probably the biggest problem is that email postage will work only if a significant amount of all Mail User Agents (MUAs) use this system. On the other hand, there is little incentive for software producers of one MUA to implement a email postage system if the other MUAs do not use their system, a situation which is known in game theory as *prisoner's dilemma*.

3.3.2 Computational Approach

An interesting variant of email postage is, instead of requiring senders of email to pay a certain amount of money, to require them to compute a mathematical problem, for example, a hash collision, that needs a few seconds of computing time to solve, but can be validated easily afterwards. A central authority which issues and validates stamps is not necessary in this variant, since participants of this system can generate and validate stamps for themselves.
This so-called *computational approach* was first suggested by Dwork and Naor [75] and later experimentally implemented as *Hashcash* by Back [76] and as *Camram* by Johansson [77]. Its biggest advantage compared to the email postage approach is that there is no central authority which would be difficult to build, maintain, and defend against denial-of-service attacks. An often raised objection against the computational approach is that it would be possible for spammers to compute the mathematical problems with botnets, thus undermining the system. This is true to a certain extent, but even if every computer in a botnet would be used this way, it would not be possible to send as much spam as is sent nowadays, as Johansson points out on his web page [77].

3.3.3 Challenge / Response (C/R)

Challenge/response (C/R) is an approach against spam that works as follows: Each email that is sent from an previously unknown sender is automatically answered with a challenge, that is, a simple calculation or a CAPTCHA[1] to find out, if the sender of the email is human or a spam bot. If the challenge is answered correctly, the sender gets on a whitelist and the email

[1] "A CAPTCHA™ is a program that can generate and grade tests that most humans can pass, but current computer programs can't pass." [9] Often the human has to read distorted texts to pass the test, which is impossible for a computer.

sent before is delivered. All emails from this sender can then pass the spam filter without subsequent tests (see section 3.2.2).
An advantage of this method is that, theoretically, false negatives occur only if the spam senders answer the challenges correctly, which can be ruled out since this would be too much effort for them. Disadvantages of C/R are: *(a)* Since sender addresses of spam emails are usually forged, the challenges are often sent to innocent third parties. This is annoying for them, and if they answer the challenge correctly, because, for example, they do not understand, why this email was sent to them, false negatives can occur. *(b)* A certain amount of ham email is machine-generated, for example, confirmation emails from booking systems or emails from mailing lists. Since senders of these emails cannot respond to each particular challenge, it is necessary to put them on a whitelist when using C/R. *(c)* Sometimes, humans cannot respond to challenges. For example, visually impaired persons can have difficulties reading distorted text, which is an often used CAPTCHA. In this case, the human has to ask the receiver (using other communication media) to be put on the whitelist before being able to send email, which is cumbersome. *(d)* C/R slows down email from previously unknown users.

3.4 Other Approaches

3.4.1 Email Sender Address Authentication

One weakness of SMTP is that the client's email address is not verified by the server, which is mentioned in RFC 2821 [53, sec. 7.1]:

> SMTP mail is inherently insecure in that it is feasible for even fairly casual users to negotiate directly with receiving and relaying SMTP servers and create messages that will trick a naive recipient into believing that they came from somewhere else. Constructing such a message so that the "spoofed" behavior cannot be detected by an expert is somewhat more difficult, but not sufficiently so as to be a deterrent to someone who is determined and knowledgeable.

Forging sender email addresses is, therefore, usual in spam emails. Sender addresses are forged in spam emails to circumvent email address blacklists (see section 3.2.1), and to avoid having to deal with complaint emails (see section 3.4.2). From the spammer's point of view, it is, therefore, sufficient to use random-generated sender addresses, that do not exist at all. This is a little bit different for authors of phishing emails: to increase the credibility of their emails, it is more suitable for them to use sender addresses of organizations they want to impersonate in their attack. This can be avoided by sender address authentication schemes. Since it is possible for spammers to use sender address authentication for their own domains, it cannot prevent spam in general [78].
There are several different ways to authenticate an email sender address. One could use *email encryption standards* such as S/MIME [79, 80] or OpenPGP [81], but there are some disadvantages in doing this: the email body is altered when the email is signed with one of these

methods, and there is some effort from the users necessary since they have to set up and distribute their public keys. Another way for email sender address authentication is to use *sender address authentication schemes*. These schemes avoid the problems mentioned above: they are set up and administered per domain, and they do not alter the email's body, but only its header lines. Three important sender address authentication schemes are introduced in the following.

Example 1: Sender Policy Framework (SPF)

The idea of *Sender Policy Framework* [82, 83] (SPF; formerly known as "Sender Permitted From") is to specify, which IP addresses are used for sending email in a domain. This specification is stored in the DNS, more precisely, in the SPF record of a domain. Receivers of emails can check, whether the IP address from which the email is sent matches the IP address which is defined in the SPF record of the sender domain. The sender domain is derived from the email's *envelope address* (the sender mailbox address specified via the SMTP-command `MAIL FROM:`), or the domain name the SMTP client declared via the `EHLO` or `HELO` command. If the sender IP address does not match the IP address defined in the SPF record, the receiver can assume that the sender address of the email is forged and reject the email.
A disadvantage of SPF is that it breaks *MTA level email forwarding*, that is, forwarding of an email by rewriting the email's envelope address, because in this case the IP address defined in the SPF record for the sending domain does not match the actual sender IP address of the host that forwards the message. The solution for this problem is to use the *Sender Rewriting Scheme* [84], which is a way to forward a message that rewrites the sender address so that the SPF check of the receiver accepts the message and the original sender address is still available for delivering bounce messages. From the sender's point of view, the problem with this approach is that email delivery to forwarded addresses depends on the fact that the forwarding MTA uses SRS, which is out of the sphere of influence of the sender.
Examples of domains that publish SPF records are `microsoft.com`, `hotmail.com`, `ibm.com`, `google.com`, `gmail.com`, `bankofamerica.com`, `dell.com`, and `sap.com`.

Example 2: Sender ID

Sender ID [85, 86] is a sender address authentication scheme proposed by Microsoft that is very similar to SPF. As in SPF, SPF records in the DNS are used to define which IP addresses of a domain are allowed senders of email. Sender ID additionally introduces a concept called *Purported Responsible Address* (PRA), which is "the identity of the party that appears to have most proximately caused that message to be delivered." [87]. The PRA is determined from the `From:`, `Sender:`, `Resent-From:`, and `Resent-Sender:`-header fields in the email, depending on which of these header fields exist in the email. In comparison with SPF, in Sender ID it is additionally possible to declare different valid IP addresses for different PRAs. The drawback of this additional functionality is that it is not sufficient to use data from the

message envelope (SMTP level), but also the content of the mail itself, more specifically the header of the email has to be analyzed.
Domains that publish SPF records including PRA declarations are, for example, `aol.com`, `ebay.com`, `paypal.com`, and `amazon.com`.

Example 3: DomainKeys Identified Mail (DKIM)

DomainKeys Identified Mail [88] (DKIM) is a sender authentication method that uses asymmetric cryptography. It is the combined successor of Yahoo!'s DomainKeys and Cisco's Identified Internet Mail (IIM) [89] sender authentication methods. The idea behind DKIM is that all emails from a domain are signed using a private key. This signature can be verified using a public key that is deposited in the DNS entry of this domain. The verification can be carried out by Mail Transfer Agents (MTAs) or Mail User Agents (MUAs) [90]. DKIM is broadly supported by IT organizations, notably AOL, Cisco, eBay/PayPal, IBM, PGP Corporation, and Yahoo!.

3.4.2 Writing Complaints

Writing complaints to an ISP that hosts a spammer's email server or website or to law enforcement authorities is a legal way to fight back against spammers. Many ISPs take complaints serious and shut down their services to spammers if enough complaints are received. To avoid complaints, spammers disguise the true origin of their emails by inserting forged header lines into their messages. Therefore, some effort and knowledge is required to determine the ISP from which a spam email is sent, which is a disadvantage of this method. Another problem is that some ISPs do not react to complaints, probably because they signed a *pink contract*, that is an agreement which allows the customer to send spam in exchange for a fee that is higher than usual [91]. ISPs that sell such contracts consequently ignore complaints. In this case, the only remaining alternative for the receiver of spam emails is to escalate the complaint to the ISPs upstreaming provider.

3.4.3 Active Response

Active response designates concerted visits of spammers websites by receivers of spam emails or volunteers. The idea is that most websites set up by spammers are designed to deal with only the tiny fraction of the spam emails' receivers that is really interested in buying the advertised products. If enough other receivers visit an website advertised by spam, this increases the spammers' bandwidth costs, probably enough to make spamming economically uninteresting. While certainly effective, the legality of this method is questionable since it is very similar to a DDoS-attack.

Example 1: Make LOVE not Spam

A notable example of this technique was the campaign "Make LOVE not Spam" launched by Lycos in 2004. Lycos then offered a screensaver that automatically visited selected spam websites to slow them down. Load balancing algorithms were used to ensure that the visited sites were not completely disabled. According to Starring, the Swedish firm that developed the campaign for Lycos, during one month 100.000 spam sites were irritated by over 110.000 screensavers [92].

Example 2: Blue Frog

Another prominent historical example of active response is Blue Frog. Blue Frog was an open source tool that automatically filed complaints on web sites advertised in spam emails. For each spam email received, exactly one complaint was created. Blue Security, the Israeli firm that developed that tool, provided a service that spammers could use to remove users of Blue Frog from their mailing lists. Blue Security finally had to stop their operation when a spammer launched a DDoS-attack against it [93,94]. At the moment, a successor project called *Okopipi*[2] is under development. Okopipi uses a distributed model to avoid being vulnerable to DoS-attacks. [96]

Problems of Active Response

While concerted actions against spammers have shown their effectiveness in the past, their major disadvantage is that they are illegal, at least as long as other entities than the official legal authorities carry them out. Despite best efforts of the initiators of such actions, it cannot be ruled out that innocent websites are hit, as happened for example in the "Make LOVE not spam"-campaign with `www.artofsense.com` [97]. Another problem is collateral damage that occurs to sites that are hosted by the same ISP as the target site advertised by spammers: When a spammer's website is visited by many clients simultaneously, not only the spammer's website suffers from adverse effects, such as a slow response time to the website, but also other websites from innocent customers hosted by the same ISP are affected by the same problems. This and several other problems with active response are discussed in more detail by Dittrich [98].

3.4.4 HTTP Tarpits

A preventive approach against spam is the use of *HTTP tarpits*. The idea is to keep spammers from collecting email addresses of web pages (*harvesting*) by intentionally slowing down the answers to HTTP requests from malicious web crawlers while sparing benign web crawlers and regular visitors of a web page. Further on, HTTP tarpits try to keep malicious crawlers

[2] *Okopipi* is the Tirio Indian name of the Blue Poison Dart Frog from South America [95].

as long as possible in the tarpit through dynamically generated hyperlink loops. Additionally, some HTTP tarpits poison the spammer's email address database by creating random email addresses that the email harvester collects in the process, which forces the spammer to remove all email addresses harvested from the web site with the HTTP tarpit if they do not want to accept a high percentage of bogus email addresses in their address lists [99].
Eggendorfer examined HTTP tarpits and found them as an effective preventive anti-spam measure [65].

3.4.5 Obfuscating Email Addresses on Web Pages

Obfuscating email addresses on web pages is a preventive measure against spam. It is targeted against spammers that acquire email addresses of potential receivers of their messages from the web by harvesting (see section 1.2.2). The easiest way to avoid the harvesting of an email address is to not publish it on the web, but this is not feasible for many email users. For example, web pages for businesses have to contain email addresses to allow potential or existing customers getting in contact with the business. Users, who have to publish their email addresses, can protect their addresses from being harvested by obfuscating it. Several different ways for obfuscating email addresses are presented in the following.

Simple Substitutions

A way to obfuscate email addresses that is easy to do, yet still effective are *simple substitutions*. In this approach, distinctive characters of an email address are substituted by character strings, so that the email address is still readable by humans, while it is not found by harvesters. Typically `@` is replaced with `(at)`, and `(dot)` is the substitute for periods in the email address. An example for an email address obfuscated this way is `gmb (at) seclab (dot) tuwien (dot) ac (dot) at`. The advantages of this method are its simplicity and effectiveness. The disadvantage of this method is, that it is impossible to use an email address obfuscated this way simply by clicking on it: the user has to undo the substitutions before using it in an email client.
A variant of the obfuscation of email addresses by simple substitutions is to substitute the characters of the email address by their corresponding *character references*. Character references are an alternative way to specify characters in HTML. Web browsers automatically replace character references by their corresponding characters, so that email addresses obfuscated this way can be used in the same way as plain text email addresses: neither clicking on the address nor copying and pasting it are blocked by this approach. The disadvantage of the use of character references is that it is a relative simple obfuscation, that is, it is very easy to adapt harvesters so that they can circumvent this obfuscation. But until many email addresses are obfuscated in this way, it can nevertheless protect addresses to a certain degree, because most harvesters do not bother to decode email addresses in character references when there are much more addresses available in plain text. An example for an email address

obfuscated by the substitution of its characters with the corresponding numeric references is gmb@seclab.tuwien.ac.at.

Graphical Email Addresses

Another way to obfuscate an email address on a web page is to use *graphical email addresses*. Instead of the email address in plain text, a graphic displaying the email address is published on the web. With this approach, the same goal is achieved as with simple substitutions: the email address is invisible for harvesters, but, nevertheless, readable for most human users. There are several disadvantages connected with this method: visually impaired persons that have to use special computer equipment to access the Internet are troubled by the use of graphics instead of plain text, since this special equipment is usually unable to deal with graphics. Another disadvantage is that users cannot use graphical email addresses simply by clicking on it or with copy and paste, but are burdened with the need to type in the email address. Additionally, graphical email addresses usually do not fit into the overall design of a web page, since the presentation of text on a web page can be different for different web browsers.

Encoding Email Addresses with JavaScript

Encoding email addresses with JavaScript is way to obfuscate email addresses that allows them to be clicked, copied and pasted, and used in the same way as if they were published in plaintext. To decode email addresses encoded by such an approach, the web page must be processed by a JavaScript-capable browser. The advantage of this approach is that sophisticated encodings are possible, that are not as easy to break as those created with simple substitutions, provided that the harvester does not use JavaScript. Since there are more than enough email addresses published in plaintext on the web, spam senders do not put effort in harvesting email addresses encoded with JavaScript, which is the reason why this obfuscation method works currently. The disadvantage is that this could change as soon as this obfuscation method is used widely. Besides this, another disadvantage is that users that cannot or do not want to use JavaScript (for example, users of handheld devices that are unable to process JavaScript, or users who do not want to use JavaScript for security reasons) are excluded from using email addresses obfuscated that way. An example for the implementation of obfuscating email addresses with JavaScript can be found in [100].

3.5 Comparison of Different Approaches against Spam

3.5.1 Legend

Table 3.1 is a comparison of the in Chapter 3 presented approaches against spam. The column with the heading *Operator* indicates, who operates a specific anti-spam approach. This can be either the sender, the receiver, both the sender and the receiver, or a separate entity. The next column with the heading *Dep. on user share* indicates, how the effectiveness of a specific approach depends on the total share of users that approach has. The dependency can be either positive (marked with '+'), that is, the more users take advantage of a specific approach, the more effective it works, or negative ('-'). If the effectiveness of an approach does not depend on its user share, the dependency is zero ('0'); if it grows exponentially with its user share, it is marked with '+ (expon.)'.
The other columns with the headings *Effort for the ...* indicate, how much effort a certain anti-spam approach costs different actors in the spam problem. There are individual columns for the efforts of the separate operator, if it exists (*Operator*), the receiver (*Rcvr.*), the average sender (*Avg. Sndr.*), the bulk email sender (*Bulk Sndr.*), the spam sender (*Spammer*), and innocent thirds (*Innoc. 3rds*).

3.5.2 Interpretation

It can be seen that there is no single best approach against spam. There are approaches against spam that are very burdensome for the spammer (Whitelisting, Email Postage, Computational Approach, Challenge/Response, Active Response), but each one of these approaches has its own disadvantage, be it the impossibility of strangers to contact the users of an approach (Whitelisting), the dependency of its effectiveness from the user share (Email Postage, Computational Approach), an enormous effort for one of the actors in the spam problem (Email Postage, C/R), or collateral damage, that is, effort for innocent thirds (C/R, Active Response).

Table 3.1: Comparison of different approaches against spam.

Approach	Operator	Dep. on	Effort for the					
		user share	Operator	Rcvr.	Avg. Sndr.	Bulk Sndr.	Spammer	Innoc. 3rds
Legal approach	separate	n/a	******	**	0	****	****	0
Opt-out	sender	0	n/a	******	0	***	****	0
Robinson lists	separate	0	******	**	0	**	**	0
Port 25 blocking	separate	+	**	0	*	0	*****	0
Bayesian filters	receiver	-	n/a	*****	**	**	******	0
Email addr. blacklist.	receiver	-	n/a	******	0	**	**	0
IP addr. blacklisting	separate	+	******	**	**	****	****	*
URI blacklisting	separate	+	******	**	0	****	****	0
Whitelisting	receiver	0	n/a	******	*	0	*******	0
Paid whitelists	separate	n/a	******	**	0	*****	*******	0
Greylisting	receiver	0	n/a	****	**	*****	******	0
Hash-b. w/o user inp.	separate	+	***	**	0	******	****	0
Hash-b. w. user inp.	separate	+	****	**	0	**	****	0
Heuristic filters	receiver	-	n/a	**	**	****	****	0
SMTP tarpits as filter	receiver	-	n/a	**	*	****	****	0
Spam-only SMTP tarpits	separate	-	**	n/a	0	0	*	0
Email postage	separate	+ (expon.)	*******	**	**	******	*******	0
Computational approach	sndr.&rcvr.	+ (expon.)	n/a	**	**	******	*****	0
Challenge/Response	sndr.&rcvr.	0	n/a	**	**	*******	*******	******
Sender address auth.	sender	n/a	n/a	**	**	**	**	0
Writing complaints	receiver	+	n/a	******	0	***	**	0
Active response	separate	+	******	**	0	****	******	******
HTTP tarpits	separate	+	**	****	0	0	******	0
Obfuscat. email addr.	receiver	-	n/a	**	**	0	******	0

4 Existing Attacks against Bayesian Spam Filters

The goal of attacks against spam filters is to get spam emails delivered in spite of the presence of filters. The currently existing attacks against Bayesian spam filters aim to achieve this by *adding* words to the spam emails. The objective is that these additional words are used in the classification of the email in the same way as the original words, thereby tampering with the classification process.

4.1 Random Word Attack

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 outweigh 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 [101,102] [12, pp. 135–137], 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. For this reason, they are unsuitable for compensating the original tokens' high spam probabilities. In contrast to that, Wittel and Wu have run a successful random word attack against a Bayesian spam filter [103]. They assume that the contradicting results could be "due to differences between training corpora or how the tested filters score messages." Another possible reason for the discrepancy could be "the minimal header information that was present in the emails they were using", as Graham-Cumming has mentioned [104].

4.2 Common Word Attack

An improvement of the random word attack is to add words to the spam email 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 random-chosen words for Bayesian filters, thus being better suited for an attack. This attack works, as several authors have shown, but the number of additional words that they needed to make a spam message unrecognizable for a Bayesian spam filter varies: Wittel and Wu found that approximately 50 additional common

words suffice to disguise a very short spam message (without headers) [103]. Lowd and Meek calculated that at least about 1000 words are needed to disguise a spam message that has an average spam probability (inclusive headers) [102]. These results do not necessarily contradict each other, since it is sound to assume that the higher the spam probability of the original spam message is, the more additional words are needed to disguise that message.

4.3 Frequency Ratio Attack

Lowd and Meek improved the common word attack by adding words that are common in the language but uncommon in spam. To determine these words, they used several representative English text corpora[1] and a spam corpus and "ranked each word by its frequency in each English corpora relative to its frequency in the spam corpus" [102]. They called their attack *frequency ratio attack*. Lowd and Meek calculated that about 150 words with a high frequency ratio suffice to make a spam message with an average spam probability unrecognizable for a Bayesian spam filter.

4.4 Random Text Attack

Another attack against Bayesian spam filters is the *random text attack* ("Bayesian poisoning"), in which the words to be added to the spam email are constructed from randomly chosen letters of the alphabet. There are several variants of this attack, two of which are presented in this section.

Zdziarski [12, pp. 137–139] described a random text attack that is carried out in two steps: first, a certain number of messages that carry words constructed from random letters in their headers are sent to the victims. This messages have to be learnt as being innocent by the victims' Bayesian filters. This can be achieved, for example, by disguising them as unsuspicious answers to messages on electronic mailing lists. The added words occur in innocent messages only, which causes them to receive a very low spam probability in the victims' Bayesian filters. This is exploited in the second step, where a spam message is sent to the victims that is modified by adding the same random-letter-words to the message-header as before. The random-letter-words are used in the classification of the message by the Bayesian spam filter in the same way as the other words of the message. Since the random-letter-words have a very low spam probability, the total spam probability of the message is lower than it would be without the additional random-letter-words. This increases the chances that the message is classified as legitimate.

Graham-Cumming [104] examined another variant of the random text attack: for each spam message to be sent, n spam messages, whose bodies consist of exactly one single random-

[1] A *corpus* is a collection of texts. It is used for example to calculate statistics on the language it represents, such as word frequencies.

letter-word, are sent too. In contrast to Zdziarski's version of the attack, all random-letter-words are different in this attack and are learnt by the attacked Bayesian spam filter as spammy tokens. This increases n_S, the total number of spam emails that have been used to train the Bayesian filter, thus reducing the spam probability of *all* tokens in the filter's token database. This can be seen by analyzing equation (2.2):

$$P(S|T) = \frac{\frac{n_S(T)}{n_S\uparrow}}{\frac{n_S(T)}{n_S\uparrow} + \frac{n_{\bar{S}}(T)}{n_{\bar{S}}}} \implies P(S|T)\downarrow \tag{4.1}$$

If n_S increases while all other variables stay constant, the numerator and the first summand of the denominator decrease, while the second summand of the denominator stays constant. For this reason, the spam probability of an arbitrary token $P(S|T)$ decreases. This lowers the combined spam probability of every message to be classified, which in turn makes it more likely that the Bayesian spam filter classifies messages as legitimate. Graham-Cumming has shown that a random text attack of this type works, but is inefficient: "this attack required 314,700 additional spams to get 101 out of 3,417 spam messages delivered and at the same time the attack improved the filter's ability to identify ham messages correctly".

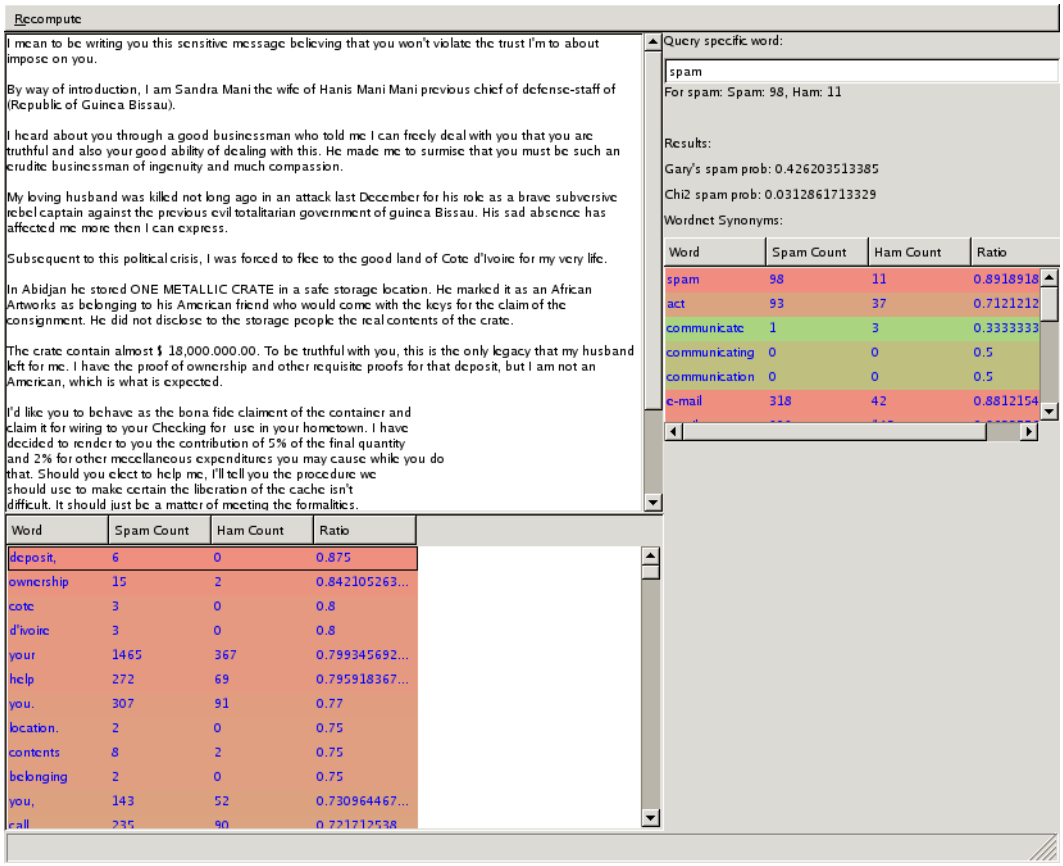
5 A New Approach: Word Substitution

All attacks against Bayesian filters described in Section 4 have one thing in common: they add certain words to spam emails. 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. Widespread use of algorithms that are able to detect these additional words, such as Zdziarski's Bayesian Noise Reduction algorithm [105], would foil attacks that add words to messages. However, there is another possibility for disguising spam emails: instead of adding known good words to compensate for the bad words in the spam email, one could exploit redundancies in the language and substitute words with a high spam probability for synonyms with a lower spam probability, as first hinted at by Bowers [106].
Bowers wrote a GUI which assisted him in substituting manually the high-spam-probability-words of a spam message. This successfully lowered the message's spam probability.
A word substitution attack requires that words in spam emails have similar spam probabilities in all the receivers' Bayesian spam filters. In this thesis, first, spam emails from three different spam archives are examined to determine whether this precondition is met. After that, words with a high spam probability are substituted automatically in a test set of 100 spam messages. Last, the effectiveness of the attack is measured by comparing the filter efficiency of three different spam filters on the modified spam messages and on the original versions.

5.1 Preconditions

For a successful substitution attack, it is necessary that Bayesian spam filters 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 by *(a)* the number of appearances of this word in spam emails, *(b)* the number of appearances of this word in ham emails, *(c)* the total number of spam emails, and *(d)* the total number of ham emails the Bayesian filter has classified. If the spam emails and the ham emails 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 emails used for training Bayesian spam filters of different users the same? This is

Figure 5.1: Bowers' GUI for substituting words in spam emails.



clearly not the case. But many spam filters are set up for more than one user, and thus, use broader samples of spam and ham email for training. Therefore, it cannot be ruled out that the precondition is satisfied.

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, thus foiling the attack. For ham messages, this topic has not been researched thoroughly, probably because it is difficult to obtain large message corpora to analyze for privacy reasons. The volatility of spam emails is a better researched area: Sullivan [107] 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: "because spam features seem to mutate gradually, those features will, generally speaking, exhibit extraordinary

longevity (measurable in months, not minutes)." Pu and Webb [108] 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 some spam construction techniques become extinct over time while others are able to exist despite filtering. Changes in spam construction techniques occur over several months, which confirms Sullivan's claim.

5.1.1 Spam Volatility of Different Spam Archives

In order to determine whether spam messages change slowly enough to allow a substitution attack, three different spam archives were examined. First, messages received in the year 2005 were extracted from the archives and divided by the month they were received. Next, lists of the most frequently used tokens were created for each month. Then, the *overlap* of these lists was measured 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.
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, it was also measured how many of the 100 most frequently used tokens of one month appear among the 200 most frequently used tokens of another month.

Results

The results 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. If tokens that are not completely missing in the other month, but are only a little bit less frequent (that is, among the 200 most frequently used tokens) are counted too, the overlap is even higher. Many of the terms used in spam messages do not change over the course of a year, which indicates that the spam probabilities Bayesian filters assign to these terms do not change too much either. This could suffice for substitution attacks to work successfully.

5.2 Effectiveness of Substitution Attacks

The effectiveness of a substitution attack is the degree to which it is able to fulfill its goal: to get spam emails delivered in spite of the presence of filters. To estimate it, first, a test set

of spam messages was created in which tokens with a high spam probability were replaced with tokens with a lower spam probability. These messages were then sent to different spam filters. The filters' false negative percentages (FNPs), that is, the proportion of spam messages that were erroneously classified as ham, were measured; the effectiveness of the attack on the different filters was calculated from the variation of the FNPs.

5.2.1 Creating a Test Set of Spam Messages

To estimate the effectiveness of a substitution attack in penetrating spam filters, a test set of spam messages was created: 100 unique spam messages that were received between January and May 2006 and with a body written in English were randomly selected from *Bruce Guenter's SPAM Archive* [109] (Test Set A). All header lines, except for those starting with `Subject:`, `MIME-Version:`, and `Content-Type:`, were removed from the test emails for two reasons: the header lines, with exception of the subject-line, are not altered in a substitution attack and, therefore, have no influence on the effectiveness of the attack. On the other hand, it is possible that tokens from the header-lines of our test messages are already learnt by the test-filters, which would bias them in favor of marking the test messages as spam (Test Set B).
Spam messages that use HTML were then converted into plain text. To change the emails as little as possible with regard to RHSBLs and Bayesian filters, HTML hyperlinks in the messages, that is, URIs in `<a>`-tags were copied into the plain text, other markup was deleted entirely. The resulting spam messages had still one characteristic that could distort an effectiveness measurement and hinder the automatic substitution of suspicious words, namely the alterations performed by tokenizer attacks. Tokenizer attacks use substitutions of characters by similar looking numbers, insertions of random characters, and many other tricks such as those mentioned on Graham-Cumming's site *The Spammers' Compendium* [112] to disguise words from filters. The changes of these attacks were removed manually (Test Set C).
Finally, tokens with a high spam probability were automatically replaced by synonyms of these tokens with lower spam probability. To find words with a semantics similar to the word that should be replaced, Grady Ward's *Moby Thesaurus* [113] was consulted (Test Set D).

5.2.2 Automatic Substitution of Spam Tokens

The process to automatically substitute words with a high spam probability first splits the email into single words. Then, the spam probability is determined for each of these words. For each word with a spam probability higher than that of a newly learnt token, a synonym with a lower spam probability is looked up in a thesaurus. This new word then replaces the original, suspicious word.
To determine which synonym should be used for a substitution, first, the spam probability of all synonyms are calculated. In preliminary experiments, simply the synonym with the lowest spam probability was selected for the substitution. The disadvantage of that approach was

that many substitutions found that way distorted the text's semantics. That is, they did not fit into the context of the word to be substituted. To mitigate this problem, the substitution algorithm was extended so that the context was also taken into account: for each synonym, the *Google count*[1] of the phrase of the synonym in the context[2] of the word to be substituted was determined and compared with the Google count of the original phrase. If the Google count of the synonym-phrase was lower than that of the original phrase, the synonym was ranked down in the list of possible synonyms: if there were two synonyms with equal spam probability, the synonym whose synonym-phrase had the higher Google count was preferred. Synonyms that did not fit into a certain context were not used for substitution, as long as there were other, better-fitting synonyms.
There are several ways to rank words according to their spam probability: The ratio between the frequency of a word in an English corpus and in a spam corpus could be used as measure for the spam probability of this word, as Lowd and Meek did for their frequency ratio attack (see Section 4). For this thesis, the spam probability was calculated directly using a Bayesian spam filter: DSPAM 3.4.9 was trained with 77,514 spam messages from Bruce Guenter's *SPAM Archive* [109] (that were received between January and August 2005), as well as with 78,189 ham messages from the Enron email dataset [114]. Since the Enron corpus contains not only ham email, but also some spam messages, it was necessary to use only these messages that are definitely ham emails. This was done by using only emails in the sent-directories of the corpus. Viruses or worms that automatically send email could have left traces in the sent-directories and, therefore, could have interfered with our measurements. For this reason, *F-Prot Antivirus for Linux* from FRISK Software International [115] was used to ensure that the ham messages that were used to train the Bayesian filter did not contain viruses or worms.

5.2.3 Example

The creation and the content of the four test sets is illustrated here by an example. Further examples, that point out the changes made by the replacement of spammy words with synonyms can be found in Appendix B.

A: Original Message

```
Return-Path: <AlineFraser@expedition-nepal.com>
```

[1] The *Google count* is Google's estimation of the number of web pages that contain a certain search term or phrase.

[2] As context the immediately preceding and succeeding words of the word to be substituted were used, as long as they were in the same sentence as that word. The exact number of words from the context that were used for the construction of the synonym-phrases depended on the number of synonyms that existed for the word to be substituted, and on the Google counts of the phrases built from that synonyms: the more synonyms exists that have synonym-phrases with a Google count that is greater than or equal to the Google count of the original phrase, the more context words were used to build the synonym phrases (and the original phrase for comparison). This ensured that for more frequent phrases, a bigger context was used in the synonym determination.

```
Delivered-To: em-ca-bruceg@em.ca
Received: (qmail 31517 invoked from network); 26 Mar 2006 11:59:21 -0000
Received: from host-81-190-252-22.elk.mm.pl (81.190.252.22)
  by churchill.factcomp.com with SMTP; 26 Mar 2006 11:59:21 -0000
Received: from snider.rivulet.nl ([59.48])
by edify.usurpation.org with smtp (Exim 2.95 #7 (Debian))
id 19IBYn-0000n4-00
for <tore-inference-passerby@depth.simplectic.debian.org>; Sun, 26 Mar 2006 17:00:01 +0500
Message-ID: <150.83@161.dyn.indifferent.edu>
From: "Michael Riggs" <AlineFraser@expedition-nepal.com>
Date:
To: bruceg@em.ca
Subject: unbeatable progressive jack-pots
X-Mailer: Ximian 7.5
MIME-Version: 1.0
Content-Type: text/html; charset="us-ascii"
Content-Transfer-Encoding: 7bit

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD>
<META http-equiv=Content-Type content="text/html; charset=iso-8859-1">
</HEAD>
<BODY><BR>Welcome to Green Table where you are the star!<BR><BR>
Open a real money account and claim your $300 welcome bonus.<BR><BR>
Green Table will reward your 1st deposit with a 100% match bonus up to $100.<BR><BR>
On  top of this your 2nd deposit will be rewarded with a generous 50% bonus up to $200
<BR><BR>
Enjoy 70+ games, exclusive tournaments and amazing promotions in a 100% secure online
casino.<BR><BR>
See you soon on our tables.<BR>
<A href="http://casinorich.info">casinorich.info</A><BR><BR>
Green Table Director<BR><BR><BR>
<A href="http://casinorich.info/opt-out.php">Opt-out link</A>
</BODY>
</HTML>
```

B: Header Lines Removed

```
Subject: unbeatable progressive jack-pots
MIME-Version: 1.0
Content-Type: text/html; charset="us-ascii"
Content-Transfer-Encoding: 7bit

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD>
<META http-equiv=Content-Type content="text/html; charset=iso-8859-1">
</HEAD>
<BODY><BR>Welcome to Green Table where you are the star!<BR><BR>
Open a real money account and claim your $300 welcome bonus.<BR><BR>
Green Table will reward your 1st deposit with a 100% match bonus up to $100.<BR><BR>
On  top of this your 2nd deposit will be rewarded with a generous 50% bonus up to $200
<BR><BR>
Enjoy 70+ games, exclusive tournaments and amazing promotions in a 100% secure online
casino.<BR><BR>
See you soon on our tables.<BR>
```

```
<A href="http://casinorich.info">casinorich.info</A><BR><BR>
Green Table Director<BR><BR><BR>
<A href="http://casinorich.info/opt-out.php">Opt-out link</A>
</BODY>
</HTML>
```

C: Plaintext, Tokenizer Attacks Removed

```
Subject: unbeatable progressive jackpots

Welcome to Green Table where you are the star!

Open a real money account and claim your $300 welcome bonus.

Green Table will reward your 1st deposit with a 100% match bonus up to $100.

On top of this your 2nd deposit will be rewarded with a generous 50% bonus up to $200

Enjoy 70+ games, exclusive tournaments and amazing promotions in a 100% secure online
casino.

See you soon on our tables.
casinorich.info: http://casinorich.info/

Green Table Director


Opt-out link: http://casinorich.info/opt-out.php
```

D: Spammy Words Substituted

```
Subject: unique in jackpots

Cheer to Original Table where you are the star!

Open a good property list and have your $300 welcome overage.

Original Table will honor your 1st place with a 100% match overage up to $100.

On all of this your 2nd post will be rewarded with a generous 50% plus up to $200

Love 70+ tourney, any tournaments and wonderful promotions in a 100% sign online
casino.

See you soon on our tables.
casinorich.info: http://casinorich.info/

Original Table Director


Opt-out date: http://casinorich.info/opt-out.php
```

5.2.4 Measuring the Effectiveness of the Attack for Different Filters

Three different filters were used to classify the messages from our test set: SpamAssassin 3.0.6 [62], the spam filter of Gmail [116], and the spam filter of Yahoo! Mail [117].
SpamAssassin was tested in two configurations: the default configuration, which includes network tests[3], and an alternative configuration, where the network tests were disabled. SpamAssassin was trained with 39,648 ham messages and 58,004 spam messages from one supervisor's personal email correspondence. Autolearning and autowhitelisting was disabled during the tests (by inserting the lines `bayes_auto_learn 0` and `auto_whitelist_factor 0` into `~/.spamassassin/user_prefs`) to avoid distorting the subsequent tests.
The tests of the spam filters of Gmail and Yahoo! Mail are black box tests, because the message classification algorithms used by these filters are not published. Under the assumption that these spam filters include a learning component that adapts to the emails different users receive, classifying one test set would influence the classification of other test sets. For this reason, all tests were carried out on newly created Gmail- respective Yahoo! Mail-accounts.
DSPAM 3.4.9, which was used for determining the spam probability of the single words of the test messages, was also included in our tests. Note that this reflects the ideal case from the attacker's point of view, since the spam probability of the individual words is perfectly known here. For this reason, the results for this filter should be considered as an upper bound on the effectiveness of the attack. The features "chained tokens", "training loop buffering", and "automatic whitelisting" were disabled for our tests (by commenting out the particular lines in `/usr/local/etc/dspam.conf`) for the same reason as mentioned above.

5.2.5 Interpretation of the Results

The results of the experiment are shown in Table 5.5. The difference between the FNPs for Test Set D (spam words substituted) and Test Set C (plaintext, no attacks) is a measure for the *effectiveness* of a substitution attack. More precisely, the effectiveness of the attack is the ratio between the actual difference of the FNPs and the maximal possible difference (100 minus the FNP for Test Set C).
The FNPs of DSPAM 3.4.9 provide an upper bound on the other filters' FNPs, as this filter was used for determining the spam probabilities of words in the test messages. DSPAM's FNP for Test Set D is 94, which means that 6 messages of the test set were detected as spam by DSPAM in spite of the changes made by the substitution of the spam words. This 6 messages have some common characteristics: they are very short and contain many tokens that are not substitutable, such as brand names, URIs, or email addresses.

[3] SpamAssassin's *network tests* are these tests that cannot be executed solely by the computer that runs SpamAssassin, but require access to the Internet. Examples are: *(a)* querying DNS blacklists (that is, lists of spam-sending IP addresses and URIs advertised in spam messages), *(b)* using hash-based methods (such as Vipul's Razor, Pyzor, and DCC), and *(c)* verifying the sender domain's SPF record.

The other numbers demonstrate that a substitution attack can work successfully against real-life spam filters. SpamAssassin 3.0.6 (local tests only) has the highest FNPs in this experiment: 65 of the 100 original test spam messages were not detected by this filter. This filter in the very same configuration is used for filtering one of the supervisors' email correspondence, where it does not show abnormal high FNPs. Possibly the reason for this deviation is that the Bayesian component of SpamAssassin is highly trained on this supervisor's personal messages and, therefore, is relatively inaccurate when used on other messages.
Comparing the FNPs of Test Set A (original spams) and Test Set B (test spams without header lines) shows that without being able to analyze the header lines, the spam filters FNPs rise. Since the header often contains indications for spam messages, this result was expected.
Of particular interest is the comparison between the FNPs of Test Set B (test spams without header lines) and Test Set C (plaintext, no attacks). It can be seen that converting the HTML-emails into plain text and removing the tokenizer attacks (such as substitutions of characters by similar looking numbers) increases the FNPs of the filters. A possible explanation for this could be that rule-based spam filters such as SpamAssassin are able to detect HTML-based tricks and character substitutions. For Bayesian spam filters, the increase could be explained by the fact that character substitutions only work when the spam filter has never seen this variation of the altered word before, otherwise, the modified word has a higher spam probability than the unmodified word. Zdziarski [12, p.124] describes this phenomenon as follows: "One of the biggest mistakes spammers can make is using approaches that generate unique identifiers of spam." Apparently, some tricks spammers use do more harm to themselves than good in the long run, which causes them to become extinct. This phenomenon has been examined in more detail by Pu and Webb [108].

5.3 Problems

There are two problems with substitution attacks: the bad readability of the disguised messages, and the low effectiveness in disguising spam emails for spam filters.
The readability of a message suffers from a substitution attack for the following reasons: To find suitable synonyms for words with a high spam probability, it is not always sufficient to look up the word in a thesaurus, because many words are ambiguous, that is, there is more than one meaning for a word. Thesauruses, such as Grady Ward's *Moby thesaurus* [113] (which was used in the experiments described in this thesis), typically contain synonyms for all different meanings of the terms they contain. Many substitutions that were made automatically in the experiments described above are distorting, that is, words were substituted by synonyms that have another meaning than the original words in the particular contexts. Taking into account the context of the word to be substituted (see Section 5.2.2) mitigated this problem somewhat, but could not avoid distorting substitutions entirely. Examples for such substitutions from the experiments are: `Adobe`/`Stone`, `Office`/`Room`, `Special offer`/`Differential issue`, and `Click here`/`Pad tonight`.

The reason for the low effectiveness of the attack is that the high spam probability of a message is caused in general not only by tokens that can be replaced by synonyms, but also by other tokens: URIs and email addresses are in principle not substitutable; other words, especially proper nouns (such as brand names) and abbreviations either do not have synonyms at all or they do not have synonyms with a low spam probability. In our experiments, prominent examples for tokens with a high spam probability that could not be replaced by a synonym with a lower spam probability are: `http`, `php`, `Viagra`, `Norton`, `XP`, `Hoodia`, `biometric`, `nanotechnology`, and `lottery`.

5.4 Possible Solutions

The attack would possibly be more effective if not only single words were substituted by synonyms, but if entire passages were rewritten entirely to avoid words with a high spam probability. Since it is infeasible to do this automatically, it would have to be done manually. It is questionable whether the additional effort necessary is justified by the possible gain in effectiveness of the attack.
Some distortions occur when the part of speech[4] of a word changes because of the substitution. This distortions could be avoided automatically, namely by using part-of-speech tagging algorithms to determine the part of speech of the original word in the spam message and substituting it considering this additional information.

[4]The *part of speech*, also known as word class, designates the role of a word in a sentence. The most important parts of speech are: noun, verb, adjective, adverb, pronoun, preposition, conjunction, and interjection.

Table 5.1: Overlap of the most frequently used tokens in Bruce Guenter's *SPAM Archive* [109] (in percent). Note that the calculation of the overlap is meaningful only for different months.

	in top-100 of											
	Ja	F	Mr	Ap	My	Je	Jl	Ag	S	O	N	D
top-100 of												
January	—	78	74	75	74	77	68	72	68	74	75	73
February	78	—	90	86	84	85	84	85	79	84	84	84
March	74	90	—	89	84	85	85	86	80	84	86	84
April	75	86	89	—	90	85	82	82	75	78	83	80
May	74	84	84	90	—	80	76	77	72	76	78	78
June	77	85	85	85	80	—	82	84	76	84	86	83
July	68	84	85	82	76	82	—	91	82	81	84	86
August	72	85	86	82	77	84	91	—	83	83	86	85
September	68	79	80	75	72	76	82	83	—	79	80	79
October	74	84	84	78	76	84	81	83	79	—	88	82
November	75	84	86	83	78	86	84	86	80	88	—	89
December	73	84	84	80	78	83	86	85	79	82	89	—
	in top-200 of											
	Ja	F	Mr	Ap	My	Je	Jl	Ag	S	O	N	D
top-100 of												
January	—	95	96	87	91	92	87	87	90	91	90	87
February	87	—	100	99	98	99	99	98	98	98	97	94
March	82	100	—	99	97	100	99	99	100	98	98	96
April	85	96	98	—	100	99	96	97	97	95	95	93
May	82	94	96	100	—	98	94	95	95	93	92	90
June	86	99	97	96	99	—	100	99	98	98	98	99
July	76	93	93	92	87	94	—	100	95	93	97	96
August	80	94	94	93	89	95	100	—	97	94	95	97
September	76	86	86	86	82	86	86	86	—	86	85	85
October	84	99	98	96	94	98	99	97	100	—	99	97
November	84	97	97	94	93	98	99	97	100	100	—	100
December	80	94	97	94	91	96	99	98	96	99	100	—

Table 5.2: Overlap of the most frequently used tokens in *SpamArchive.org* [110] (in percent).

	in top-100 of											
	Ja	F	Mr	Ap	My	Je	Jl	Ag	S	O	N	D
top-100 of												
January	—	77	71	65	73	76	77	79	75	74	73	74
February	77	—	77	74	81	82	80	81	78	79	78	77
March	71	77	—	68	82	83	82	83	83	83	81	82
April	65	74	68	—	76	72	67	69	67	65	65	66
May	73	81	82	76	—	90	83	82	81	80	81	81
June	76	82	83	72	90	—	88	86	84	82	82	83
July	77	80	82	67	83	88	—	87	85	89	89	89
August	79	81	83	69	82	86	87	—	89	85	83	84
September	75	78	83	67	81	84	85	89	—	87	84	85
October	74	79	83	65	80	82	89	85	87	—	95	95
November	73	78	81	65	81	82	89	83	84	95	—	95
December	74	77	82	66	81	83	89	84	85	95	95	—
	in top-200 of											
	Ja	F	Mr	Ap	My	Je	Jl	Ag	S	O	N	D
top-100 of												
January	—	91	96	89	86	94	97	100	95	95	91	92
February	86	—	96	97	97	95	95	94	96	96	97	97
March	81	90	—	91	93	94	94	93	93	94	95	95
April	78	87	88	—	96	90	82	87	87	85	82	87
May	84	91	98	99	—	99	96	96	98	97	97	98
June	88	94	100	96	99	—	100	99	100	100	99	99
July	91	96	96	94	97	95	—	100	98	99	100	100
August	90	95	99	95	95	98	100	—	100	100	100	100
September	86	91	99	93	95	99	99	99	—	99	100	100
October	89	95	94	92	96	93	99	97	97	—	100	100
November	85	93	93	89	94	92	99	96	97	100	—	100
December	87	92	93	91	94	92	98	96	96	99	100	—

Table 5.3: Overlap of the most frequently used tokens in *TLIQuest Spam Archives* [111] (in percent).

	in top-100 of											
	Ja	F	Mr	Ap	My	Je	Jl	Ag	S	O	N	D
top-100 of												
January	—	77	79	70	76	81	75	79	78	75	79	75
February	77	—	78	63	80	82	81	80	77	79	78	74
March	79	78	—	74	78	81	77	80	79	78	78	75
April	70	63	74	—	72	69	65	68	70	68	68	63
May	76	80	78	72	—	87	87	84	81	78	77	79
June	81	82	81	69	87	—	89	87	83	84	82	84
July	75	81	77	65	87	89	—	89	87	84	82	86
August	79	80	80	68	84	87	89	—	93	85	83	83
September	78	77	79	70	81	83	87	93	—	85	85	83
October	75	79	78	68	78	84	84	85	85	—	88	83
November	79	78	78	68	77	82	82	83	85	88	—	87
December	75	74	75	63	79	84	86	83	83	83	87	—
	in top-200 of											
	Ja	F	Mr	Ap	My	Je	Jl	Ag	S	O	N	D
top-100 of												
January	—	93	97	94	89	95	93	95	96	93	93	93
February	88	—	93	90	93	96	97	94	92	96	92	92
March	87	98	—	95	93	93	96	94	92	93	93	92
April	80	83	92	—	96	91	85	90	86	84	82	83
May	83	94	92	98	—	99	99	99	95	93	92	95
June	91	99	94	94	99	—	100	99	99	100	99	99
July	84	96	89	91	99	100	—	99	98	98	97	100
August	88	96	93	94	98	99	100	—	100	99	98	100
September	87	95	93	95	97	100	100	100	—	99	98	100
October	90	96	88	89	95	98	98	97	98	—	100	98
November	87	97	88	90	92	96	98	96	97	100	—	100
December	84	93	83	85	94	93	98	95	96	97	98	—

Table 5.4: Overlap of the most frequently used tokens in three different spam archives for 2005.

Parameter	Data set		
	Bruce Guenter's SPAM Archive [109]	SpamArchive.org [110]	TLIQuest Spam Archives [111]
Emails examined	127,120	334,477	52,799
Overlap 100/100			
min. (%)	68	65	63
max. (%)	91	95	93
avg. (%)	81.2	80.0	79.2
Overlap 100/200			
min. (%)	76	78	80
max. (%)	100	100	100
avg. (%)	94.1	94.6	94.2

Table 5.5: Effectiveness of a substitution attack, measured by the false negative percentages (FNPs) of different spam filters

Filter	FNP for Test Set				Effectiveness of Attack	
	A	B	C	D	abs.	% of max. poss.
SpamAssassin 3.0.6	30	46	51	56	5	10.2
SpamAssassin 3.0.6 (local tests only)	65	70	78	89	11	50.0
Gmail	–[a]	24	29	33	4	5.6
Yahoo! Mail	–[a]	37	43	57	14	24.6
Upper bound:						
DSPAM 3.4.9 (used in attack)	0	14	22	94	72	92.3

Note: The following test sets were used: (A) original spam emails; (B) spam emails without header lines; (C) spam emails without header lines, converted into plaintext, and tokenizer attacks removed; (D) same as (C), but words with a high spam probability substituted by words with a lower spam probability. N = 100.

[a] Since sending the test spam messages per email alters the header lines, it is impossible to measure the filter efficiency for the original messages without local access to the filter.

6 Summary and Conclusion

Nowadays, it is common to disguise spam messages by making them as short as possible or entirely unreadable for spam filters, but there is another way to penetrate Bayesian filters: one could modify spam messages by replacing words with a high spam probability by synonyms of these words with a lower spam probability. Although there are problems with substitution attacks, such as the low effectiveness in penetrating spam filters and the bad readability of spam messages modified by them, substitution attacks could be interesting for spammers because of one characteristic of the resulting spam messages: they do not contain additional blocks of text, which is a detectable feature of spam messages modified by other attacks.
This thesis has shown a new way to circumvent Bayesian spam filters. Spam filters in general will never work perfectly, but maybe there are other aspects of the spam problem that are more important than this: although effective spam filters are available for a few years now, there are still many email users that neither use filters nor other measures to attack the spam problem. As long as there are such users, the spam business will remain lucrative for the spam senders. Probably not a single approach, but a combination of several different anti-spam methods, including the education of email users, will be able to reduce the amount of spam significantly.

Bibliography

[1] MessageLabs Intelligence: September 2007. http://www.messagelabs.com/mlireport/MLI_Report_September_Q3_2007.pdf.

[2] The Radicati Group. Free Industry Stats. http://www.radicati.com/news/facts.asp, May 2007.

[3] Postini corporate spam filtering statistics, including emails blocked, spam data and more. http://www.postini.com/stats/, October 2007.

[4] Jack Hitt. *Confessions Of A Spam King*. The New York Times, September 28, 2003. Available at http://query.nytimes.com/gst/fullpage.html?res=9C00E0D71E3AF93BA1575AC0A9659C8B63.

[5] John Leyden. *We'll buy smut if you send us the spam*. The Register, 26th April 2006. Available at http://www.theregister.co.uk/2006/04/26/spam_response_survey/.

[6] The Quotations Page. http://www.quotationspage.com/quote/1992.html.

[7] SPAM.com. http://www.spam.com/.

[8] Wikipedia. Internet bot — Wikipedia, The Free Encyclopedia. http://en.wikipedia.org/w/index.php?title=Internet_bot&oldid=163560455, 2007. [Online; accessed 11-October-2007].

[9] The CAPTCHA Project. http://www.captcha.net/.

[10] John Graham-Cumming. Anti-Spam Tool League Table. http://www.jgc.org/astlt.html.

[11] T. Berners-Lee, R. Fielding, and L. Masinter. Uniform Resource Identifier (URI): Generic Syntax. RFC 3986 (Standard), January 2005.

[12] Jonathan A. Zdziarski. *Ending Spam: Bayesian Content Filtering and the Art of Statistical Language Classification*. No Starch Press, San Francisco, Calif., 2005.

[13] Paul Graham. A plan for spam. http://www.paulgraham.com/spam.html, August 2002.

[14] Tom M. Mitchell. *Machine learning*. WCB/McGraw-Hill, New York, N.Y., 1997.

[15] Christopher D. Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, Mass., 1999.

[16] Tim Peters. [python-dev] re: [python-checkins] python/nondist/sandbox/spambayes gbayes.py,1.7,1.8. http://mail.python.org/pipermail/python-dev/2002-August/028216.html, 22 August 2002.

[17] Gary Robinson. A statistical approach to the spam problem. *Linux Journal*, 2003(107):3, March 2003. Also available at http://www.linuxjournal.com/article/6467.

[18] M. E. Maron. Automatic indexing: An experimental inquiry. *Journal of the ACM*, 8(3):404–417, 1961.

[19] Tom Van Vleck. The history of electronic mail. http://www.multicians.org/thvv/mail-history.html, September 2004.

[20] Wikipedia. ARPANET — Wikipedia, The Free Encyclopedia. http://en.wikipedia.org/wiki/ARPANET, 2007. [Online; accessed 18-February-2007].

[21] Ray Tomlinson. The first email. http://openmap.bbn.com/~tomlinso/ray/firstemailframe.html.

[22] Jonathan B. Postel. On the junk mail problem. RFC 706, November 1975.

[23] Brad Templeton. Reaction to the DEC spam of 1978. http://www.templetons.com/brad/spamreact.html.

[24] Jonathan B. Postel. Simple mail transfer protocol. RFC 788, November 1981.

[25] Paul Mockapetris. Domain names – concepts and facilities. RFC 882, November 1983.

[26] Paul Mockapetris. Domain names – implementation and specification. RFC 883, November 1983.

[27] Jason D. M. Rennie. ifile: An application of machine learning to e-mail filtering. In *Proceedings of the KDD-2000 Workshop on Text Mining*, pages 59–64, 2000. Also available at http://people.csail.mit.edu/~jrennie/papers/ifile00.pdf.

[28] Jason D. M. Rennie. The ifile web site. http://people.csail.mit.edu/jrennie/ifile/, November 2005.

[29] Mehran Sahami, Susan Dumais, David Heckerman, and Eric Horvitz. A bayesian approach to filtering junk e-mail. In *Learning for Text Categorization: Papers from the AAAI Workshop. AAAI Technical Report WS-98-05*, pages 55–62. American Association for Artificial Intelligence, Menlo Park, California, July 1998.

[30] Discussion on Slashdot about Graham's essay"A Plan for Spam". http://developers.slashdot.org/article.pl?sid=02/08/16/1428238&mode=thread&tid=156.

[31] SpamProbe – A fast Bayesian spam filter. http://spamprobe.sourceforge.net/.

[32] SpamBayes. http://spambayes.sourceforge.net/.

[33] Bogofilter home page. http://bogofilter.sourceforge.net/.

[34] POPFile – automatic email classification. http://popfile.sourceforge.net/.

[35] The DSPAM project. http://dspam.nuclearelephant.com/.

[36] Controlling the Assault of Non-Solicited Pornography and Marketing Act of 2003 ("CAN-SPAM Act of 2003"). 15 U.S.C. § 7701 et seq., Pub. L. No. 108-187, S. 877. Also available at http://uscode.house.gov/download/pls/15C103.txt and http://thomas.loc.gov/cgi-bin/query/z?c108:S.877:.

[37] MX Logic wraps up 2006 with year-end threat summary, 2007 predictions. http://www.mxlogic.com/news_events/press_releases/release.cfm?id=140&year=2007, 8 January 2007.

[38] Directive 2002/58/EC of the European Parliament and of the Council of 12 July 2002 concerning the processing of personal data and the protection of privacy in the electronic communications sector (Directive on privacy and electronic communications). http://europa.eu.int/eur-lex/pri/en/oj/dat/2002/l_201/l_20120020731en00370047.pdf.

[39] David E. Sorkin. Spam Laws. http://www.spamlaws.com/.

[40] Nicola Lugaresi. European Union vs. spam: A legal response. In *Proceedings of the First Conference on Email and Anti-Spam (CEAS)*, July 2004. Also available at http://www.ceas.cc/papers-2004/145.pdf.

[41] Evangelos Moustakas, Chandrasekaran Ranganathan, and Penny Duquenoy. Combating spam through legislation: A comparative analysis of US and european approaches. In *Proceedings of the Second Conference on Email and Anti-Spam (CEAS 2005)*, 2005. Also available at http://www.ceas.cc/papers-2005/146.pdf.

[42] The Spamhaus Block List. http://www.spamhaus.org/sbl/.

[43] Spam Prevention Early Warning System. http://www.spews.org/.

[44] Spam and Open Relay Blocking System. http://www.sorbs.net/.

[45] Not Just Another Bogus List. http://www.njabl.org/.

[46] SpamCop Blocking List. http://spamcop.net/bl.shtml.

[47] Spam URI Realtime Blocklists. http://www.surbl.org/.

[48] Realtime URI Blacklist. http://www.uribl.com/.

[49] The Abusive Hosts Blocking List. http://www.ahbl.org/.

[50] RFC-Ignorant.org. http://www.rfc-ignorant.org/.

[51] Goodmail Systems. http://www.goodmailsystems.com/.

[52] Sender Score. http://www.senderscore.com/.

[53] John C. Klensin. Simple mail transfer protocol. RFC 2821 (Proposed Standard), April 2001.

[54] Distributed Checksum Clearinghouse. http://www.rhyolite.com/anti-spam/dcc/.

[55] David Goldberg, David Nichols, Brian M. Oki, and Douglas B. Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70, December 1992.

[56] Vipul's Razor. http://razor.sourceforge.net/.

[57] Cloudmark, Inc. http://www.cloudmark.com/.

[58] Pyzor. http://pyzor.sourceforge.net/.

[59] Aleksander Kołcz, Abdur Chowdhury, and Joshua Alspector. The impact of feature selection on signature-driven spam detection. In *Proceedings of the First Conference on Email and Anti-Spam (CEAS)*, July 2004. Also available at http://www.ceas.cc/papers-2004/147.pdf.

[60] Ruslan Ibragimov. Send-Safe Mailer. http://www.send-safe.com/send-safe.html.

[61] John Graham-Cumming. The bulk email tool Send-Safe. jgc's spam and anti-spam newsletter, 30 November 2004. Also available at http://www.jgc.org/antispam/11302004-1053f2f83813fef2b02e82f38a1bee87.pdf.

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

[63] Tom Liston. LaBrea-intro history. http://labrea.sourceforge.net/Intro-History.html.

[64] Kang Li, Calton Pu, and Mustaque Ahamad. Resisting spam delivery by TCP damping. In *Proceedings of the First Conference on Email and Anti-Spam (CEAS)*, July 2004. Available at http://www.ceas.cc/papers-2004/191.pdf.

[65] Tobias Eggendorfer. Comparing SMTP and HTTP tar pits in their efficiency as an anti-spam-measure. In *Proceedings of the MIT Spam Conference 2006*, March 2006. Available at http://www.spamconference.org/SC2006.iso.

[66] Martin A. Lamb. TarProxy: a statistically-driven SMTP tarpit. http://www.martiansoftware.com/tarproxy/.

[67] Lutz Donnerhacke. Teergrubing wrapper. http://www.iks-jena.de/mitarb/lutz/usenet/antispam.html.

[68] Paul Grosse. SMTarPit. http://www.fresh.files2.serveftp.net/smtarpit/index.html.

[69] Daniel A. Rehbein. Mit einer Teergrube TCP/IP-Verbindungen blockieren. http://www.mailbox-internet.de/teergrube.html.

[70] MailChannels Corporation. Email connection management and traffic shaping through traffic control. http://www.mailchannels.com/connectionmanagement.html.

[71] SMTP tar pit feature for Microsoft Windows Server 2003. Microsoft Knowledge Base Article 842851, Revision 9.1, October 27 2006. Available at http://support.microsoft.com/kb/842851.

[72] John R. Levine. An overview of e-postage. Technical report, Taughannock Networks, February 2004. http://www.taugh.com/epostage.pdf.

[73] Martín Abadi, Andrew Birrell, Michael Burrows, Frank Dabek, and Ted Wobber. Bankable postage for network services. In Vijay A. Saraswat, editor, *ASIAN*, volume 2896 of *Lecture Notes in Computer Science*, pages 72–90. Springer, 2003.

[74] Brad Templeton. E-Stamps. http://www.templetons.com/brad/spam/estamps.html.

[75] Cynthia Dwork and Moni Naor. Pricing via processing or combatting junk mail. In *CRYPTO '92: Proceedings of the 12th Annual International Cryptology Conference on Advances in Cryptology*, pages 139–147, London, UK, 1993. Springer-Verlag.

[76] Adam Back. Hashcash - a denial of service counter-measure. http://www.hashcash.org/papers/hashcash.pdf, August 2002.

[77] Eric S. Johansson. Camram. http://www.camram.org/.

[78] Paul Roberts. Spammers using sender authentication too, study says. http://www.infoworld.com/article/04/08/31/HNspammerstudy_1.html, August 2004.

[79] J. Galvin, S. Murphy, S. Crocker, and N. Freed. Security Multiparts for MIME: Multipart/Signed and Multipart/Encrypted. RFC 1847 (Proposed Standard), October 1995.

[80] S. Dusse, P. Hoffman, B. Ramsdell, L. Lundblade, and L. Repka. S/MIME Version 2 Message Specification. RFC 2311 (Informational), March 1998.

[81] J. Callas, L. Donnerhacke, H. Finney, and R. Thayer. OpenPGP Message Format. RFC 2440 (Proposed Standard), November 1998.

[82] Meng Weng Wong and Wayne Schlitt. Sender Policy Framework (SPF) for authorizing use of domains in e-mail, version 1. RFC 4408 (Experimental), April 2006.

[83] Sender Policy Framework. http://www.openspf.org/.

[84] SRS: Sender Rewriting Scheme. http://www.openspf.org/SRS, December 2006.

[85] Jim Lyon and Meng Weng Wong. Sender ID: Authenticating e-mail. RFC 4406 (Experimental), April 2006.

[86] Sender ID Home Page. http://www.microsoft.com/senderid/.

[87] Jim Lyon. Purported Responsible Address in e-mail messages. RFC 4407 (Experimental), April 2006.

[88] DomainKeys Identified Mail (DKIM). http://www.dkim.org/.

[89] Identified Internet Mail. http://www.identifiedmail.com/.

[90] E. Allman, J. Callas, M. Delany, M. Libbey, J. Fenton, and M. Thomas. DomainKeys Identified Mail (DKIM) signatures. Internet-draft, IETF, January 2007. draft-ietf-dkim-base-08.

[91] The on-line hacker Jargon File, version 4.4.7. http://catb.org/jargon/html/P/pink-contract.html, October 2003.

[92] Make LOVE not Spam. http://mlns.starring.se/.

[93] Eran Reshef and Eilon Solan. The effects of antispam methods on spam mail. In *Proceedings of the Third Conference on Email and Anti-Spam (CEAS 2006)*, pages 142–151, 2006. Also available at http://www.ceas.cc/2006/24.pdf.

[94] Ryan Singel. Wired news: Under attack, spam fighter folds. http://www.wired.com/news/technology/0,70913-0.html, May 2006.

[95] Wikipedia. Dendrobates azureus — Wikipedia, The Free Encyclopedia. http://en.wikipedia.org/w/index.php?title=Dendrobates_azureus&oldid=151447662, 2007. [Online; accessed 9-October-2007].

[96] Wikipedia. Okopipi (software tool) — Wikipedia, The Free Encyclopedia. http://en.wikipedia.org/w/index.php?title=Okopipi_%28software_tool%29&oldid=97328490, 2006. [Online; accessed 2007-01-11].

[97] Brian McWilliams. Chongq and the spam vampires. http://www.oreillynet.com/pub/a/network/2004/12/03/chongq.html, March 2004.

[98] David Dittrich. How bad an idea was 'Make Love Not Spam?' Let me count the ways. http://staff.washington.edu/dittrich/arc/workshop/lycos-response-v3.txt, 2005.

[99] Devin Carraway. Sugarplum – spam poison. http://www.devin.com/sugarplum/, April 2003.

[100] Tobias Eggendorfer. Private address. Design spam-proof homepages. *Linux Magazine*, pages 26–29, September 2004. Also available at http://w3.linux-magazine.com/issue/46/Spam-proof_Homepages.pdf.

[101] John Graham-Cumming. How to beat an adaptive spam filter. MIT Spam Conference 2004. Slides available at http://www.jgc.org/SpamConference011604.pps.

[102] Daniel Lowd and Christopher Meek. Good word attacks on statistical spam filters. In *Proceedings of the Second Conference on Email and Anti-Spam (CEAS 2005)*, July 2005. Also available at http://www.ceas.cc/papers-2005/125.pdf.

[103] Gregory Lee Wittel and Shyhtsun Felix Wu. On attacking statistical spam filters. In *Proceedings of the First Conference on Email and Anti-Spam (CEAS)*, July 2004. Also available at http://www.ceas.cc/papers-2004/170.pdf.

[104] John Graham-Cumming. Does Bayesian poisoning exist? *Virus Bulletin*, pages S2–S4, February 2006. Also available at http://www.virusbtn.com/spambulletin/archive/2006/02/sb200602-poison.

[105] Jonathan A. Zdziarski. Bayesian noise reduction: Contextual symmetry logic utilizing pattern consistency analysis. MIT Spam Conference 2005. http://www.zdziarski.com/papers/bnr.html.

[106] Jeremy Bowers. (Preliminary) Bayes Attack Report. http://web.archive.org/web/20050206210806/www.jerf.org/writings/bayesReport.html, February 2003.

[107] Terry Sullivan. The more things change: Volatility and stability in spam features. MIT Spam Conference 2004. http://www.qaqd.com/research/mit04sum.html.

[108] Calton Pu and Steve Webb. Observed trends in spam construction techniques: A case study of spam evolution. In *Proceedings of the Third Conference on Email and Anti-Spam (CEAS 2006)*, pages 104–112, 2006. Also available at http://www.ceas.cc/2006/4.pdf.

[109] Bruce Guenter. SPAM Archive. http://www.untroubled.org/spam/.

[110] CipherTrust. SpamArchive.org. ftp://mirrors.blueyonder.co.uk/sites/ftp.spamarchive.org/pub/archives/submit/. SpamArchive.org ceased operation in July 2006, thus the original website http://www.spamarchive.org/ is no longer accessible.

[111] Ryan Thoryk. TLIQuest spam archives. http://web.archive.org/web/20051104234750/http://www.tliquest.net/spam/archive/. This spam archive ceased operation in May 2006, thus the original website http://www.tliquest.net/spam/archive/ is no longer accessible.

[112] John Graham-Cumming. The Spammers' Compendium. http://www.jgc.org/tsc/.

[113] Grady Ward. Moby Thesaurus. http://www.dcs.shef.ac.uk/research/ilash/Moby/mthes.html. Also available at http://www.gutenberg.org/etext/3202.

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

[115] F-Prot Antivirus. http://www.f-prot.com/.

[116] About Gmail. http://mail.google.com/mail/help/intl/en/about.html.

[117] Yahoo! Mail. http://www.yahoo.com/r/m2.

A The very first Spam Email (from 1978)

```
Mail-from: DEC-MARLBORO rcvd at 3-May-78 0955-PDT
Date:  1 May 1978 1233-EDT
From: THUERK at DEC-MARLBORO
Subject: ADRIAN@SRI-KL
To:   DDAY at SRI-KL, DAY at SRI-KL, DEBOER at UCLA-CCN,
To:   WASHDC at SRI-KL, LOGICON at USC-ISI, SDAC at USC-ISI,
To:   DELDO at USC-ISI, DELEOT at USC-ISI, DELFINO at USC-ISI,
To:   DENICOFF at USC-ISI, DESPAIN at USC-ISI, DEUTSCH at SRI-KL,

[386 lines of receiver addresses omitted for better readability]

ZEGERS@SRI-KL
ZOLOTOW@SRI-KL
ZOSEL@LLL-COMP
DIGITAL WILL BE GIVING A PRODUCT PRESENTATION OF THE NEWEST MEMBERS OF THE
DECSYSTEM-20 FAMILY; THE DECSYSTEM-2020, 2020T, 2060, AND 2060T.  THE
DECSYSTEM-20 FAMILY OF COMPUTERS HAS EVOLVED FROM THE TENEX OPERATING SYSTEM
AND THE DECSYSTEM-10 <PDP-10> COMPUTER ARCHITECTURE.  BOTH THE DECSYSTEM-2060T
AND 2020T OFFER FULL ARPANET SUPPORT UNDER THE TOPS-20 OPERATING SYSTEM.
THE DECSYSTEM-2060 IS AN UPWARD EXTENSION OF THE CURRENT DECSYSTEM 2040
AND 2050 FAMILY. THE DECSYSTEM-2020 IS A NEW LOW END MEMBER OF THE
DECSYSTEM-20 FAMILY AND FULLY SOFTWARE COMPATIBLE WITH ALL OF THE OTHER
DECSYSTEM-20 MODELS.

WE INVITE YOU TO COME SEE THE 2020 AND HEAR ABOUT THE DECSYSTEM-20 FAMILY
AT THE TWO PRODUCT PRESENTATIONS WE WILL BE GIVING IN CALIFORNIA THIS
MONTH.  THE LOCATIONS WILL BE:

               TUESDAY, MAY 9, 1978 - 2 PM
                   HYATT HOUSE (NEAR THE L.A. AIRPORT)
                   LOS ANGELES, CA

               THURSDAY, MAY 11, 1978 - 2 PM
                   DUNFEY'S ROYAL COACH
                   SAN MATEO, CA
                   (4 MILES SOUTH OF S.F. AIRPORT AT BAYSHORE, RT 101 AND RT 92)

A 2020 WILL BE THERE FOR YOU TO VIEW. ALSO TERMINALS ON-LINE TO OTHER
DECSYSTEM-20 SYSTEMS THROUGH THE ARPANET. IF YOU ARE UNABLE TO ATTEND,
PLEASE FEEL FREE TO CONTACT THE NEAREST DEC OFFICE
FOR MORE INFORMATION ABOUT THE EXCITING DECSYSTEM-20 FAMILY.
```

Source: [23]

B Examples

In this appendix, several emails are shown before and after they were manipulated by a substitution attack. The first three examples (Example 1–Example 3) are examples of messages that are well suited for a substitution attack, that is, examples of spam emails that were not recognized by the spam filters after the substitution. The last three examples (Example 4–Example 6) are examples of the opposite, that is, these messages were recognized by the spam filters in spite of the substitution of their suspicious words.
Emails before substition were taken from Test Set C, and emails after substitution were taken from Test Set D (see Chapter 5). For better distinguishability, words that were replaced with synonyms are printed in boldface type.

B.1 Example 1

B.1.1 Before Substitution

Subject: Hey bro, found this **site**

Finally the **real** thing - no **more** **tip**-offs! Enhancment Patches are **hot** right now, **VERY** **hot**! Unfortunately, **most** are **cheap** imitations and do very little to **increase** your **size** and **stamina**. Well this is the **real** thing, not an **imitation**! **One** of the very originals, the **absolutely** strongest **Patch** available, anywhere!

A **top** team of British scientists and **medical** doctors have worked to **develop** the state-of-the-**art** **Penis** Enlargment **Patch** delivery system which automatically increases **penis** **size** up to 3-4 **full** inches. The patches are the easiest and **most** **effective** way to **increase** your **size**. **You** won't have to take pills, get under the **knife** to perform **expensive** and very **painful** **surgery**, use any pumps or other devices. **No** one will **ever** find out that you are using our **product**. Just apply one **patch** on your **body** and **wear** it for 3 days and you will start noticing **dramatic** results.

Millions of **men** are taking **advantage** of this **revolutionary** new **product** - Don't be left **behind**!

As an added **incentive**, they are **offering** **huge** **discount** specials right now, check out the **site** to see for **yourself**!

Here's the **link** to check **out**!

Name	Patches	**Regular**	**Now**	
Steel Package	10 Patches	$79.95	$49.95	**Free shipping**
Silver Package	25 Patches	$129.95	$99.95	**Free shipping** and **exercise** manual included
Gold Package	40 Patches	$189.95	$149.95	**Free shipping** and exercise **manual** included
Platinum **Package**	65 Patches	$259.95	$199.95	**Free shipping** and exercise **manual** included

B.1.2 After Substitution

Subject: Hey bro, found this **hall**

Finally the **right** thing - not **supplemental** **consideration**-offs! Enhancment Patches are **ok** right now, **CRAZY** **serious**! Unfortunately, **prominently** are **sorry** imitations and do very little to **improve** your **group** and **resolution**. Well this is the **right** thing, not an **unreal**! **Mixed** of the very originals, the **ok** strongest **Period** available, anywhere!

A **better** team of British scientists and **doc** doctors have worked to **improve** the state-of-the-**work** **Rocks** Enlargment **Heal** delivery system which automatically increases **testicles** **categorize** up to 3-4 **total** inches. The patches are the easiest and **prominently** **important** way to **improve** your **rank**. **She** won't have to take pills, get under the **point** to perform **valuable** and very **tough** **chopping**, use any pumps or other devices. **Withholding** one will **always** find out that you are using our **work**. Just apply one **section** on your **head** and **run** it for 3 days and you will start noticing **overdone** results.

Millions of **people** are taking **domination** of this **original** new **account** - Don't be left **butt**!

As an added **stimulus**, they are **present great waive** specials right now, check out the **bearings** to see for **me**!

Here's the **date** to check **excuse**!

Fix	Patches	**Accepted**	**As**	
Stone Bolt	10 Patches	$79.95	$49.95	**Bluff transport**
In Wedding	25 Patches	$129.95	$99.95	**Bluff transport** and exercise **guide** included
Or Wedding	40 Patches	$189.95	$149.95	**Bluff transport** and exercise **guide** included
Platinum **Wedding**	65 Patches	$259.95	$199.95	**Bluff transport** and exercise **guide** included

B.2 Example 2

B.2.1 Before Substitution

Subject: Re:

Need some love pills? So, why go to your **local drugstore**? **Why waste** time and extra **money**? **Why** let people know about your **intimate life**? Evil-wishers are **always** around to spread rumors.
We give you the issue! Make a **quick**, **secure** and ABSOLUTELY CONFIDENTIAL purchase online and receive your **LICENSED** love **life** enhancer right to your door! **No privacy** exposure, **no** time **wasted**, no exorbitant prices! **Start** a **super life now**!
http://andturn.com/
Our store is **VERIFIED BY** BBB! **All** transactions are **APPROVED BY VISA**!

B.2.2 After Substitution

Subject: Re:

Need some love pills? So, why go to your **topical bookstore**? **Explanation go** time and extra **cash**? **Question** let people know about your **personal relations**? Evil-wishers are **usually** around to spread rumors.
We give you the issue! Have a **good**, **easy** and ABSOLUTELY CONFIDENTIAL purchase online and receive your **AUTHORIZED** love **story** enhancer right to your door! **Withholding retirement** exposure, **unwillingness** time **spent**, not exorbitant prices! **Develop** a **better guy for**!
http://andturn.com/
Our house is **DETERMINED HEREWITH** BBB! **Bodily** transactions are **RECEIVED HEREWITH AUTHORITY**!

B.3 Example 3

B.3.1 Before Substitution

Subject: **Pharmacy** - **No prescription** required **bloke**

Since you are **always** looking out for me, I want to let you know about this
great chance that you can **save** on **health** products.
The **discount** prices of these **health** pills are up to 80% off the **retail**
price.

You have to find this **major** deals **heaven** I've found on the internet.
And the orders are delivered expedited to my apartment.
I can't wait until you go to their internet outlet because from now on, your
life is going to be much easier.

http://xriw.info/?3b6Sc1117245c2eS4d384a6b6fe1b8c3

B.3.2 After Substitution

Subject: **Ward** - **Disclaimer alodium** required **cat**

Because you are **often** looking out for me, I want to let you know about this
great chance that you can **help** on **realism** products.
The **grant** prices of these **form** pills are up to 80% off the **job**
parity.

She have to find this **seminar** deals **peak** I've found on the internet.
And the orders are delivered expedited to my apartment.
I can't wait until you go to their internet outlet because from now on, your
elasticity is going to be much easier.

http://xriw.info/?3b6Sc1117245c2eS4d384a6b6fe1b8c3

B.4 Example 4

B.4.1 Before Substitution

Subject: If you start taking **Penis Enlarge Patch**, the world's **attention** will be **drawn** to
your dick.

http://www.werfop.com/pt/?108&YuFkKScGTX8qY
Try Penis Enlarge Patch by **yourself** and tell your friend about it.
His gratitude will be bigger **than life**.

B.4.2 After Substitution

Subject: If you start taking **Meat Crescendo Croft**, the world's **diligence** will be **tied** to
your dick.

http://www.werfop.com/pt/?108&YuFkKScGTX8qY
Torture Meat Pump Area by **me** and tell your friend about it.
His gratitude will be bigger **elsewise one**.

B.5 Example 5

B.5.1 Before Substitution

Subject: Online Mexican **Pharmacy**

Want New Mexican Drugs? http://www.cxsq.com/

B.5.2 After Substitution

Subject: Online Mexican **Emergency**

Love Another Mexican Drugs? http://www.cxsq.com/

B.6 Example 6

B.6.1 Before Substitution

Subject: with RYAN IAN examined ROY

Wineco Productions Inc.
Mining Play
Symbol: WNCP
Price: $0.035

Go Look at the Charts of FGOVF, AGXM, PNAMF, WGDF and AOOR. **All Mining** Plays that **Have** Had Nice Runs **Recently**. **Hot Sector** or What? **Have You Ever** Had a **Big Winner**? Is The **Current** Action in The **Stock** the "Primal Stirring" **Before** a Big **Breakout**?

If **You Think** WNCP is a **WINNER, You** May **Want** to **Jump** on Board IMMEDIATELY!!

ON MARCH 17TH THE **COMPANY** DISCLOSED IT WAS **IN** NEGOTIATIONS WITH 5 COMPANIES!!

TWO ANNOUNCEMENTS **HAVE** BEEN ISSUED:

1)Wineco Productions Inc. Signs Letter of **Intent** with **Mineral** Reclamation Corp.

2)Wineco Productions Announces the **Acquisition** of Mine Tailings from **World Wide** Consulting

IS WNCP **ONE NEWS** ANNOUNCEMENT **AWAY FROM GOING** ABSOLUTELY BALLISTIC???

RADAR IT **FOR** WEDNESDAY'S OPEN!

_____

Information within this **report** contains forward looking statements within the **meaning** of **Section** 27A of the Securities **Act** of 1933 and **Section** 21B of

the SEC Act of 1934. Statements that involve discussions with respect to
projections of future events are not statements of historical fact and may
be forward
looking statements. Don't rely on them to make a decision. The Company is
not a reporting company registered under the Exchange Act of 1934. We have
received
two million three hundred thousand free trading shares from a third party
not an
officer, director or affiliate shareholder. We intend to sell all our shares
now,
which could cause the stock to go down, resulting in losses for you. This
company has:
an accumulated deficit and and a reliance on loans from officers and affiliates
to pay expenses. It is not an operating company. The company is going to need
financing to
continue as a going concern.The agreements above may not be definitive and may
not occur.
A failure to finance could cause the company to go out of business. This
report shall not be construed as any kind of investment advice or solicitation.
You can lose all your money by investing in this stock.

B.6.2 After Substitution

Subject: with RYAN IAN examined ROY

Wineco Productions Inc.
Expression Recreation
Baton: WNCP
Consideration: $0.035

Do See at the Charts of FGOVF, AGXM, PNAMF, WGDF and AOOR. Any Making
Plays that Admit Had Nice Runs Before. Bad Item or What? Tell She
Still Had a Big Star? Is The Latest Action in The Beginning the "Primal
Stirring" To a Big Release?

If She See WNCP is a MASTER, She May Need to Pass on Board IMMEDIATELY!!

IN WALK 17TH THE FOLLOWING DISCLOSED IT WAS INCOMING NEGOTIATIONS WITH 5 COMPANIES!!

COUPLE ANNOUNCEMENTS BEAR BEEN ISSUED:

1)Wineco Productions Inc. Signs Letter of Understanding with Clay Reclamation
Corp.

2)Wineco Productions Announces the Purchase of Mine Tailings from Public
Neutral Consulting

IS WNCP PERMANENT NEWSPAPER ANNOUNCEMENT STRAIGHT EXCLUDING OPERATING ABSOLUTELY BALLISTIC???

FIX IT WHEREAS WEDNESDAY'S OPEN!

_____

Bail within this book contains forward looking statements within
the purpose of Class 27A of the Securities Law of 1933 and Response 21B of
the SECOND Regulation of 1934. Statements that contain discussions with materiality to
projections of constellation events are not statements of real data and may
be forward
looking statements. Don't look on them to make a decision. The Following is
not a reporting agency filed under the Message Bluff of 1934. We have
received
two crore three cwt zillion unconstrained trading shares from a second approver
not an
official, manager or pass shareholder. We plan to handle all our shares
now,
which could have the game to go down, resulting in losses for you. This
group has:
an accumulated net and and a dependence on loans from officers and affiliates
to know expenses. It is not an ongoing aggregation. The following is going to need
support to
continue as a going materiality.The agreements above may not be final and may
not be.
A failure to finance could have the following to go out of engagement. This
information shall not be construed as any way of providing counsel or selling.
She can spend all your stuff by investing in this fund.