Phase 1 of Project “Spamabwehr II”

W. Gansterer, M. Ilger, A. Janecek, P. Lechner, J. Strauß

Technical Report FA384018-3

Institute of Distributed and Multimedia Systems
Faculty of Computer Science
University of Vienna, Austria

December 2005
This report summarizes the activities of phase 1 of the project FA 384018 "Spamabwehr 2" of the Institute of Distributed and Multimedia Systems at the University of Vienna, funded by Internet Privatstiftung Austria (IPA), mobilkom austria, and UPC Telekabel.

Copyright: ©2005 by University of Vienna. All rights reserved. No part of this publication may be reproduced or distributed in any form or by any means without the prior permission of the authors. The Institute of Distributed and Multimedia Systems at the University of Vienna does not guarantee the accuracy, adequacy or completeness of any information and is not responsible for any errors or omissions or the result obtained from the use of such information.

About the Authors: Project "Spamabwehr 2" was launched in May 2005 at the Institute of Distributed and Multimedia Systems (Distributed Systems group) at the Faculty of Computer Science of the University of Vienna.

The Team: Dr. Wilfried Gansterer (project leader), Michael Ilger (part time), Andreas Janecek (part time), Peter Lechner (part time), and Jürgen Strauß (full time).

The institution: The Faculty of Computer Science (Fakultät für Informatik) is lead by Dean Prof. Dr. Günter Haring. The Institute of Distributed and Multimedia Systems, headed by Prof. DDr. Gerald Quirchmayr, is one of the working groups within this faculty.

Contact for Project "Spamabwehr":
phone: +43-1-4277-39652
e-mail: wilfried.gansterer@univie.ac.at, spamabwehr@ani.univie.ac.at

Institute of Distributed and Multimedia Systems
University of Vienna
Lenaugasse 2/8, A-1080 Vienna (Austria)
Contents

Executive Summary 4

1 Properties of Spam Data 6
  1.1 Spam-Mail Statistic ......................................................... 6
  1.1.1 Filling the Database .................................................. 6
  1.1.2 Queries .......................................................................... 7
  1.1.3 History and Export ....................................................... 7
  1.2 Status and Outlook ......................................................... 9

2 Analysis of Spammers’ Business Model 10
  2.1 SpamSim ........................................................................... 10
  2.2 Status and Outlook ......................................................... 11

3 Analysis of Antispam Methods 12
  3.1 Different Training Schedules for Mozilla Thunderbird ................. 12
  3.1.1 Test Setup ......................................................................... 12
  3.1.2 Results ........................................................................... 13
  3.1.3 Summary ......................................................................... 16
  3.2 Performance Evaluation of SpamAssassin Rules ............................. 16

4 Development of New Methods 17
  4.1 Token Buckets for Outgoing Spam Prevention ............................ 17
  4.1.1 Basic Idea ........................................................................ 17
  4.1.2 Implementation ............................................................... 17
  4.1.3 Parametrization .............................................................. 18
  4.2 A New Antispam System ...................................................... 18
  4.2.1 Update on EPF ............................................................... 18
  4.2.2 Current Status of the Prototype ....................................... 19

5 Conclusions and Outlook 20
Executive Summary

This report summarizes the current status of the research project FA384018 “Spamabwehr 2” which was launched in May 2005 at the Institute of Distributed and Multimedia Systems at the University of Vienna and is supported by Internet Privatstiftung Austria (IPA), mobilkom Austria, and UPC Telekabel.

In this phase, we mainly focussed on four topics: (i) analyzing the properties and characteristics of spam messages in order to have a sound basis for methodological innovations; (ii) analyzing the business model underlying the spamming phenomenon in order to investigate approaches which target the main motivation behind spamming (fighting the problem at its root instead of treating symptoms); (iii) analyzing the performance of widely used antispam methods (Bayes filter, individual rules of the rule set of SpamAssassin, probably the currently most popular antispam tool); and (iv) proceeding with the development of our new approaches.

Chapter 1: Properties of Spam Data

Many methods used for spam defense (especially rule sets) rely on specific properties of spam messages. In order to validate these assumptions and to have a solid foundation for new innovative ideas, we decided to implement a database tool for automatically extracting statistics about certain important properties of spam.

Since September 2004, the Computer Center of the University of Vienna (Zentraler Informatikdienst, ZID) provides us with a continuous stream of spam messages coming from spam traps. We store these messages in mbox files and so far used them only for performance tests.

However, we did not have any tools available for analyzing the properties of this spam data. Thus, we developed the concept for a tool “Spam-Mail Statistic”, which was implemented by Mr. Markus Daberger. It reads messages from MBOX files or IMAP folders and stores the most important information about these messages in a database. Predefined and self-defined statistical queries allow the methodical analysis of the stored data and the investigation of relations between them (including comparisons between spam and ham collections). The statistics can be displayed in tables or graphs. Mr. Daberger is currently working on a Master’s thesis in the context of which he will use and further develop this tool.

Chapter 2: Analysis of Spammers’ Business Model

The different business models underlying spamming have been summarized in [1] and [2]. It has also been indicated that technical antispam solutions which aim at lowering spammers’ profits (ideally so far that they lose commercial interest) are essential components in comprehensive antispam measures.

In this chapter we summarize the concept and the ongoing development of “SpamSim”, a tool for analyzing the business models of spammers under variable boundary conditions. SpamSim implements various business models of spammers. We can simulate the influence of crucial factors (for example, the response rate) on the profits achieved. This allows us to simulate the influence of technical solutions and
even legal measures on the business model of spammers. Therefore, we can perform an *a priori* evaluation of cost-based spam prevention methods.

**Chapter 3: Analysis of Antispam Methods**

Among the most interesting “post-send” filtering methods for detecting spam is Bayes-filtering. From the point of view of an educated user this method may even seem as an acceptable solution to the spam problem, because in certain situations it may detect a reasonably high percentage of spam.

However, there are obvious disadvantages of Bayes filters. First of all, they act “post-damage” and thus are of limited value from the point of view of an ISP. Moreover, the maintenance effort is significant. Our goal in this phase of the project was to investigate this maintenance effort and the dependence of the performance of Bayes filters on the training effort invested.

For this purpose, Mr. Daniel Bründl tested two Bayes filters: the open source Bogofilter and the filter of Mozilla Thunderbird. The latter one was evaluated over a period of 52 weeks with three different training schedules (training only once initially, training periodically every four weeks, or training periodically every two weeks). The results confirm the dependency of detection rates on the training effort.

Bayes filters are an integral component of many antispam systems, also of the probably most widespread tool, SpamAssassin, which uses more than 600 tests and checks in its rule set. For the development of an efficient antispam system it is highly relevant to know which of these tests are the “important ones” for a generic spam detection on the server side. Based on the spam data collected from ZID, we therefore started a project analyzing the performance of individual tests in the SpamAssassin set. In the context of his Master’s thesis, Mr. Michael Nußbaumer performs this analysis. The goal is to “rank” the SpamAssassin tests according to their detection rates as well as according to their resource demands.

**Chapter 4: Development of New Methods**

Our research into new methods currently proceeds in two main directions. On the one hand, we have adapted the token bucket concept for spam prevention [3]. The basic idea is to harm the business model underlying spamming, virtually without affecting regular e-mail users. The concept should ideally be utilized by ISPs not only for reducing the amount of spam on the internet but also for *self protection*.

On the other hand, based on the preliminary work and on the prototype definition described in [2], we are developing a flexible and adaptive server-side antispam solution. A distinctive feature is the combination of a new core spam detection engine based on state-of-the-art classification techniques with an optional greylisting component, which makes the system highly flexible in terms of resource demand and essentially immune to denial-of-service attacks. Implementation is in progress and first experimental evaluations are expected in early 2006.

**Chapter 5: Conclusions and Outlook**

Our research clearly indicates that it is very unlikely that a single approach will “solve” the spam problem. Various components have to act together: First of all, the mechanisms behind the spam phenomenon have to be thoroughly understood. This allows one to design adaptive strategies based on fundamental principles which are more effective and persistent than widespread ad-hoc solutions based on static rule sets. Moreover, a comprehensive antispam solution has to integrate approaches from different sides, controlling incoming and outgoing traffic, combining authentication and classification. As long as the basis is the current mail transfer infrastructure (SMTP, etc.), neither authentication nor classification is by itself sufficient to handle all of the possible spamming scenarios.
Chapter 1

Properties of Spam Data

Representative training data sets are indispensable for evaluation and performance tests of antispam methods. Whereas collecting ham messages is difficult as it depends on volunteers, spam messages can be collected automatically via spam traps. A spam trap is a mailbox which is not used by a human and has only one intention – to collect spam mail for further analysis. ZID collects spam e-mail via a variety of spam traps. We get a copy of the messages arriving at the ZID spam traps and store them in mbox files.

However, due to the enormous amount of data coming in through these spam traps it is very difficult to have a good insight into important properties of the spam data. Nevertheless, the performance of many methods for spam defense (Bayes filters, many rule sets, etc.) critically depends on such properties. Therefore, we decided to implement a database tool called “Spam-Mail Statistic” for automatically extracting statistics about certain important properties of spam. The remainder of this chapter summarizes the status of this effort.

1.1 Spam-Mail Statistic

Spam-Mail Statistics is a tool developed for an in-depth analysis of spam and ham messages. The text of the messages is split into parts, important information is extracted from these parts and then stored in a database together with the original message. It is possible to perform statistical queries with different parameters on this database. At the moment we use a MySQL database, but the system contains a database abstraction layer which allows to switch to other database management systems. The tool is implemented as a stand-alone Java 5-application Its main functionality is described in the following.

1.1.1 Filling the Database

The tool can load messages in mbox format or from imap-folders. It extracts the most relevant properties from each message, for example

- send and receive date,
- message size,
- number of lines,
- number of “RECEIVED” lines,
- language (english or german or other language),
• a list of words occurring in the message and their frequencies, etc.

and saves them into the database. Besides these properties the original e-mail itself is saved for later queries. Spam and ham can be read separately, which makes comparative analysis possible.

1.1.2 Queries

The main purpose of the tool is the analysis of spam properties. It offers some predefined queries, such as getting the number of received lines in each e-mail, or getting averages and standard deviations of parameters over some period. It is possible to add conditions for every extracted value to get a customized result set, and it also allows for self-defined queries. More specifically, the user can query all e-mails in the database where certain parameters occur or do not exist, or where they are equal to, less than or greater than user-defined values.

The results of queries are displayed in tables (see Fig. 1.1) and in most cases also as graphs (see Fig. 1.2). A table contains, depending on what the user is asking for, either a list of values found in the database and the result of an addition (for example, a list of days in ascending order and the number of e-mail sent on each day), or statistics such as the total sample size, the minimum, average and maximum value of the sample, the sample variance and the standard deviation. Fig. 1.3 shows a typical query – the number of spam messages which arrived in a certain time period.

Figure 1.1: Number of spam messages arriving on specific days

1.1.3 History and Export

Every query result is saved and can be reviewed later. This makes it possible to compare query results when more or other e-mails have been inserted into the database between queries. Every single query result, but also the sum of all results can be exported to a csv-file.
Figure 1.2: Graph of Spam-Mail Statistic (number of messages arriving daily in a certain time period)

Figure 1.3: Query for the number of e-mail messages aggregated by date
1.2 Status and Outlook

Except for some minor details and enhancements the tool is fully implemented and ready to use. It will now be used in the context of Mr. Daberger’s Master’s thesis to analyze a big amount of spam- and ham-mail which we collected over a year from the ZID spamtraps and from other sources.

Currently, we are starting the process of filling the database with historical e-mail data, which we expect to be the most time consuming part. Once this is finished, we can start analyzing the data. The result of this analysis will allow us to verify or falsify the assumptions of widely used antispam methods and also provide a solid foundation for new innovative ideas.
Chapter 2

Analysis of Spammers’ Business Model

The different business models motivating the spam phenomenon have been summarized in [1] and [2]. In order to design and calibrate an antispam method which is capable of interfering with this business model, a simulation tool for modelling the effect of various parameters onto the profit achieved by spammers is designed.

We developed the concept for “SpamSim”, a tool which implements the most important cost and profit factors and models their interdependencies. Using this tool, it will become possible to answer questions such as “What percentage of the spam messages has to be filtered out in order to make spamming unprofitable?”, “How many messages does a spammer have to send out in order to be profitable?”, etc.

2.1 SpamSim

The goal of spammers is to make money fast, which is possible due to the low cost for sending e-mail [4]. Some antispam methods target the business model of spammers by increasing the cost associated with sending e-mail (see, for example, [1]).

In [2] a very brief discussion of a few concrete scenarios of spammers’ business is given. SpamSim is a tool for the simulation of the economical background of spamming and the impact of different antispam methods used with different parameters. Thus, it becomes possible to simulate a variety of much more detailed scenarios and to compare them. By examining the break-even point for spammers in terms of cost and profit, we can evaluate the effectiveness of various antispam methods.

The tool is implemented as a Windows GUI (Graphical User Interface) application, programmed in Visual C#, operating on the Microsoft .NET framework 1.1 [5]. The development environment is Microsoft Visual Studio 2003.

To provide a flexible solution, the scenarios are defined in XML (eXtensible Markup Language) [6] files. Each scenario is defined by the elements of the cost factors. For each element it is possible to set a default value. This structure is loaded into the fixed cost factor structure of the application, which generates the GUI dynamically.

For the calculation we consider defining the formula in a MathML file. MathML is "...an XML application for describing mathematical notation..." [7]. The calculation is defined for the elements of each cost factor in a scenario and for the whole scenario itself. To improve usability, a standard and an advanced view mode will be provided. In the standard mode the elements for fine adjustment are hidden and the pre-defined default values are used. The different simulation tasks of an application can be saved.
as XML files and loaded again.

The simulation result is displayed as a line diagram (for an example, see Fig. 2.1). The most commonly used diagram shows the number of e-mails sent versus corresponding cost and revenue. The result values can be exported as XML file or as CSV file for further use in spreadsheet or statistical tools. The diagram can also be exported as an image.

![Figure 2.1: Screenshot of SpamSim (line diagram showing number of messages sent out versus corresponding cost and revenue)](image)

### 2.2 Status and Outlook

Currently the dynamic loading of the elements from the definition file is working and the GUI is designed in its basics (with the functionality of displaying a diagram – compare Fig. 2.1).

After implementing the dynamic definition of the calculation the first analysis can start. In that phase, some further enhancements, especially of the GUI, are also planned. At this stage the differentiation between the parameters of advanced and standard mode will be done, too.

Using this tool the efficiency of new technical antispam methods can be evaluated theoretically. Highly successful antispam approaches which make spamming unprofitable can be identified.
Chapter 3

Analysis of Antispam Methods

In this chapter, we summarize some detailed performance evaluation of two very popular and wide-spread open source software tools, which both fall in the category “post-send”/filtering approaches: the Bayes filter implementation in the public domain e-mail client Mozilla Thunderbird [8] and the more comprehensive SpamAssassin filter [9] (which also includes a Bayes filter).

From the point of view of an experienced user such tools may seem to be an acceptable solution to the spam problem, because under certain circumstances they may detect a reasonably high percentage of spam. However, there are several important inherent disadvantages of filtering approaches such as Bayes filters or the rule sets of SpamAssassin. First of all, they act “post-damage” and thus are of limited value from the point of view of an ISP. Moreover, their maintenance effort is significant. And, in the case of SpamAssassin, many rules are included which tend to be only of marginal importance and thus cause unnecessary load on the computational resources.

One of our goals in this phase of the project is to investigate these disadvantages of commonly accepted “solutions” to the spam problem and to carefully evaluate two aspects: (i) the dependence of the performance of a Bayes filter on the training effort invested (maintenance effort), and (ii) the individual detection rate and overall significance of each SpamAssassin rule. The results of this evaluation will form the basis for the classification engine of our prototype currently under development (see Section 4.2).

3.1 Different Training Schedules for Mozilla Thunderbird

Mozilla Thunderbird uses a Bayes filter to identify spam messages. Any Bayes filter must be trained, at least initially. In order to evaluate the dependency of the performance of the Thunderbird Bayes filter on the frequency of training/on the training effort, we tested the filter performance with three different training schedules over a period of 52 weeks.

We compared the following three training schedules: (i) “static”—train the Bayes filter only once (initially), (ii) “dynamic four weeks”—train the Bayes filter periodically every four weeks, and (iii) “dynamic two weeks”—train the Bayes filter periodically every two weeks. Intuitively, one would expect the best detection rates with the filter trained most frequently (biweekly) and a significant decrease of the detection rate over time for the filter trained only initially. It was our goal to substantiate this intuition and to quantify it using empirical evaluations.

3.1.1 Test Setup

We tested version 1.0.6 of Mozilla Thunderbird. The spam threshold was set to .99 (using the command “mail.adaptivefilters.junk_threshold”,99” in file user.js).
The test samples used were subsets of the spam coming from the spam traps of the ZID (cf. Chapter 1). We extracted spam from 52 consecutive Tuesdays, ranging from October 5th, 2004, to September 27th, 2005. Fig. 3.1 shows average monthly sizes (arithmetic mean) of these Tuesday samples. For training the filter we took samples of 200 spam and ham messages, respectively. When training periodically, we always took different samples. The ham was collected from volunteers.

### 3.1.2 Results
The results of this evaluation are shown in Fig. 3.2. Summarizing, we note that

1. the detection rate of the static variant is significantly lower than the detection rates of the dynamic variants (as expected).
2. the detection rates of the two dynamic variants do not differ much (not very surprising).
3. there seems to be no general decreasing trend in the detection rate of the static variant (rather surprising).
Figure 3.2: Detection rates of Mozilla Thunderbird (Bayes filter) for different training schedules over a 52 week period
<table>
<thead>
<tr>
<th>week #</th>
<th>date</th>
<th>messages total</th>
<th>bounces</th>
<th>fraction of bounces</th>
<th>dr (2weeks/4weeks/static)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>05.10.04</td>
<td>10809</td>
<td>5439</td>
<td>0.50</td>
<td>0.93 / 0.93 / 0.93</td>
</tr>
<tr>
<td>2</td>
<td>12.10.04</td>
<td>14603</td>
<td>7957</td>
<td>0.54</td>
<td>0.93 / 0.93 / 0.93</td>
</tr>
<tr>
<td>3</td>
<td>19.10.04</td>
<td>11177</td>
<td>5683</td>
<td>0.51</td>
<td>0.83 / 0.92 / 0.92</td>
</tr>
<tr>
<td>4</td>
<td>26.10.04</td>
<td>7942</td>
<td>3386</td>
<td>0.43</td>
<td>0.72 / 0.88 / 0.88</td>
</tr>
<tr>
<td>5</td>
<td>02.11.04</td>
<td>15377</td>
<td>7284</td>
<td>0.47</td>
<td>0.80 / 0.86 / 0.88</td>
</tr>
<tr>
<td>6</td>
<td>09.11.04</td>
<td>3439</td>
<td>182</td>
<td>0.05</td>
<td>0.39 / 0.37 / 0.29</td>
</tr>
<tr>
<td>7</td>
<td>16.11.04</td>
<td>2839</td>
<td>4</td>
<td>0.01</td>
<td>0.85 / 0.55 / 0.67</td>
</tr>
<tr>
<td>8</td>
<td>23.11.04</td>
<td>3876</td>
<td>435</td>
<td>0.11</td>
<td>0.82 / 0.68 / 0.75</td>
</tr>
<tr>
<td>9</td>
<td>30.11.04</td>
<td>3043</td>
<td>4</td>
<td>0.01</td>
<td>0.83 / 0.81 / 0.73</td>
</tr>
<tr>
<td>10</td>
<td>07.12.04</td>
<td>2595</td>
<td>0</td>
<td>0.00</td>
<td>0.84 / 0.79 / 0.71</td>
</tr>
<tr>
<td>11</td>
<td>14.12.04</td>
<td>2723</td>
<td>0</td>
<td>0.00</td>
<td>0.95 / 0.77 / 0.64</td>
</tr>
<tr>
<td>12</td>
<td>21.12.04</td>
<td>2323</td>
<td>3</td>
<td>0.001</td>
<td>0.96 / 0.85 / 0.62</td>
</tr>
<tr>
<td>13</td>
<td>28.12.04</td>
<td>2920</td>
<td>2</td>
<td>0.0007</td>
<td>0.95 / 0.92 / 0.62</td>
</tr>
<tr>
<td>14</td>
<td>04.01.05</td>
<td>6323</td>
<td>2522</td>
<td>0.40</td>
<td>0.95 / 0.94 / 0.90</td>
</tr>
</tbody>
</table>

Table 3.1: Bounces as percentage of overall spam from ZID spam-traps for selected dates. “dr” denotes “detection rate”.

4. after a first sharp decrease of all detection rates after around five weeks, the performance of all three training schedules improves again (very surprising!).

At first sight, especially observations 3. and 4. are very surprising, because they contradict the intuitive expectation. However, a more detailed investigation of the test data reveals some interesting aspects which have been unnoticed so far (due to the big volume of spam data it is usually impossible to screen the messages manually): There are some periods when the fraction of “bounces” (error messages such as delivery failure notices) in the data from the spam traps is very high (see Table 3.1). This is due to the fact that spammers sometimes seem to randomly put harvested e-mail addresses also into the “From:” field of a message (and not only into the “To:” field).

In Fig. 3.2 we show the approximate fraction of bounces in all the test samples as the blue parts in the bars of the bar charts, and in Table 3.1 we show the corresponding raw data for the first 14 weeks. Note that the October samples have a very high portion of bounce messages (around 50%!), whereas the bounce fraction between weeks six and thirteen is very low. After that, it increased again.

**Estimating the Fraction of Bounces**

A bounce message is a notification message returned to the sender of an e-mail indicating that the message could not be delivered. Whereas a “soft bounce” means that a message could not be delivered temporarily, a “hard bounce” indicates that the message could not be delivered permanently (for example, because the recipient does not exist).

Unfortunately, RFCs do not define a standard format for bounces. In order to estimate the fraction of bounces in each spam test set we filtered the messages depending on whether the following keywords appeared in the subject line or not: “Delivery failure”, “Delivery Notification”, “Failure notice”, “Mail could not be delivered”, “Mail Delivery Failure”, “Mails System Error”, “Returned Mail”, “Undeliverable” and “Undelivered”.

This list of keywords is not complete in the sense that it is not guaranteed that we were able to find all bounces (because, as mentioned before, there is no definition of a standard format for failure messages). However, we confirmed experimentally with some of our test sets that this filtering strategy is the best
among various options (finds most of the bounces) and thus yields a very good estimate for the fraction of bounces.

**Relationship Between Fraction of Bounces and Detection Rate**

All bounces have a relatively high degree of structural similarity. This similarity has an important influence on the detection rate of a Bayes filter: Once a bounce message appears in a training set, the probability that later bounces will be detected is high (because of structural similarity). Once bounces have been trained, a high fraction of bounces in a test set tends to artificially increase the detection rate. On the other hand, if the fraction of bounces in a test set is (very) low, all previously trained bounces do not contribute much and thus the detection rate tends to be relatively low.

This is confirmed in Fig. 3.2, which shows that the detection rate is high, if the amount of bounces is high and vice versa. Now we can also explain the somewhat surprising observations made in our experiment. Due to the low fraction of bounces, in weeks six to thirteen we observe essentially the expected behavior of decreasing detection rates and the lowest detection rates for the static variant. After this period, the fraction of bounces increases significantly, which keeps the detection rates at a misleadingly high level.

**3.1.3 Summary**

A Bayes filter that is trained periodically can achieve detection rates of 90% or more. However, since spammers react to antispam methods and change their strategies and techniques, the detection rate tends to decrease (potentially severely) if the Bayes filter is not trained.

Another very important lesson to be learned from our investigations is that detailed information about the properties of spam data from spam traps is required in order to properly analyze experimental results. Without such an understanding, many observations can be quite misleading. Manually examining big numbers of spam messages is obviously prohibitively time consuming, and thus our spam statistic tool outlined in Section 1.1 will be extremely valuable for a rigorous evaluation of existing and newly developed methods.

**3.2 Performance Evaluation of SpamAssassin Rules**

Bayes filters are an integral component of many antispam systems. The currently probably most widespread tool, SpamAssassin, uses more than 600 other rules, tests and checks in its rule set (one of them being a Bayes filter). These rules differ in their significance, classification power, and also in their resource demand. For the development of an efficient antispam system it is consequently highly relevant to know which of these tests are the “important ones” for a generic spam detection on the server side. Based on the spam data collected from ZID, we therefore also started a project analyzing the performance of individual tests in the SpamAssassin set. In the context of his Master’s thesis, Mr. Nußbaumer performs this analysis.

The goal is to “rank” the SpamAssassin tests according to their detection rates as well as according to their resource demands. Based on the results, we can determine how many of these rules can be omitted with a controlled (small) loss in detection performance. This information will also be integrated into our new antispam system (cf. Section 4.2).

Mr. Nußbaumer’s experiments are currently in progress, and the final results will be presented in the next project report.
Chapter 4

Development of New Methods

In this chapter, we briefly summarize our activities in the development of new antispam approaches. This comprises two focus areas: (i) a new concept for outgoing spam prevention (see Section 4.1), and (ii) the ongoing work on our prototype of a new antispam system (see Section 4.2).

4.1 Token Buckets for Outgoing Spam Prevention

We only give a very short review of our concept for outgoing spam prevention based on token buckets. Details can be found in [3] (also available online at http://spam.ani.univie.ac.at).

4.1.1 Basic Idea

The concept of a token bucket was originally developed in the context of network traffic shaping. We have adapted this concept to the context of antispam methods and implemented a plugin to control outgoing e-mail traffic. It allows a flexible and adaptive limitation of outgoing e-mail traffic which can be tuned such that it becomes transparent to the regular e-mail user while impeding the business model of a spammer.

4.1.2 Implementation

Our token bucket approach is implemented as policy delegation server for the open source mail server Postfix [10]. This implies that it can easily be added to our extended Postfix (EPF) mailserver which we introduced in [2] and also into our prototype (see Section 4.2).

For each user, a triplet consisting of (i) the mailbox name, (ii) the time $t_0$ of the last successful submission of the “RCPT TO:” command (SMTP status code “250”), and (iii) the number $T$ of tokens currently available in his bucket is stored. Every time a client transfers a message to the outgoing mail server, a plug-in is triggered after the execution of the “RCPT TO:” command within the SMTP dialogue. If the e-mail has $r$ recipients, the plug-in is executed $r$ times.

Each execution of the plug-in computes the number of available tokens based on the following parameters: capacity $\beta$ (= maximum number of tokens available per user), token consumption per recipient $T_c$, and token growth per time unit $\rho$.

Based on the current system time $t$ (when the sending takes place) and on $t_0$, the new number $T$ of tokens currently available in the bucket can be computed using the following equation:

$$T(t) = \min\left(\min\left(T(t_0) + (t - t_0) \cdot \rho, \beta\right)\right)$$
This number $T(t)$ is the basis for the decision whether the e-mail can be sent out to a certain recipient or not. If $T(t) - T_c \geq 0$, the recipient is accepted, the number of available tokens is updated in the user’s triplet as $T := T(t) - T_c$, and the plug-in returns status code “250 OK” to the mail server. If $T(t) - T_c < 0$, the e-mail cannot be sent, the recipient is refused, and the plug-in returns “554 Not enough tokens available” back to Postfix. Alternatively, one could delay sending of this e-mail until enough tokens are available. Fig. 4.1 shows the processing path of this concept.

4.1.3 Parametrization

The central parameters for the token bucket (capacity $\beta$, token growth rate $\rho$) can be chosen by the ISP. If they are chosen properly, the concept becomes essentially invisible to a regular user, but at the same time it harms the business model of a spammer. More details about how to find a good parametrization are given in [3].

4.2 A New Antispam System

In this section, we first give a brief update on the performance of the EPF server, which we developed in Phase 3 of Project “Spamabwehr I” (cf. [2]). Then, we report on the latest developments on the way towards a prototype for a comprehensive antispam system with very attractive features.

4.2.1 Update on EPF

As described in [2], we have implemented EPF as an extended SMTP server, which includes checks for basic properties of the SMTP-based e-mail transfer process. This can provide indications about whether
an e-mail should be considered spam or not.

In particular, we integrated plugins for open proxy checking, open relay testing and greylisting into a single plugin. We monitored the performance of EPF by directing all incoming traffic to a certain e-mail address through EPF. This e-mail address receives predominantly spam, although due to historical reasons it is not an actual spamtrap. Thus, it cannot be precluded that every once in a while a ham message is sent to this address.

Table 4.1 shows the results of the greylisting component of EPF during the period June 1st to October 23rd, 2005. During this period a total of 9543 messages arrived at this specific e-mail address and they were all greylisted.

The standards define that if a message is blocked (bounces), the sending mail server should try to resend the message after a pre-defined time-out. Spammers’ mail servers are often not fully implemented, and, consequently, in many cases they do not respond to bounced messages. In the above mentioned period, only 35 of the greylisted messages (0.37%) were re-sent by the sending mail server.

4.2.2 Current Status of the Prototype

We are currently developing a prototype for a flexible and adaptive server-side antispam solution. It is based on the open source JAVA SMTP server “Apache James” and implements a multi-layered approach comprising the following components (which can be used independently), both developed and implemented from scratch:

- a very sophisticated and flexible adaptive greylisting strategy; and
- a new core spam detection engine based on state-of-the-art classification techniques.

The greylisting component makes the system highly flexible in terms of resource demand and essentially immune to denial-of-service attacks.

The spam detection engine utilizes a list of properties which is determined for each e-mail. In order to keep this list as short and compact as possible (which keeps the resource demand of our system low) we are working on identifying the “most important” properties of e-mail messages in terms of the spam problem, i.e., the properties with the highest ability to differentiate between spam and ham (cf. Section 3.2). Important strengths of our spam detection engine are the following: (i) it can provide intermediate classification results at any time (even if the checks on the e-mail have not been completed), and (ii) it is able to detect and adapt to “hidden” or newly developing features of spam messages.

As of today, the greylisting component is implemented and working in its basic functionality. The spam detection engine is completed in its basic functionality as well, and we are currently working on finding the optimal definition of the underlying property list. Once this step is finished, we will set up an evaluation environment (the greylisting component requires a live-stream of spam for evaluation). We expect to have first experimental evaluations in early 2006. Based on these results, we will fine tune our system, and, in parallel, prepare a scientific publication documenting our work.
Chapter 5

Conclusions and Outlook

We have summarized our activities in Phase 1 of project “Spamabwehr II” in the period June to November 2005. Since our central goal is the development of a comprehensive, highly effective antispam system we partly focussed on the investigation of several important basic aspects. The result of these investigations will guide us in several design decisions in the final phase of the development of the prototype of our new antispam system and will allow the fine tuning which is required for excellent performance. More specifically, we focussed on four areas:

- Analyzing the properties and characteristics of spam messages: In this context, we are developing a database tool which allows for statistical analysis of spam and ham data.

- Analyzing the business model underlying the spamming phenomenon: We are developing “SpamSim”, a tool for analyzing the influence of several antispam measures on the business model underlying spamming.

- Analyzing the performance of widely used antispam methods (Bayes filter, individual rules of the rule set of SpamAssassin, probably the currently most popular antispam tool): We have illustrated the dependence of the performance of a Bayes filter (Mozilla Thunderbird) on the training effort over a longer period of time (one year). The results made us aware of a potential difficulty related to data from spam traps. Moreover, we are currently performing an in-depth analysis of all individual tests included in SpamAssassin.

- Proceeding with the development of our new approaches: We have developed and published a concept for preventing outgoing spam, and we have made substantial progress in the development and implementation of our new multi-layered antispam system.

First experimental results with our prototype are currently expected in early 2006. We are right on schedule for completing this prototype (and the related publications) by the end of April 2006.
Bibliography


