Multi-Level Reputation-Based Greylisting

Andreas G.K. Janecek, Wilfried N. Gansterer, K. Ashwin Kumar
University of Vienna, Research Lab Computational Technologies and Applications
Lenaugasse 2/8, 1080 Vienna, Austria
{andreas.janecek, wilfried.gansterer}@univie.ac.at, kumar.kashwin@gmail.com

Abstract

We present the idea and implementation details of a highly effective and reliable e-mail filtering technique. At the core of the component-based architecture is a novel combination of an enhanced self-learning variant of greylisting with a reputation-based trust mechanism. These strategies provide separate feature extraction and classification components with the opportunity of utilizing the time between two delivery attempts of an e-mail message.

The approach presented features a very high spam blocking rate and also minimizes the workload on the client side, as no responsibility for messages classified as spam is taken. The reputation-based trust mechanism decreases the delay in the transfer process of e-mail messages from reliable senders and also reduces the number of erroneously blocked legitimate messages.

1. Introduction

The enormous volume of unsolicited bulk e-mail (spam) continues to be a driving force for research in reliable anti-spam filters. In recent years, numerous strategies were developed and employed to prevent e-mail spam, but none of them can be considered a final solution as spammers could easily find ways to circumvent them. Most of the anti-spam technology available is *re-active*, and there is an ongoing game of "cat-and-mouse" between spammers and anti-spam product developers. In this paper, we try to make progress towards the ultimate goal of finding a fool-proof anti-spam technique which is efficient, effective, robust, reliable, and unlike other methods, has persisting impact.

We have introduced the basic concept of a reliable three-component architecture for e-mail filtering in [11]. In this paper, we discuss and evaluate a new approach for the first component in this architecture, the greylisting component. *Greylisting* is a well-known strategy for blocking spam which has been discussed in [15, 17]. It is based on the fact that most spammers do not use fully developed SMTP (Simple Mail Transfer Protocol) servers for sending out messages. In particular, in many cases their software tends *not* to resend a message which is temporarily rejected by the receiving SMTP server. Such behaviour differs from the standard behaviour of an RFC (Request for Comments)-conforming SMTP server, which should resend temporarily rejected messages within a certain period of time. Consequently, e-mail sent from a standard-conforming SMTP server can be expected to be resent when greylisted, whereas most of the spam currently is not.

However, "conventional" greylisting has several flaws. In particular, it can be bypassed easily by automatically resending messages, and problems arise when server farms are involved in the sending process (cf. Section 3.2). In this paper, we focus on a new combination of a new 2-level greylisting technique with a feature selection and classification component for overcoming these flaws. An integrated reputation-based trust system capable of adaptively learning a reputation score for a sender's domain significantly reduces the overhead caused for regular e-mail traffic.

2. Related Work

According to their point of action in the e-mail transfer process, anti-spam methods can be categorized into three groups: *pre-send* methods, *post-send* methods and new protocols, which are based modifying the transfer process itself. Because of their potential to avoid most of the waste of resources caused by spam (network traffic, workload on receiving server etc.), pre-send methods [9, 3] are very important. Unfortunately, their widespread acceptance and application cannot be expected in near future.

Most of the currently used e-mail filtering techniques belong to the group of post-send methods. Amongst others, it comprises techniques such as black- and whitelisting [10] or rule-based filters. The latter block e-mail depending on a pre-determined set of rules. Sophisticated rule-based filter systems such as SpamAssassin [2, 16] require training samples of known spam and non-spam messages in order to fine-tune parameters and optimize learning.

0-7695-3102-4/08 $25.00 © 2008 IEEE
DOI 10.1109/ARES.2008.187

Bayesian classifiers [1] are another example for popular post-send anti-spam methods. However, as Bayesian classification is exclusively based on textual features, this approach can be fooled by "diluting" the spam message with enough obviously "innocent" words. Delany et al. [6, 7] have studied and assessed a case-based reasoning e-mail classification as a lazy learner method that outperforms Naive Bayesian (NB) classifiers. Lai [14] performed a comparative study of the performance of various machine learning methods in spam filtering. Chuan et al. [5] presented a novel spam filter based on an LVQ-based (Learning Vector Quantization) neural network, and Fdez-Riverola et al. [8] presented an instance-based reasoning e-mail filtering model that outperformed classical machine learning techniques and other successful lazy learner approaches in the domain of spam filtering. Bratko et al. [4] have investigated an approach based on adaptive statistical data compression models. These compression models were reported to outperform currently established spam filters.

From the point of view of an *individual* user, many filtering methods may achieve reasonably satisfactory results, *provided* they are trained, tuned and maintained permanently. This effort required for sustaining satisfactory performance is one of the disadvantages of existing filtering methods. An even more substantial drawback is the fact that most of these methods act *after* the message has been transferred and received by the recipient. Consequently, a big part of the damage caused by spam has already been done before the classification results are available. Post-send methods in general are not able to reduce the overhead, the waste of bandwidth, processing power, memory and time caused by spam.

Greylisting [13] is a popular approach to minimizing the overhead and waste of resources caused by spam. The basic idea is to (temporarily) reject an e-mail message when it is first received and to accept it only if it is resent. The efficiency of greylisting has been demonstrated empirically [15, 17]. Nevertheless, as mentioned before, in its conventional form it has important drawbacks: It introduces delays into the delivery process of e-mail, legitimate messages may potentially get lost if an SMTP server is used with a non RFC-conforming configuration, and difficulties may arise in the process of distinguishing between first and second delivery attempts of e-mail messages, especially when e-mail server farms are involved. Moreover, conventional greylisting cannot guarantee sustained success, because it is quite easy for spammers to adapt to it and to bypass it.

Related strategies (for example, *defNullSpam* [12]) send an automated response with a verification code to the origin of an incoming e-mail. This code has to be sent back manually by the sender before the e-mail is delivered. This approach requires human interaction to ensure delivery and thus complicates the handling of legitimate mass e-mail.

3. Basics of Greylisting

In this section, the basic concept of greylisting is reviewed. It is based on properties of the Simple Mail Transfer Protocol (SMTP).

3.1. Simple Mail Transfer Protocol

SMTP, as defined in the RFC 2821 (`http://rfc.net`) is the de-facto standard protocol for e-mail transfer. It is independent of the particular transmission subsystem, requiring only a reliable ordered data stream channel. In the following, a simple SMTP dialogue between a sending (*sen*) and a receiving (*rec*) SMTP server is shown.

```
01. rec: 220 serverHost (JAMES SMTP Server)
02. sen: HELO someClient.com
03. rec: 250 Hello someClient.com
04. sen: MAIL FROM:<ashwin@someClient.com>
05. rec: 250 Ok
06. sen: RCPT TO:<andreas@serverHost.com>
07. rec: 250 Ok
08. sen: DATA
09. rec: 354 End data with <CR><LF>.<CR><LF>
10. sen: Subject: Info
11. sen: From: ashwin@someClient.com
12. sen: To: <andreas@serverHost.com>
13. sen: Hello Andreas!
14. sen: .
15. rec: 250 Ok: queued as 12345
16. sen: QUIT
17. rec: 221 Bye
```

Lines 1 and 2 establish the connection between sender and receiver. In line 3 the receiving SMTP server acknowledges the receipt of the last packet sent from the sender by returning the status code 250 (see below). In lines 4 to 7 the sender address and the recipient address are transmitted and acknowledged. In line 8 the sender initializes the sending of the message data. The receiving server responds with a 354 status code ("start mail input"). In lines 10 to 14 the message data is transmitted and again acknowledged in line 15. In lines 16 and 17 the connection is closed. The status codes, which are sent from the receiving to the sending SMTP server, are defined in the RFCs 2821 and 1893.

- *250 Ok:* If the last packet received from the sending SMTP server is accepted by the receiving SMTP server, a 250 reply is sent, which indicates that the requested mail action has been valid and completed.
- *354 Start mail input:* The receiving process is ready to receive the message data.
- *451 Temporary failure:* A temporary failure message (*try back later*) is returned, indicating that the requested action was aborted. The status codes 450 or 452 indicate different types of temporary failures.
- *550 Fatal failure:* The requested action is not taken (*mailbox unavailable*). The message is rejected permanently.

Mail transfer agents (MTAs) compliant with RFCs 2821 and 3461 *resend* messages that were not accepted from the receiving SMTP server within a certain period of time, which is set by the server administrator. For example, if the receiving SMTP server returns a temporary failure status code (450-452) at line 7 in the SMTP dialogue on page 2, then RFC-compliant sending MTAs will try to resend the message, while non RFC-compliant MTAs might simply discard this response.

3.2. Conventional Greylisting

Conventional greylisting relies on the fact that spammers (currently) tend not to use RFC-compliant software to send out messages. At the moment, most spammers seem to use a simple "fire-and-forget" mechanism to send out spam which ignores temporary rejections. A schematic illustration of conventional greylisting is given in Figure 1: When a sending SMTP server initiates the process of e-mail transfer, the receiving SMTP server records a *characteristic triplet* of the message, usually consisting of (*i*) the IP address of the host attempting the mail delivery, (*ii*) the sender's address, and (*iii*) the recipient's address. The receiving server then searches for this characteristic triplet in a local database. If no existing record matches, the message is refused with a "temporary failure" response (return code "451") and the triplet is stored locally. If, on the other hand, the triplet matches an existing record, the message is accepted and delivered to the final recipient.

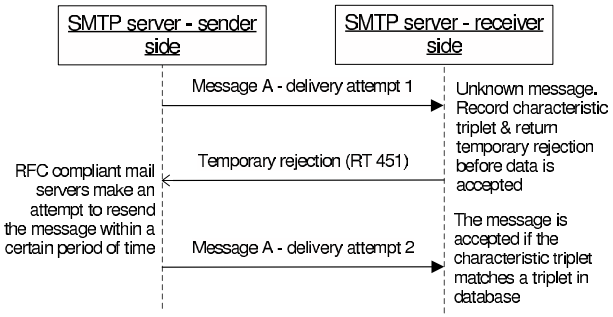

Figure 1. Conventional greylisting

Obviously, it is important to require that the message be resent within a certain time. This can be handled by introducing a minimum and a maximum allowable time after the first sending attempt and a corresponding period of validity for the characteristic triplet.

The basic idea underlying greylisting has very high potential for spam defense, especially since the time available between two delivery attempts of a temporarily rejected message can be utilized for classifying it (cf. Section 5.3). Nevertheless, as mentioned above, there are several open issues and weaknesses in the conventional greylisting process: (*i*) Greylisting introduces delays in the mail delivery process. (*ii*) What is a *reliable* way for deciding whether an SMTP session refers to a previous delivery attempt? (*iii*) Spammers can easily adapt and bypass conventional greylisting by resending messages or by sending *potentially different* messages with identical characteristic triplets. (*iv*) Large organizations often use mail-server farms for handling their outgoing e-mail traffic. In this case, attempts for resending a temporarily rejected e-mail message do not necessarily originate from the same IP address and the characteristic triplets might not match. (*v*) How should messages with multiple recipients be handled? Conventional greylisting rejects a message immediately after specification of a single recipient and treats each recipient separately.

We have developed an enhanced variant of greylisting which addresses these issues. In Section 4, we review the system architecture presented in [11]. In Section 5, the new greylisting component is introduced and discussed. Moreover, a *reputation-based trust mechanism* is presented, which reduces the delay for trusted connections.

4. Review of System Architecture

In previous work [11] we have presented an architecture which provides a framework for combining enhanced greylisting strategies with feature extraction and classification processes. The main goal of this architecture is to reduce the waste of resources (workload, memory, etc.) caused by spam. The focus is on strategies for detecting spam as early as possible—ideally *before* the receiving mail server has assumed responsibility for delivering a message to the final recipient.

The architecture consists of three components. The first main component comprises the receiving SMTP server with integrated greylisting. This component sets the stage for the following feature extraction and the classification components. Within the feature extraction component, a set of features is determined and these features are then used for the classification of the message as spam or not spam.

In the greylisting strategy discussed in [11], a new arriving e-mail message first passes through the receiving SMTP server and is temporarily rejected *after* its data has been received. Then a *characteristic quartuplet* of the message, consisting of (*i*) IP address of the sending SMTP server, (*ii*) the sender's address, (*iii*) the recipient's address, and (*iv*) a checksum of the message body is extracted. After this, the message passes through the feature extraction and classification components. The resulting "spam value" is then stored together with the quartuplet. If the same message is received for a second time (as a reaction to the temporary rejection), the previously computed classification result is *already available*. Depending on this result, the message is permanently rejected (if it has been classified as spam) or accepted.

Important issues remain to be discussed. (*i*) How can we handle messages originating from server farms? (*ii*) How can we avoid that the system is bypassed by simply sending the same message twice? (*iii*) Is it possible to reduce the delay caused by greylisting in the transfer time for legitimate e-mail? In this paper, we address these issues by focussing on an improved greylisting component. We propose and evaluate a more robust and more efficient appproach based on a novel 2-level greylisting strategy which is combined with the classification component and with a reputation-based trust mechanism. An overview of the architecture with the new components introduced in this paper is given in Figure 2.

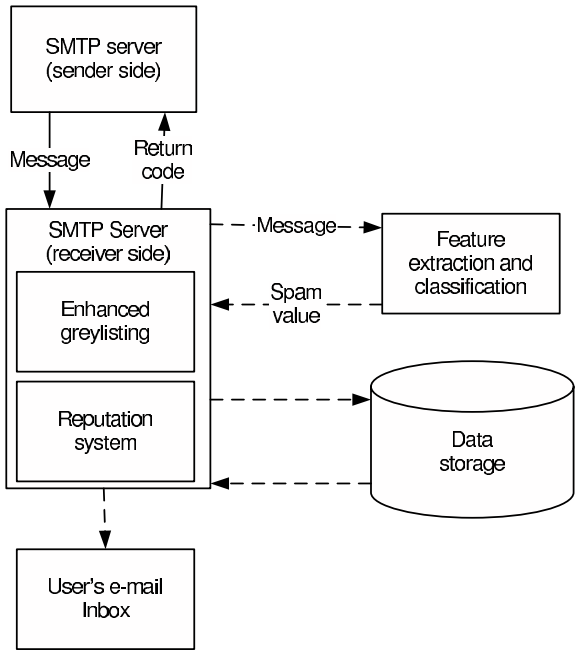

Figure 2. System architecture

5. 2-Level Greylisting with Reputation System

In this section, a *2-level* greylisting strategy (abbreviated as "2g" in the following) and its enhancement with a reputation system (abbreviated as "2g-r") are discussed.

5.1. 2-Level Greylisting

Our greylisting component comprises two levels. The purpose of the first level is to block the large amount of spam (currently) sent from non RFC-compliant MTAs (cf. Section 3.2), while the second level is mainly responsible for detecting spam which is sent through RFC-compliant MTAs based on analyzing e-mail content using filtering techniques.

All incoming e-mail messages are received by the SMTP server, which first extracts a characteristic triplet of the message. This triplet consists of (*i*) the last part of sender's domain name, (*ii*) the sender's address, and (*iii*) the recipient's address.

Note that this triplet differs slightly from the triplet used in the conventional variant of greylisting in order to be able to deal with e-mail sent out from mail-server farms. We do not store the individual host attempting the delivery, but the last part of the sender's domain name (for example, "`univie.ac.at`"). Only if no DNS entry can be found for a specific IP address, then the IP address of the host attempting the mail delivery becomes part of the triplet.

As the next step, this triplet is searched in a local data base. If it is not found, then the message is sent to the first greylisting level L_1. If the triplet matches an already existing entry in the data base, the message is passed on to the second greylisting level L_2. The second level is responsible for handling various exceptions, such as messages with the same characteristic triplet but different content, or messages, which even have the same characteristic triplet *and* the same content, but still originate from distinct sending attempts (i. e., messages which were sent twice, but *not* as a reaction to a temporary rejection—for example, by spammers who try to circumvent conventional greylisting.

First Level (L_1). Any newly arriving e-mail reaches the first level of greylisting. At this level, the message is temporarily rejected *before* the DATA command (at line 7 in Code 1 in Section 3.1). At this stage, the characteristic triplet as defined above is stored locally in a data base together with a *level counter* indicating the level of the greylisting process (set to 1 at this stage), a record create time (the time when the message has been received, i.e., local time of the receiving SMTP server) and an expiration time (indicating how long this information should be saved).

Second Level (L_2). If the characteristic triplet of an arriving message matches an existing entry in the data base *and* the message has been resent within the validity period (see Section 3.2), then it arrives at the second level of greylisting. Here, it is temporarily rejected again, but this time *after* the DATA command (at line 17 in Code 1 in Section 3.1).

At this stage in the SMTP protocol (after receiving the data of the message) the "mail sent time" information, which is stored in the e-mail header, is available. The "mail sent time" indicates the time at which the *receiving* SMTP server has received (but not necessarily accepted) the message for the first time (in our case this is basically the time at which this message was temporarily rejected). This "mail sent time" can now be compared to the record create time stored together with the characteristic triplet in the data base. Consequently, based on this information it becomes possible to distinguish between two *different* messages sent from the same domain name and from the same sender address to the same recipient(s), because their "mail sent times" differ.

If the "mail sent time" and the record create time do not match, then this message has to be considered a completely new message (even if the characteristic triplet already exists). It is handled in the same way as a newly arriving message from an unknown sender, i. e., it is temporarily rejected and its level counter is set to 1.

If "mail sent time" matches the record create time for this characteristic triplet stored in the data base, then we have a second delivery attempt of a previously seen message. In this case the level counter of the entry in the greylist data base is set to 2 and a checksum is computed over the e-mail content. Then the checksum and the message content are both stored in the data base.

The time between the second and the third delivery attempt can now be used for an in-depth examination of the message content using suitable filtering techniques (see, for example, [10]). In our architecture, this is done by the classification daemon (see Section 5.3) which classifies each message as spam or legitimate e-mail ("ham"). After the classification the data base entry for this message is updated with the result of the classification process.

If the same message is received a third time (within the validity period) the classification result is *already available*. To ensure that this message is identical to the one which has been classified, the checksum of the message received is computed and compared to the checksum stored in the greylist entry. If the checksums match, the message is either rejected permanently or accepted.

This 2-level greylisting is a highly effective way for detecting and blocking spam. However, it introduces a potentially significant delay (depending on the configuration of the sending SMTP server) into the delivery process of regular e-mail. Consequently, we combine it with a newly developed reputation-based trust mechanism to reduce the transfer delay for messages sent from *trustworthy* senders as discussed in the following.

5.2. Reputation System

A message which has to go through both levels of greylisting has to be sent *three* times in total before it reaches the recipient. This may introduce significant delays in the transfer process of legitimate e-mail. Consequently, we developed a reputation-based trust mechanism which minizimes the transfer delay for trusted senders. Integration of this mechanism into the 2-level greylisting outlined in Section 5.1 leads to the *reputation-enhanced 2-level greylisting* (abbreviated as "2g-r") which is discussed here.

The basic idea is a variant of dynamic black- and whitelisting based on a *reputation score* for each sender, which is adapted dynamically to the history observed. In general, it is not possible to verify the sender's address in a standard SMTP dialogue. The IP-address of the MTA cannot be forged, though, and thus the reputation score is actually assigned to the *domain* of the sender. Based on the reputation score the system decides whether a message is blocked or accepted without any investigation, or whether it should go through one or two levels of greylisting. The reputation score depends on the history of the sending domain and also on the currentness of the last interaction with this sending domain, as the following pseudo-code shows.

if *sender's domain NOT in reputation list* **then**
    message has to go through L_1 and L_2
    assign reputation score after classification
else
    update reputation score based on the time passed
    since previous message from this domain
    if *maximum reputation* **then**
        deliver message directly to inbox
    if *medium reputation* **then**
        message has to go through L_2
        update reputation score after classification
    if *minimum reputation* **then**
        message has to go through L_1 and L_2
        update reputation score after classification
end

It suffices to distinguish three degrees of reputation which can be assigned to a sending domain.

Maximum Reputation. Sending domains who recently have sent only legitimate e-mail can achieve *maximum reputation*. Messages received from such domains automatically bypass the greylisting process and are directly delivered to the recipient's inbox.

Medium Reputation. Domains who predominantly sent legitimate e-mail, in particular in the recent past, or domains who earlier had sent only legitimate e-mail but no messages recently may find themselves at *medium reputation*. A message received from such domains automatically bypasses the first level of greylisting (L_1), but has to go through the second level (L_2), where it is temporarily rejected after the data has been transmitted. Thus, the message content is classified by the classification modul (see Section 5.3). When such a message is received for a second time a classification result is already available, and based on this result the message is either permanently rejected or accepted. Moreover, if the message is classified as ham, the reputation score of the domain is increased. If the message is classified as spam, the reputation score of the domain is decreased.

Minimum Reputation. Domains which did not send legitimate e-mail for a longer period of time, or domains which recently sent spam have *minimum reputation*. These messages are treated the same way as messages from domains without any reputation value and have to go through both levels of greylisting. The reputation score is then adjusted depending on the classification result.

5.3. Feature Extraction and Classification

In the architecture reviewed in Section 4, the feature extraction and classification components perform the classification process. The classification modul is an independent, permanently running process which reads a message from the data base, classifies it as spam or ham, and updates the corresponding data base entry with the computed result.

This process can be implemented as a daemon which checks the entries of the greylist data base regularly. Entries with a level counter 2 for which the spam value has not been computed yet are to be classified. In the prototype we developed (see Section 6.1), the SpamAssassin system [2] was used to extract certain features from each message and to classify the message as spam or ham. The classification process establishes a connection to the SpamAssassin system and hands over the message content. The SpamAssassin system classifies the message and returns the result to the classification process, which writes the corresponding spam value in the data base.

5.4. Advantages of System Architecture

The 2-level greylisting mechanism proposed here improves conventional greylisting in two central aspects (see also Section 6). First, it becomes possible to properly handle legitimate e-mail messages sent out by server farms (which are often used by large organizations for outgoing e-mail). In such cases, the IP address associated with a reaction to temporary rejection can differ from the IP address associated with the original delivery attempt, and usually the IP ranges of such server farms are not disclosed by these organizations. In our "2g"-methodology most of these cases can be dealt with by replacing the IP address with the basic part of the domain name and using it as a part of the characteristic triplet. As an example, consider the case that the first delivery attempt of an e-mail message from gmail (`www.gmail.com`) comes from the IP address which translates into `wz-out.google.com`, but the second delivery attempt after temporary rejection comes from the IP address associated with `nz-out.google.com`. Since the IP addresses and the host names differ, e-mail from gmail

might be greylisted forever. Our system solves the problem by storing `google.com` as part of characteristic triplet.

Second, conventional greylisting can be bypassed by sending two messages with identical characteristic triplets. Our approach takes into account the sent-time of an e-mail which is written into its header at the initial delivery attempt. Comparing this information with the e-mail sent-time stored locally in the greylist data base with the characteristic triplet allows for distinguishing repeated delivery attempts from independent sending processes.

By integrating the reputation mechanism, several other central drawbacks of conventional greylisting can be addressed. (*i*) The reputation-based greylisting automatically adapts to changing behavior of sending domains and is consequently much more flexible than standard white- and blacklisting techniques. (*ii*) Our approach reliably reduces the transfer delay caused by conventional greylisting for legitimate e-mail without significantly reducing the detection and blocking rates. (*iii*) The reputation mechanism also reduces the load on the components of the architecture and thus reduces the vulnerability to denial of service (DoS) attacks. In particular, sending domains with maximum reputation do not have to go through the potentially time-consuming feature extraction and classification components. This way, they are much less affected by DoS attacks coming from domains which are unknown or have only low reputation.

6. Experimental Evaluation

In this section, three versions of greylisting are experimentally evaluated and compared: (*i*) conventional greylisting, (*ii*) 2-level greylisting as presented in this paper, and (*iii*) reputation-enhanced 2-level greylisting. These three methods are evaluated in terms of the spam blocking rate as well as in terms of the transfer delay of legitimate e-mail messages.

6.1. Setup and Test Data

In our experiments, the Apache JAMES server (`http://james.apache.org`) was used as SMTP server. The characteristic information of arriving e-mail messages and the data for the reputation system was stored in a MySQL data base. The SpamAssassin system [2] was used for feature extraction and classification after the second level of greylisting.

The system was tested using live streams of e-mail traffic from several spam traps (honey pots). A spam trap is an e-mail address that is not used and thus unknown to other humans, but detectable by automatic e-mail harvesting tools used by spammers. As a consequence, any e-mail sent to such a spam trap address has to be spam. On average, we received five spam messages per second.

For both greylisting levels the required minimum time after a delivery attempt was set to $T_{\min} = 60$ [s], i. e., any repeated delivery attempt within $T_{\min}$ after the previous one was temporarily rejected again without advancing the greylisting level. The expiration time of a delivery attempt was set to $T_{\max} = 4$ [h], i. e., if a temporarily rejected message was not received again within this time after the previous temporary rejection, the greylist entry for this message was removed from the data base.

6.2. Blocking Rates

In this section, we summarize the experimentally observed blocking rates, i. e., the percentage of the spam messages which did not reach the final recipient due to the greylisting and classification mechanisms discussed here.

Overall Results. The failure rates of conventional greylisting as well as for 2-level greylisting observed for four different periods are illustrated in Table 1 using the following notation: T denotes the total number of *different* spam messages which arrived at our SMTP server (messages received more than once were only counted once). "cg" denotes the total number (percentage) of spam messages which have successfully passed an implementation of conventional greylisting. "2g" denotes the total number (percentage) of spam messages which successfully passed our implementation of the 2-level greylisting and the classification component.

Table 1. Different variants of greylisting

Period	Date	T	cg	2g
1	24/03/07	128 763	568 (0.44%)	112 (0.09%)
2	06/04/07	142 905	260 (0.18%)	36 (0.03%)
3	09/10/07	149 176	601 (0.40%)	66 (0.04%)
4	10/10/07	123 639	414 (0.33%)	16 (0.01%)

It is observed that 2-level greylisting significantly reduces the number of spam messages which were not blocked by the system ("false negatives") compared to conventional greylisting (by factors of five and more). In the following, we look at more detailed statistics for the levels L_1 and L_2 as well as for the classification component.

First Level. The blocking rates of Level L_1 of the 2-level greylisting are shown in Table 2 using the following notation.

- NR_1 denotes the total number of spam messages which were temporarily rejected at L_1 and did *not* return.
- RO_1 denotes the total number of spam messages which were temporarily rejected at L_1 and did return a second time, but outside the period of validity.
- RV_1WT denotes the total number of triplets which were temporarily rejected at L_1 and appeared again within the period of validity, but with a wrong "mail sent time" (cf. Section 5.1).
- RV_1SUC denotes the total number of spam messages which were temporarily rejected at L_1, returned a second time within the period of validity, and also had the correct "mail sent time". These messages are the only ones which successfully pass L_1.

Overall, $NR_1 + RO_1 + RV_1WT + RV_1SUC = T$. The differences between the results for the Level L_1 in our 2g-method and the results for conventional greylisting shown in Table 1 are due to the checking of the "mail sent times" as discussed in Section 5.1.

Again, we observe that the overwhelming majority of spam messages currently does not return after a temporary rejection. The number of messages which returned outside the period of validity is consistently relatively low, which indicates that our choice

Table 2. Statistics for greylisting level L_1

Period	T	NR_1	RO_1	RV_1WT	RV_1SUC
1	128 763	128 123	72	422	146
2	142 905	142 501	144	173	87
3	149 176	148 478	97	465	136
4	123 639	123 126	99	379	35

for this parameter is acceptable. The values for RV_1WT are certainly non-negligible. This indicates that a certain fraction of spam is sent out (at least) twice in order to bypass conventional greylisting. In the future, we may see an increase of this value as spammers adapt to conventional greylisting. Only RV_1SUC messages successfully pass the first level of greylisting. Their content is read and they are temporarily rejected for a second time. Meanwhile, their content is analyzed by the SpamAssassin system.

Second Level. In Table 3 we summarize statistics about Level L_2 using the following notation:

- T_2 is the number of messages which successfully passed L_1 and arrived at L_2 ($T_2 = RV_1SUC$).
- NR_2 denotes the total number of spam messages which were temporarily rejected at L_2 and did not return.
- RO_2 denotes the total number of spam messages which were temporarily rejected at L_2 and did return a third time, but outside the period of validity (the same one as for L_1).
- RV_2WT denotes the total number of triplets which were temporarily rejected at L_2 and appeared again within the period of validity, but with a wrong "mail sent time".
- RV_2SUC denotes the total number of spam messages which were temporarily rejected at L_2, returned a third time within the period of validity, and also had the correct "mail sent time". These messages have successfully passed the second level of greylisting.

Table 3. Statistics for greylisting level L_2

Period	T_2	NR_2	RO_2	RV_2WT	RV_2SUC
1	146	5	0	0	141
2	87	9	0	0	78
3	136	18	2	0	118
4	35	2	0	0	33

The values NR_2 are low, but greater than zero. Consequently, it happens—although not very often—that a spam message is resent exactly *once*. As expected, the values for RO_2 are very low (a message which has been resent within the validity period before and is resent a second time is very likely to be resent within the same period again). It is no surprise that no message with an incorrectly "mail sent time" has been reveiced at this level, because those messages have been sorted out at L_1. A large majority of the messages arriving at L_2 successfully passed the second level of greylisting. Based on the classification results available at the time when they return a third time, they will be either permanently rejected (if classified as spam) or accepted.

Classification Component. The classification results for those RV_2SUC messages which passed L_2 are shown in Table 4. S denotes the number of messages which were classified as spam by the SpamAssassin system, and H indicates how many messages were classified as ham. The last column in Table 4 shows the overall percentages of spam messages which were blocked by the combination of the 2-level greylisting with SpamAssassin. These values are very high and indicate the excellent blocking performance of our method.

Table 4. Classification component

Period	RV_2SUC	S	H	**blocked tot.**
1	141	29 (21%)	112 (79%)	**99.91** %
2	78	42 (54%)	36 (46%)	**99.97** %
3	118	52 (44%)	66 (56%)	**99.96** %
4	33	17 (52%)	16 (48%)	**99.99** %

6.3. Transfer Delay

One of the central remaining drawbacks of the 2g-methodology is the increase in the transfer time of legitimate messages, because the two temporary rejections cause an even larger delivery delay than in conventional greylisting. The integration of the reputation-based trust mechanism introduced in Section 5.2 yields very good improvements in this aspect.

In order to illustrate the reductions in the transfer delays for legitimate e-mail we manually sent test messages from various domains to our prototype implementation. Table 5 summarizes the average transfer delays for legitimate messages depending on the reputation score assigned to the sending domain.

Table 5. Average transfer delays

Sending domain	Reputation score		
	low	med	high
`google.com`	25 min	6 min	0 min
`hotmail.com`	6 min	3 min	0 min
`yahoo.com`	26 min	8 min	0 min
`iiita.ac.in`	15 min	6 min	0 min

Obviously, the average delay is zero for a sending domain with high reputation since its messages bypass both levels of greylisting. The average delays observed for lower reputation scores depend on the configuration of the sending SMTP server, which determines after which time period a temporarily rejected e-mail is resent.

In a more comprehensive study, we subscribed to various newsletters in order to increase the volume of our stream of legitimate e-mail. These newsletters represented various types of domains, for example, `www.cs.wisc.edu/dbworld/`, `www.boerse-express.com/`, or `www.vyoms.com`. All reputation scores were initialized with the lowest possible value.

Figure 3 focusses on the reduction of the transfer delay for these and manually sent messages achieved by integrating the reputation-based trust mechanism into the 2g-methodology. It

shows the overall percentage of legitimate e-mail messages which experienced at most a certain transfer delay. The transfer delay in minutes varies along the x-axis. The two curves shown in Figure 3 correspond to reputation-enhanced 2-level greylisting (upper curve) and to 2-level greylisting without the reputation-based trust mechanism (lower curve).

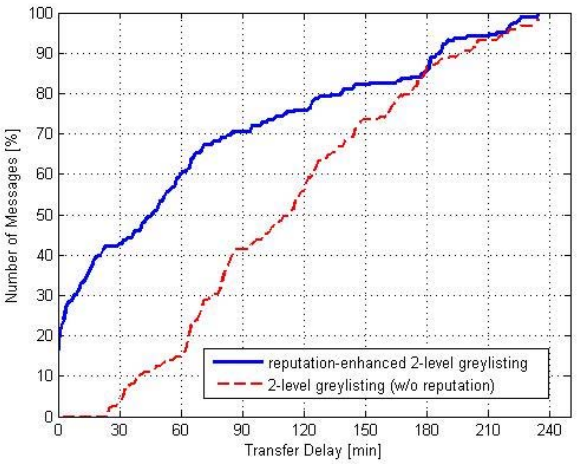

Figure 3. Transfer delays for legitimate e-mail with and without reputation system

It can be observed that the reputation system significantly reduces the transfer times. For the 2g-methodology without reputation mechanism, all messages experienced a delay of at least 23 minutes, and only about 15% of the messages were delivered within one hour. In contrast, for the 2g-r-methodology, about 20% of all messages were received without any delay, and around 60% of the messages were received within one hour.

7. Conclusion

We have presented and evaluated the concept of a 2-level greylisting technique to be used in the context of a previously developed component-based architecture for detecting and filtering unsolicited bulk or commercial e-mail ("spam"). In contrast to existing greylisting approaches, our method successfully handles messages originating from server farms and it cannot be bypassed by sending identical messages repeatedly. It has been illustrated that this technique allows for achieving very high spam blocking rates of 99.9% and higher.

A potential disadvantage of multi-level greylisting is the delayed delivery. Thus, we have presented a reputation-based trust mechanism whose integration into the 2-level greylisting technique was shown to significantly reduce the transfer delay for legitimate messages caused by the additional greylisting level.

Acknowledgement. This research was partly supported by Internet Privatstiftung Austria.

References

[1] I. Androutsopoulos, J. Koutsias, K. Chandrinos, and C. D. Spyropoulos. An experimental comparison of naive bayesian and keyword-based anti-spam filtering with personal e-mail messages. In *SIGIR '00: Proceedings of the 23rd annual international ACM SIGIR conference on research and development in information retrieval*, pages 160–167, 2000.

[2] Apache Software Foundation. SpamAssassin open-source spam filter, 2006. `http://spamassassin.apache.org/`.

[3] A. Back. Hashcash – a denial of service counter-measure, 2002. `http://www.hashcash.org`.

[4] A. Bratko, G. V. Cormack, B. Filipic, T. R. Lynam, and B. Zupan. Spam filtering using statistical data compression models. *Journal of Machine Learning Research*, 6:2673–2698, 2006.

[5] Z. Chuan, L. Xianliang, H. Mengshu, and Z. Xu. A lvq-based neural network anti-spam email approach. *SIGOPS Oper. Syst. Rev.*, 39(1):34–39, 2005.

[6] S. J. Delany, P. Cunningham, and L. Coyle. An assessment of case-based reasoning for spam filtering. *Artif. Intell. Rev.*, 24(3-4):359–378, 2005.

[7] S. J. Delany, P. Cunningham, D. Doyle, and A. Zamolotskikh. Generating estimates of classification confidence for a case-based spam filter. In *ICCBR*, pages 177–190, 2005.

[8] F. Fdez-Riverola, E. L. Iglesias, F. Daz, J. R. Mendez, and J. M. Corchado. Spamhunting: An instance-based reasoning system for spam labelling and filtering. *Decis. Support Syst.*, 43(3):722–736, 2007.

[9] W. N. Gansterer, H. Hlavacs, M. Ilger, P. Lechner, and J. Strauß. Token buckets for outgoing spam prevention. In *Proceedings of the IASTED International Conference on Communication, Network and Information Security*, pages 17–22, 2005.

[10] W. N. Gansterer, M. Ilger, P. Lechner, R. Neumayer, and J. Strauß. Anti-spam methods - state of the art. Technical Report FA384018-1, Institute of Distributed and Multimedia Systems, Faculty of Computer Science, University of Vienna, 2005.

[11] W. N. Gansterer, A. G. K. Janecek, and P. Lechner. A reliable component-based architecture for e-mail filtering. In *ARES '07: Proceedings of the The Second International Conference on Availability, Reliability and Security*, pages 43–52, Washington, DC, USA, 2007. IEEE Computer Society.

[12] A. Gardner. defnullspam. `http://www.defnullspam.com/`, 2007.

[13] E. Harris. The next step in the spam control war: Greylisting. `http://projects.puremagic.com/greylisting/whitepaper.html`, 2003.

[14] C.-C. Lai. An empirical study of three machine learning methods for spam filtering. *Know.-Based Syst.*, 20(3):249–254, 2007.

[15] J. R. Levine. Experiences with greylisting. In *Proceedings of the 2nd Conference on Email and Anti-Spam*, 2005.

[16] C. McGregor. Controlling spam with spamassassin. *Linux J.*, 2007(153):9, 2007.

[17] R. D. Twining, M. M. Williamson, M. Mowbray, and M. Rahmouni. Email prioritization: reducing delays on legitimate mail caused by junk mail. Technical report, HP Laboratories Bristol, 2004. `http://www.hpl.hp.com/techreports/2004/HPL-2004-5.pdf`.