Fighting Spam on the Sender Side: A Lightweight Approach

Wouter Willem de Vries, Giovane Cesar Moreira Moura, and Aiko Pras

University of Twente
Centre for Telematics and Information Technology
Faculty of Electrical Engineering, Mathematics and Computer Science
Design and Analysis of Communications Systems (DACS)
Enschede, The Netherlands
w.w.devries@student.utwente.nl, {g.c.m.moura,a.pras}@utwente.nl

Abstract. Spam comprises approximately 90 to 95 percent of all e-mail traffic on the Internet nowadays, and it is a problem far from being solved. The losses caused by spam are estimated to reach up to $87 billion yearly. When fighting spam, most of the proposals focus on the *receiver-side* and are usually *resource-intensive* in terms of processing requirements. In this paper we present an approach to address these shortcomings: we propose to *i*) filter outgoing e-mail on the sender side and (*ii*) use lightweight techniques to check whether a message is spam or not. Moreover, we have evaluated the accuracy of these techniques in detecting spam on the sender side with two different data sets, obtained at a Dutch hosting provider. The results obtained with this approach suggest we can significantly reduce the amount of spam on the Internet by performing simple checks at the sender-side.

1 Introduction

Spam comprises approximately 90 to 95 percent of all e-mail traffic on the Internet nowadays [1,2]. To deal with all this unsolicited e-mail, companies have to spend computer and network resources, and human labour hours, which causes economic losses. It is estimated that worldwide spam causes losses from $10 billion to $87 billion [3] yearly.

Among the reasons why there is so much spam on the Internet is the cost of sending it: it costs virtually nothing. It is estimated that the transmission of a spam message is 10,000 times cheaper than receiving it [4]. Moreover, e-mail protocols do not impose any security restrictions on *how e-mail is sent*, which turns any machine on the Internet into a potential spam server.

To fight this massive number of spam, recipients rely on *receiver-side* techniques, such as automatic filtering employed by e-mail clients or mail server filters (such as SpamAssassin). Despite being to a certain degree efficient, these techniques present a major drawback: they are very resource-intensive [5]. Due to that, very often Internet Service Providers (ISP's) are forced to be more aggressive in rejecting e-mail messages upfront if they do not comply to very

F.A. Aagesen and S.J. Knapskog (Eds.): EUNICE 2010, LNCS 6164, pp. 188–197, 2010.

rigorous requirements [6], to prevent the filtering mail servers from overloading [6]. In turn, this method may lead to false positives, causing legitimate e-mail to be rejected. As a consequence, such aggressive techniques may compromise the reliability of e-mail communication while still not being able to filter all spam.

In face of these problems associated with sending and filtering spam, in this paper we present a new approach to fight spam, that focuses on *filtering spam on the sender-side* and *is not as resource-intensive* as traditional solutions. If it is so cheap to send spam messages, and so resource intensive to detect on the receiver side, why not tackle the problem by avoiding spam from being sent instead, at the sender side? If we could significantly restrain the transmission of spam, the amount of incoming e-mails would be much lower.

In order to detect spam on the sender-side, we propose the employment of four non-resource intensive techniques in terms of CPU cycles. These techniques focus on detecting spam based on e-mail message's core attributes, such as advertised URLs. In order to evaluate the effectiveness of these techniques, we have conducted a series of tests on a production network of a large Dutch hosting provider.

The rest of this paper is organized as follows: in section 2 we introduce the background and current solutions employed by ISPs to fight spam, and the shortcomings associated with each solution. Next, in Section 3 we present our proposal for fighting spam on the sender-side using low resource-intensive techniques. After that, in Section 4 we present an evaluation of the effectiveness of these techniques. To do that, we have used data from real-life networks from a Dutch hosting provider. Finally, in Section 5 we present the conclusions and future work[1].

2 Background and Current Solutions to Fight Spam

In this section we present a study on the techniques used by spammers to conduct their campaigns. Moreover, we provide a review of related work and existing solutions employed to fight spam.

2.1 Background

Currently, three main approaches are employed by spammers to easily send vast amounts of spam at low cost while maintaining most or all anonymity: direct spamming, open mail relay, and botnets [8,9]. Direct spammers are those that use their own mail servers to send larges amounts of spam. Key characteristic of direct spammers is that they do not try to hide their activities and identity, they send large amounts of spam from a limited amount of IP addresses. Some of those people are publicly listed in the Register of Known Spam Operations (ROKSO) [10], so ISPs can recognize and reject them, should they request connectivity. Occasionally direct spammers even resort to hijacking entire IP prefixes [11].

[1] It should be noted that an initial version of this paper has been presented at the twelfth Twente Student Conference on Information Technology [7]. However, that was an internal event of the University of Twente, of which the proceedings have not officially been published by any publisher.

On the other hand, open mail relay is when mail servers are configured in a way that allows any user, authorized or not, to send e-mail through it. This allows spammers to send their spam almost anonymously, only the operator of the misconfigured e-mail server could find out about the identity of the spammer. For example, Sendmail version 5 was an open relay by default [8]. Many inexperienced administrators make this configuration mistake, so spammers actively scan the internet for open relays. E-mail server software is fortunately increasingly configured to be secure by default, so this issue is becoming less severe.

Finally, the last and more effective approach to send spam is using botnets. A botnet consists of many compromised hosts (named bots) [12], connected to a central control server maintained by a spammer. Those bots are typically Windows computers infected by a virus and owned by unaware end-users [13,14]. From the control server, the spammer commands the bots to transmit his spam campaigns. This method is rapidly advancing: it provides for a cheap, distributed and completely anonymous way to transmit spam.

In the literature, spamming host are classified according to the number of spam they generate. Pathak *et al.* [15] conducted a research on spammers' behavior by setting up an open relay and generating statistics on the spam they collected in a period of three months. They have observed the prevalence of two sets of spamming hosts: high-volume spammers (HVS) and low-volume spammers (LVS). The LVS is a set containing a high number of hosts, each sending a low volume of spam (like bots in a botnet). In contrast, The HVS is a set containing a low number of hosts, each sending a high volume of spam (like direct spammers).

2.2 Solutions Employed by ISPs to Fight Spam

ISPs are increasingly interested in preventing spam transmissions from their network, mainly due to technical and legal reasons. For instance, bad managed networks eventually results in a bad network reputation. By this ISPs are risking entire IP ranges being blacklisted, making it impossible for them to transmit the legitimate e-mail of their clients. In addition, IP blacklisting as a defensive method is expected to be much less efficient (much higher chance on false positives) in the next few years for ISPs on the receiver's side due to the rise of IPv6 [16]. In regards to legal reasons, several governments implemented legislation that requires the possibility to detect, restrain and penalize spammers. Examples include the American CAN-SPAM act of 2003 [17] and the Australian SPAM ACT 2003 [18].

In practice, the following solutions are used by ISPs to fight spam:

- Blocking SMTP port for end-users;
- Rate limiting (e.g., a maximum number of e-mail messages per hour);
- Filtering out e-mail using tools like SpamAssassin.

The simplest and most radical approach to fight spam is to block the SMTP port for end-users, and allowing connections only to the mail servers of the ISP.

Since SMTP does not impose security restrictions when sending e-mail messages, any computer on the Internet can be used as a mail server [19]. Blocking SMTP port 25 for connections outside the ISP's network block spams to other mail servers, but do not prevent spam from reaching the ISP's own mail severs. The drawback of this approach is that it severely limits the end-user in using different mail servers for different purposes.

Another approach consists in limiting the amount of allowed e-mails per hour that a client can send and the maximum number of recipients. It is a very basic measure and not effective against LVS, since each LVS only sends small amounts of e-mail over a long period of time [20]. In addition, legitimate e-mail activities (such as a newsletter) would also trigger this kind of measure.

The most used approach by ISPs is to filter incoming e-mail using mail filters like SpamAssassin. SpamAssassin performs a series of tests on the header and content of the messages to classify whether they are not spam [21]. It employs Bayesian filtering, DNS blacklist checks, among other techniques. The main disadvantage of this approach is that it is resource-intensive in terms of CPU cycles [22]. In addition, it is usually used on the receiver side, so it is not employed to filter out outgoing messages.

In summary, the main disadvantages of existing solutions are (*i*) being obtrusive to the end-user, (*ii*) being ineffective to modern spam methods, and (*iii*) being expensive. To fill this gap, in this work we propose a lightweight approach to filter out outgoing e-mail.

3 Proposed Solution

After identifying the shortcomings of existing solutions, we now present our proposal, that is based on two pillars: (*i*) on filtering outgoing e-mail on the sender side and (*ii*) using lightweight techniques to check whether a message is spam or not. The main advantage is that we can detect and block spam before it reaches the final destination at a low cost.

Figure 1 presents the architecture of our solution. In this figure, an ISP has a certain number of mail servers that are responsible for handling the e-mail traffic. When they receive the e-mail from their clients, instead of sending the e-mail directly to the other mail servers, they forward it to another host in the same network (the smarthost in the figure). This smarthost is responsible for classifying the outgoing message as spam or not, and filter out the spam ones.

To perform this e-mail message classification, we propose the use of lightweight techniques. The idea is to have very quick checks on each message that still allow good filtering results. To do that, we propose to filter e-mail based on the message's core attributes, since its requirements in terms of CPU cycles are low. Core attributes are described as elements of an e-mail which can not be easily modified or varied, such as advertised URL's. These elements can be used to develop decisive criteria, while ensuring the criteria are light on server resources [22]. For example, checking if an advertised URL is used in spam campaigns can be performed with a DNS lookup, which requires virtually no server processing power.

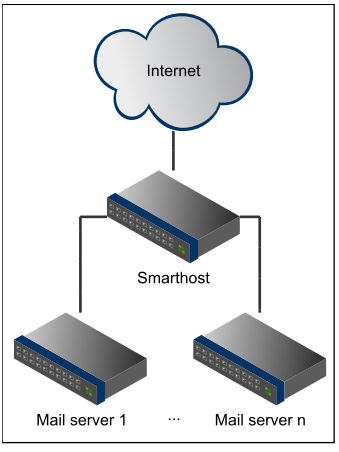

Fig. 1. Overview of new architecture

In this paper we propose the employment of the following techniques to classify a e-mail message:

- **Sender Policy Framework check (SPF):** An ISP's mail server should only transmit e-mail from domains which are actually hosted by the ISP. If an e-mail message is supposed to be sent through ISP mail servers but it does not come from a domain hosted by the ISP, it should be automatically discarded. To proceed with this verification, the filter must only check a DNS text file that contains all domains hosted by the ISP.
- **URL blacklist check:** The smarthost machine should check the URLs in the e-mail messages against several real-time blacklists. These blacklists contains URLs that are used by spammers in their campaigns. Example of these blacklists include: URIBL, SURBL, Day Old Bread, Spamhaus, Outblaze, and AbuseButler. The main advantage of this is that with one DNS lookup the filter can determine if an advertised URL is used for spam or not.
- **Google Safe Browsing (GSB)** [23]: In addition to the URL lists provided by the blacklists, the filter should check the URLs against Google Safe Browsing service. This service allows us to check the advertised URLs in real time with Google's own list of malicious hosts.
- **Full `from-address` check:** For each message, the filter should check the `from-address` field, to verify whether it exists or not. This can be quickly performed by connecting to the mail server listed in the MX record of the domain name and issuing a SMTP's RCPT TO command. To this command, the mail server will reply with a SMTP 250 or 550 code [19], which reveals whether the address exist. However, since this so-called sender callout can be quite abusive [24], we advise to only perform this check if the previous two checks suggest the e-mail is ham. This way the check will only be performed on servers owned by the user of this filter. Messages containing non-valid addresses should be discarded.

These four techniques employed by our smarthost are cheap in terms of CPU requirements, especially in comparison with the bayesian techniques. The idea is to use them all in combination to obtain a better classification, thus allowing us to have an scalable mail filter. In the next section we present an evaluation of the effectiveness of the presented techniques.

4 Evaluation the Accuracy of the Filtering Techniques

To gain insight on the effectiveness of the proposed filter, we performed analysis within the infrastructure of a large Dutch hosting provider. Due to privacy concerns, we were unable to test a proof-of-concept implementation because that would mean to inspect user's outgoing e-mail (which requires pre-authorization). To overcome this, we have conducted two different tests on real-life e-mail servers that could also give us the effectiveness of the proposed techniques.

The tests were conducted on two different data sets:

- **Dataset A – SpamAssassin log files:** These logs were obtained by analyzing the incoming e-mail to the mail servers. We have aggregated one week of log files.
- **Dataset B - E-mail Feedback Reports received from America Online:** America Online offers a service to ISP's which enables ISP's to gain insight in the spam which is being sent from their network [25]. For each e-mail which is reported as spam by an America Online user, America Online sends an E-mail Feedback Report to the originating ISP; these reports contain the headers and content of the reported spam e-mail. We aggregated these reports for one month, which means we have a set of outgoing spam e-mail to analyze. The set contains 324 EFR reports; however, it should be noted that this number of reports constitutes a far larger number of actual spam e-mails.

In the next subsections we present the results obtained for each technique.

4.1 On the Accuracy of SPF to Detect Spam

The Sender Policy Framework (SPF) check has been performed on the AOL EFR data set. For each received report, we took the domain part of the `from-address` of the spam e-mail, and we checked if this domain contained a SPF record pointing to a mail server of the hosting provider. If the domain had no SPF record or the SPF record allowed any mail server to send e-mail, we used the proposed fallback mechanism; we resolved the A record of *www.domain.tld*, and checked if this IP address was hosted in the network of the hosting provider.

Table 1 present the results for the accuracy of the Sender Policy Framework approach. As one can see, for the reporting period we received 324 EFR reports for AOL. Out of these, 125 could have been avoided if SPF was employed by the e-mail servers. This allowed us to have an accuracy of 38.6%. However, we

Table 1. Results extracted from E-mail Feedback Reports received in December 2009

AOL EFR Reports	SPF hit	`from-address` hit	GSB hit
324	125 (38.6%)	324 (100%)	165 (50.9%)

should stress that it does not mean that the approach is not good. In this case it means that 38.6% of spam was sent from non-authorized domains.

To achieve even better results, this technique should be combined with all four presented techniques in this paper.

4.2 On the Accuracy of Blacklists to Detect Spam

To evaluate the accuracy of the URL blacklist check, we have used the Dataset A (SpamAssassin log files for incoming e-mail). With this log, we could obtain the accuracy of employing blacklist for detecting spam simply by comparing the total number of incoming spam (obtained used several techniques by SpamAssassin) and the number of e-mails that were in fact classified as spam only because of the real-time blacklist. Even though we have conduced this test on *incoming e-mail*, we can expect similar results for *outgoing e-mail*, since this method is independent from source/destination.

The results obtained for the accuracy of blacklists for spam classification can be observed in Table 2. For example, on January 5th SpamAssassin has classified 7016 e-mails messages as spam. Out of this total, 51664 were correctly classified as spam using only one technique: the real-time blacklists. This gives an accuracy of 73.6% when detecting spam using only this technique.

Analyzing the results over the 8 consecutive days we have collected the data, we could observe that 36053 e-mail messages were classified as spam only by employing real-time blacklists. With these results, the overall accuracy was 74.4%. Even though it is not as high as we expected, this number can be improved when combining the next techniques together.

Table 2. Results extracted from SpamAssassin logs of 8 days in January 2010

Day	Spam e-mails	URLBL hits	Accuracy
2010/1/2	5,149	3,703	71.9%
2010/1/3	5,493	4,291	78.1%
2010/1/4	5,132	3,991	77.8%
2010/1/5	6,601	4,934	74.7%
2010/1/6	6,600	4,953	75.0%
2010/1/7	7,125	5,110	71.7%
2010/1/8	7,016	5,164	73.6%
2010/1/9	5,331	3,907	73.3%
Total	48,447	36,053	74.4%

4.3 On the Accuracy of Checking Full `from-address` to Detect Spam

The full `from-address` check was performed on the dataset B, the AOL EFR. For each received report, we connected to the mail server listed in the MX record of the `from-address` domain name and issued a SMTP's RCPT TO command. To this command, the mail server either replied with a SMTP 250 or 550 code, which reveals whether the address existed or not. A hit of this check constitutes a non-existing `from-address`.

Table 1 present the results for the accuracy of `from-address` approach. As one can observe, all the e-mail messages (324) reported by AOL contained fake e-mail addresses. This result was better than expected. We could guess that these addresses could have been randomly generated by bots while conducting their spam campaigns. These results suggest that maybe this technique might be enough to process EFR reports from AOL, for example.

4.4 On the Accuracy of the GSB Blacklist to Detect Spam

In order to access the accuracy of the GSB to detecting spam we employed the dataset B. For each received report, we checked each advertised URL against GSB. Due to the manual processing of the reports, and the time between the actual spamming activities and arrival of EFRs, there was a delay of approximately 24-48 hours between the transmission of the spam and our manual processing of the EFRs. Even though, we observed that the accuracy of using GSB we could classify 50.9% of the messages as spam, as can be observed in Table 1.

4.5 On the Overall Accuracy of Combined Techniques

In order to obtain better spam classification results, all the four mentioned techniques should be somehow combined. In this research we could not perform this, since our datasets were generated based on different sources (dataset A was obtained by analyzing outgoing e-mail while dataset B was obtained by analyzing incoming e-mail messages).

However, while deploying our techniques, all the four techniques should be employed to analyze the outgoing traffic so they can be used in combination. However, the most suitable way to combine them should be tested experimentally, which is left as future work. Despite this, we expected much better results when combining these techniques.

5 Conclusions and Future Work

In this paper we have proposed a new solution enabling ISP's and hosting providers to significantly restrain spam transmissions from their network at low cost. Our proposal is based on (*i*) filtering outgoing e-mail on the sender side and (*ii*) using four lightweight techniques to check whether a message is spam or not. The main advantage is that we can detect and block spam before it reaches

the final destination at a low cost. Moreover, this solution can be easily deployed at ISPs and hosting providers, requiring only a few modifications.

By analysis within the infrastructure of a large Dutch hosting provider, we evaluated the accuracy of the four different techniques we presented for filtering spam based on core attributes. The two best performing individual checks achieved an effectiveness of 74.4% and 100% respectively. Despite the fact that these results were obtained on relatively small data sets, they are very encouraging: for example, just by checking blacklist, we can potentially reduce more than 70% of spam generated on the sender-side. If ISPs and hosting providers start to use such approach (for several reasons, including law enforcement, good reputation, etc.), we could virtually block at the source most of spam generated nowadays. By doing that, we could reduce the spend on people, hardware and network resources to deal with spam, since most of the messages would not be forwarded to other networks.

As future work, we intend to implement and deploy a prototype to investigate real-time performance and effectiveness of the presented approach. Moreover, we intend to conduct this analysis on a much larger data set, and determine the most suitable way to combine the four detection techniques to detect spam. Next, we compare the detection rate of a setup combining the four detection techniques with the performance of a more traditional approach such as using SpamAssassin solely. We intend to achieve this by letting both setups evaluate the same set of labeled data containing spam and ham messages. Finally, we will monitor and analyze the used system resources.

References

1. Sophos. Only one in 28 emails legitimate (June 2008), http://www.sophos.com/pressoffice/news/articles/2008/07/dirtydozjul08.html
2. Spamhaus. Effective spam filtering (January 2010), http://www.spamhaus.org/effective_filtering.html
3. Soma, J., Singer, P., Hurd, J.: SPAM Still Pays: The Failure of the CAN-SPAM Act of 2003 and Proposed Legal Solutions. Harv. J. on Legis. 45, 165–619 (2008)
4. Lieb, R.: Make spammers pay before you do (July 2002), http://www.clickz.com/1432751
5. McGregor, C.: Controlling spam with SpamAssassin. Linux J. 153, 9 (2007)
6. Mori, T., Esquivel, H., Akella, A., Mao, Z.M., Xie, Y., Yu, F.: On the effectiveness of pre-acceptance spam filtering. University of Wisconsin Madison, Tech. Report TR1650 (2009)
7. de Vries, W.W.: Restraining transmission of unsolicited bulk e-mail. In: Proceedings of the twelth Twente Student Conference on Information Technology (2010)
8. Stern, H.: A survey of modern spam tools. In: Proc. of the fifth conf. on email and anti-spam (2008)
9. Sperotto, A., Vliek, G., Sadre, R., Pras, A.: Detecting Spam at the Network Level. In: Oliver, M., Sallent, S. (eds.) EUNICE 2009. LNCS, vol. 5733, pp. 208–216. Springer, Heidelberg (2009)
10. Spamhaus. The register of known spam operations (January 2010), http://www.spamhaus.org/rokso/

11. Ballani, H., Francis, P., Zhang, X.: A study of prefix hijacking and interception in the Internet. ACM SIGCOMM Computer Communication Review 37(4), 276 (2007)
12. Fabian, M.A.R.J.Z., Terzis, M.A.: My botnet is bigger than yours (maybe, better than yours): Why size estimates remain challenging. In: Proceedings of the 1st USENIX Workshop on Hot Topics in Understanding Botnets, Cambridge, USA (2007)
13. Chiang, K., Lloyd, L.: A case study of the rustock rootkit and spam bot. In: The First Workshop in Understanding Botnets (2007)
14. Mendyk-Krajewska, T., Mazur, Z.: Software Flaws as the Problem of Network Security. In: Internet-Technical Development and Applications, p. 233 (2009)
15. Pathak, A., Hu, Y.C., Mao, Z.M.: Peeking into spammer behavior from a unique vantage point. In: LEET 2008: Proceedings of the 1st Usenix Workshop on Large-Scale Exploits and Emergent Threats, pp. 1–9. USENIX Association, Berkeley (2008)
16. RIPE Labs. Spam over ipv6 (March 2010), `http://labs.ripe.net/content/spam-over-ipv6`
17. United States of America. Can-spam act of 2003 (2003), `http://uscode.house.gov/download/pls/15C103.txt`
18. Australasian Legal Information Institute. Australian spam act 2003 (2003), `http://www.austlii.edu.au/au/legis/cth/consol_act/sa200366/`
19. Klensin, J.: Simple mail transfer protocol (April 2001), `http://www.ietf.org/rfc/rfc2821.txt`
20. Pathak, A., Qian, F., Hu, Y.C., Mao, Z.M., Ranjan, S.: Botnet spam campaigns can be long lasting: evidence, implications, and analysis. In: Proceedings of the eleventh international joint conference on Measurement and modeling of computer systems, pp. 13–24. ACM, New York (2009)
21. OBrien, C., Vogel, C.: Comparing SpamAssassin with CBDF email filtering. In: Proceedings of the 7th Annual CLUK Research Colloquium (2004)
22. Pras, A., Wanrooij, W.: Filtering Spam from Bad Neighborhoods (under review). International Journal of Network Management (2009)
23. Google. Google safe browsing (January 2010), `http://code.google.com/apis/safebrowsing/`
24. UCEProtect. Sender callouts - why it is abusive (January 2010), `http://www.backscatterer.org/?target=sendercallouts`
25. AOL. E-mail feedback reports for isp's (January 2010), `http://postmaster.aol.com/cgi-bin/fbl.pl`