INTERNATIONAL JOURNAL OF NETWORK MANAGEMENT
Int. J. Network Mgmt 2010; **20:** 433–444
Published online 15 October 2010 in Wiley Online Library
(wileyonlinelibrary.com) DOI: 10.1002/nem.753

Filtering spam from bad neighborhoods

Ward van Wanrooij[1] and Aiko Pras[2,*†]

[1]*Chronowerx Holding, Arnhem, The Netherlands*
[2]*University of Twente, Enschede, The Netherlands*

SUMMARY

One of the most annoying problems on the Internet is spam. To fight spam, many approaches have been proposed over the years. Most of these approaches involve scanning the entire contents of e-mail messages in an attempt to detect suspicious keywords and patterns. Although such approaches are relatively effective, they also show some disadvantages. Therefore an interesting question is whether it would be possible to effectively detect spam without analyzing the entire contents of e-mail messages. The contribution of this paper is to present an alternative spam detection approach, which relies solely on analyzing the origin (IP address) of e-mail messages, as well as possible links within the e-mail messages to websites (URIs). Compared to analyzing suspicious keywords and patterns, detection and analysis of URIs is relatively simple. The IP addresses and URIs are compared to various kinds of blacklists; a hit increases the probability of the message being spam. Although the idea of using blacklists is well known, the novel idea proposed within this paper is to introduce the concept of 'bad neighborhoods'. To validate our approach, a prototype has been developed and tested on our university's mail server. The outcome was compared to SpamAssassin and mail server log files. The result of that comparison was that our prototype showed remarkably good detection capabilities (comparable to SpamAssassin), but puts only a small load on the mail server. Copyright © 2010 John Wiley & Sons, Ltd.

1. INTRODUCTION

According to recent studies, between 85% and 95% of all e-mail is spam [1–3]. Since manually removing spam is irritating and time consuming, it is not inconceivable that the worldwide e-mail system would collapse if there were no effective spam filters. Spam filters exist in two flavors: filters that run on the e-mail server (like SpamAssassin [4]) and filters that run on the e-mail client. To detect spam, many of these filters analyze the contents of e-mail messages in an attempt to detect suspicious keywords and patterns. Machine-learning techniques, such as Bayesian networks, are often applied to such detection [5]. Although these techniques work reasonably well, they have several disadvantages. First, since finding suspicious keyword is central processing unit (CPU) intensive, spam filtering puts a heavy load on the mail server. The University of Twente, for example, runs five mail servers in parallel to cope with the huge amount of spam. We analyzed the load on one of the (student) mail servers, and found that 64% of the CPU time was used for SPAM detection. The University of Twente would therefore be able to save three machines if there were no spam. Second, the list of keywords, patterns and rules needs to be maintained, which usually requires human effort. Third, spammers can relatively easily fool spam filters, for example by inventing an 'alternative' spelling or by putting keywords in a picture. Fourth, keywords or patterns that may characterize spam for one user may not be spam for another user. Fifth, the analysis of e-mail contents may, in some cases, raise privacy issues.

*Correspondence to: Aiko Pras, Department of Computer Science, University of Twente, PO Box 217 EWI/DACS, Enschede 7500AE Netherlands.
†E-mail: a.pras@utwente.nl

Received 31 July 2009
Revised 5 May 2010
Accepted 9 May 2010

Given these disadvantages, the question to be answered in this paper is *whether it is possible to develop a spam filter that does not need to analyze the contents of entire e-mail messages, but still rivals conventional state of the art filters in performance*.

In the context of this question, we define the contents of e-mail messages to be all elements that can be *varied* indefinitely and therefore forged (including headers, HTML code and the From address field). Elements that can be varied indefinitely are hard to detect by spam filters, which means that spam filters that analyze such elements are relatively resource (CPU, memory, I/O) hungry. Instead of analyzing elements that can be *varied*, our spam filter should only use *core* attributes that cannot easily be modified, such as the source of the message (IP address) and, in the case of phishing, the hyperlink (uniform resource identifier, URI) of the malicious website the e-mail is referring to. Since hyperlinks must be machine readable (users will click on such links, instead of retyping them), detection of such links is relatively easy.

Since several approaches to filter e-mail based on IP addresses already exist, Section 2 starts with an overview of related work. That section also discusses which and how existing techniques have been applied in our research. The concept of bad neighborhoods, which is the novel idea introduced by this paper, is presented in Section 3. How the core attributes (IP address and URI) are used in our spam filtering algorithm is the subject of Section 4. Section 5 shortly discusses our implementation, and Section 6 provides details on our evaluation. The performance of our approach is evaluated based on the following three metrics:

1. False positives (e-mails falsely classified as spam)
2. False negatives (e-mails falsely not classified as spam)
3. Algorithmic complexity (qualitatively)

Finally, Section 7 provides the conclusions and discusses further work.

2. RELATED WORK

Several research papers already exist that discuss the idea of filtering spam by analyzing the *core* attributes of e-mail messages. The motivation to perform such research ranges from avoiding the privacy issues that are related to traditional (content-based) detection [6] to improving the performance and scalability of mail servers [7]. Nearly all papers consider as core attributes the IP address of the e-mail source, since addresses of IP systems cannot easily be modified. It should be noted, however, that in extreme cases even IP addresses might be compromised, for example by using BGP hijacking. In addition, spammers can run their campaigns via botnets, and in this way use thousands of IP addresses. Some researchers therefore go one step further, and identify spammers based on 'spamming behavior' instead of IP addresses. Such behavior seems to remain relatively constant and can be used to create 'behavioral blacklists' [8].

Spam can either be analyzed at the *e-mail source*, within the *network*, or at the *e-mail destination*. Analyzing e-mail at the source may be impossible, unless the spammer does not run its own mail server, but instead relies on other systems, like open mail relays. For research purposes it is therefore possible to set up an open relay and, if this relay is used by spammers, capture traffic and analyze spammers' behavior [9]. Open mail relays are becoming rare, however, and nowadays spammers prefer to use botnets. In Kreibich *et al*. [10] the authors therefore describe an experiment in which they became part of a botnet, and in this way were able to analyze a spam campaign conducted via that botnet. They found that, for example, a single spam campaign could be targeted towards 400 million e-mail addresses over a 3-week period. It is interesting to observe that also the opposite approach has been followed: spam has been analyzed in an attempt to detect botnets [11].

Within the *network* spam can be detected by analyzing flows [12]. In Schatzmann *et al*. [7], for example, the authors found that SMTP connections may either fail (because the TCP connection is not successful), be rejected by the mail server (because the sender is blacklisted, greylisted, or the recipient

DOI: 10.1002/nem

is unknown), or accepted. Failed, rejected and accepted SMTP connections show differences in flow size and, by monitoring SMTP flows, the network operator can infer the decision made at the mail server. Flow information can be combined in an attempt to find spammers since the operator oversees all SMTP flows. Besides flow size or packet count, spammers can also be identified by analyzing the total number of packets per flow or the average number of bytes per packet [13]. Similar research is described [14] in which tcpdump was used to capture TCP flows near the mail servers. Usage of tcpdump on high-speed links may be problematic, however, which means that it may be better to rely on Netflow data. Using Netflow data only, it is possible to detect spamming machines with a 92% accuracy and a low probability of reporting false positives [15,16].

At the *e-mail destination* spam is often detected by using a combination of blacklists and keyword/pattern analysis. The blacklists are maintained by several organizations, such as the Spamhaus Project [17] and IronPort Systems [18]; a comparison of various blacklists can be found in Makey [19]. Since most blacklists can be consulted using DNS-like queries, the term DNS blacklist (DNSBL) is used to denote such lists. DNSBLs have been a popular subject of research. In Jung and Sit [20], for example, packet traces from 2000 and 2004 were analyzed. The authors found that in 2004 DNS lookups generated by mail hosts utilizing blacklists accounted for 14% of all DNS lookups. The authors also found that 80% of of the sources they identified as spammers are listed in one or more blacklists, and that the likelihood that a certain source is indeed a spammer increases when that source is included in multiple DNS blacklists. The behaviour of spammers is constantly changing [21] and spammers are nowadays increasingly relying on botnets. Unfortunately, current DNSBLs cannot easily be used to detect botnet spammers [22]. The rules for keyword and pattern analysis are tied in to specific software titles and are maintained and updated by their respective authors.

The research described in this paper focuses on using blacklists to detect spam at the e-mail destination. It builds upon the fact that the majority of received spam arrives from a few concentrated portions of the IP address space, and that spam filters can exploit these characteristics [21]. Compared to most other papers, the novel idea described in this paper is to combine different kinds of blacklists: real-time DNS blacklists, URI blacklists, policy blacklists and bad neighborhood blacklists. This last category of blacklists is used to cope with the dynamics of botnets.

3. BAD NEIGHBORHOOD CONCEPT

There are basically two kinds of spammers: high-volume spammers (HVS) and low-volume spammers (LVS) [9]. HVS send huge amounts of spam from a single mail server and thus a single IP address. Although this IP address may change over time, detection of HVS has been relatively straightforward using DNSBLs [22]. To avoid detection, LVS do not use a single mail server but instead rely on one or more botnets. Each bot in the botnet sends a relatively low number of spam messages, and additionally avoids sending multiple spam messages to the same mail server. Since a botnet may contain hundreds of thousands of machines [23], huge spam campaigns are still possible.

To give an impression of how many spam messages may be received from an individual IP addresses, Figure 1 shows the number of spam messages received by the mail servers of the University of Twente over a 7-day period in 2008. The figure shows that 90% of the nearly 300 000 IP addresses from which the University of Twente received spam sent 10 or fewer spam messages.

An important observation is that bots are not equally distributed over the Internet, but are concentrated in certain areas [21]. Our assumption is that such areas are weakly managed, and that the probability that also other systems within the same area get compromised is relatively high. In addition, we assume that systems, once infected, will first try to compromise nearby systems. This leads to bad neighborhoods, from which the probability of receiving spam is higher than from other neighborhoods. In this paper neighborhoods are defined as /24 subnets, although further research may be needed to check whether this subnet size is optimal.

Detection of bad neighborhoods can be based on DNSBLs, by counting the number of spammers already identified in that area. The higher the number, the higher the chances that that area is a

DOI: 10.1002/nem

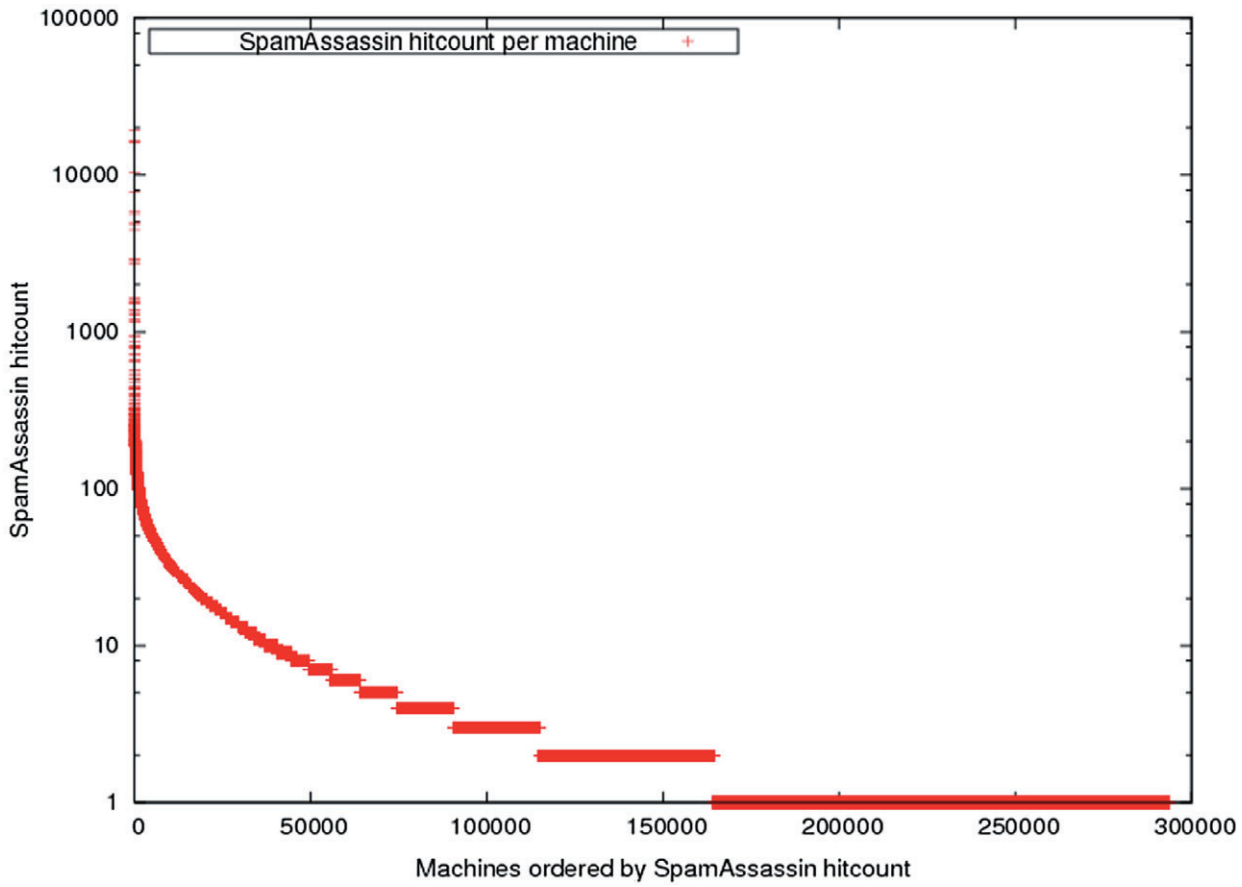

Figure 1. Number of spam e-mails per IP address (1-week period)

bad neighborhood. To make results more reliable, the outcome of multiple DNSBLs should be combined.

The fact that bad neighborhoods exist can also be presented in a visual way. Figure 2 shows which subnetworks generate most spam. This figure is created using a Hilbert space-filling curve [24,25] to plot each /24 subnet as a single pixel. Each /8 subnet is identified by its leading digits in the upper left corner of the square. The intensity of the pixel changes from clear to black to indicate the number of spam messages originating in that subnet. Network blocks that have not yet been assigned by ICANN or have been reserved (e.g., for multicasting or private networks) are filled using a special pattern. A time-lapse animation of these visualizations offers insight into the development of bad neighborhoods over time and reveals that these are surprisingly static.

Also other studies indicate that bad neighborhoods exist. In Collins *et al*. [26], for example, the term 'spatial uncleanliness' was used as an indicator for compromised hosts to cluster. In addition, the term 'temporal uncleanliness' was used to denote the likelihood that clean hosts will be infected in the future by compromised hosts from the same bad neighborhood.

4. CORE ATTRIBUTES FOR FILTERING MAIL

Two elements have been identified that can be defined as core attributes for filtering mail: the source of the mail message (IP address) and any machine-readable hyperlinks (URIs) in the mail

DOI: 10.1002/nem

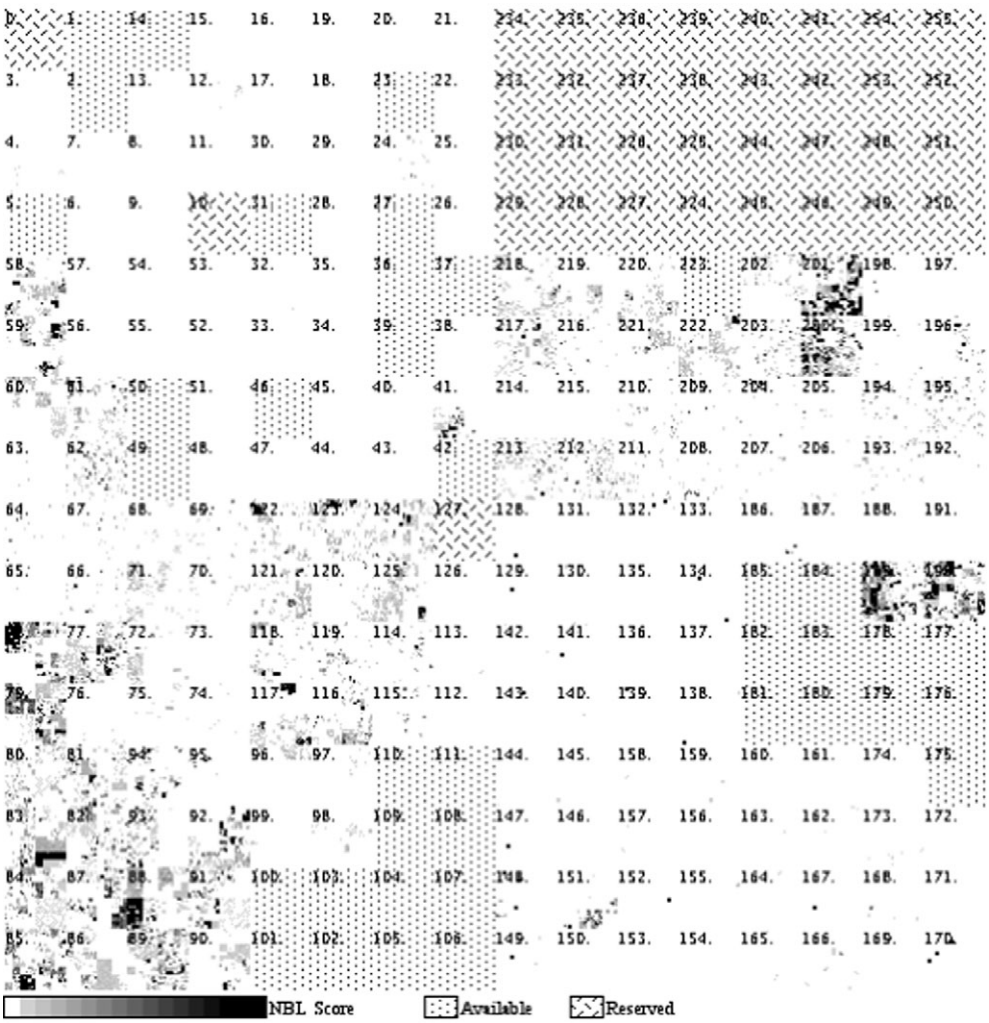

Figure 2. Visualization of bad neighborhoods (1 November 2008)

message. These attributes are checked against DNS blacklists and the scores are used to classify the mail.

4.1 IP address

The IP address that delivered the spam message to the first trusted mail server is a core attribute of a mail message. It might define the actual sender machine or a machine that was exploited for proxying the message. In either case the supply of network addresses and exploitable machines is a limited resource as it carries a cost (economic or otherwise) and this attribute is therefore a usable core attribute. The address is matched against three different collections of blacklists: 'real-time blacklist', 'bad neighborhood blacklist' and 'policy blacklist'.

4.1.1 Real-time blacklist

The collection of real-time blacklists is made up of a number of reputable DNS blacklists (such as Spamhaus' ZEN zone [17] and IronPort Systems' SpamCop blacklist [18]). The address is checked to

DOI: 10.1002/nem

establish whether previous, unremoved reports of spamming using this IP address exist. In our algorithm important criteria for these blacklists are that they should employ transparent listing and preferably automated delisting and expiration procedures.

4.1.2 Bad neighborhood blacklist

The concept of a bad neighborhood blacklist based on aggregation of real-time blacklists has been pioneered by our first prototype of February 2008. Each night copies of the real-time blacklists are downloaded and each individual listing is assigned a number of points (based on the listing being a host or network) and these points are aggregated (currently at a subnet level of /24). If, for example, only the hosts 192.0.2.5, 192.0.2.6 and 192.0.2.81 are listed, querying 192.0.2.99 against the blacklist would result in a score of three. The rationale for aggregating at 'class C' is that this can be considered the smallest building block of the Internet because it is the smallest prefix length that can be reverse delegated and smaller prefixes are not likely to be globally routable [27]. As such any listing in a /24 can be indicative of the stand of the network administrator against spam/network abuse and the use of the network (e.g., residential households, shared hosting, government agency).

4.1.3 Policy blacklist

Several DNS blacklists exist that do not list IP addresses for reports of spamming, but list addresses because network policy prohibits these from sending mail directly (direct to mail exchanger, bypassing their provider). Addresses can be submitted to such a list by the network operator or added automatically (e.g., by querying for forward/reverse DNS entries). An important selection criterion for selecting lists for use in our algorithm is that delisting should be automated for addresses that do not have any technical restriction to being a legitimate, stand-alone mailserver (e.g., a dynamic address used by an office server).

4.2 Uniform resource identifier

Any machine-readable URIs embedded in a spam message (either explicit hyperlinks or implicit URIs) is also a core attribute of that message. To prevent indefinite variations of essentially the same destination, the URI is normalized before being matched. For this purpose the URI is cropped to the domain name (e.g., http://abc@def:www.sub.example.com:8080/ghi?jkl=mno becomes www.sub.example.com) and any host names prepended to the registered domain name are removed (e.g. www.sub.example.com becomes example.com). Because the remaining domain name is a limited resource and it carries a cost, it can be used as a core attribute. This domain name is matched against a number of reputable DNS blacklists (such as SURBL [28] or URIBL [29]) that list URIs previously encountered in spam messages. A critical criterion for selecting DNS-based URI blacklists is that they employ measures against listing innocent URIs contained in spam messages.

4.3 Classification

The results for above blacklists checks need to be combined into a classification system. Because the number of scores is relatively low, at four, we have the opportunity to define thresholds for combinations of scores instead of adding all scores together to an opaque score, as is common in much pattern-based spam detection software. This also has the added advantage that the system is transparent and thus accountable; for example, this statement can be made: 'if the sending IP address is listed in N or more real-time blacklists then the mail will be blocked'. Table 1 lists the rules as a combination of thresholds for these four scores as used in our algorithm:

- rbl Number of real-time blacklist listings for the source IP address (4)
- bnbl Bad neighborhood blacklist score for the /24 subnet of the source IP address, where a host listing counts as 1 and a network listing counts as 128 (4)

DOI: 10.1002/nem

Index	rbl	bnbl	pbl	uribl
1	2	0	0	0
2	1	2	0	0
3	1	0	1	0
4	0	0	2	0
5	0	2	1	0
6	0	24	0	0
7	0	12	0	1
8	0	0	1	1
9	0	0	0	2

Table 1. Combination of thresholds

pbl Number of policy blacklist listings for the source IP address (2)
uribl Number of distinct URI blacklists that list at least one of the URIs contained in the message (3)

The thresholds have been determined during development and obviously depend on the number of blacklists used. The minimum number of blacklists for each input is contained in in parentheses in the above list.

If a message matches all thresholds of any of these rules, the algorithm considers it spam.

5. IMPLEMENTATION

An implementation for the classification algorithm in the previous section is needed in order to test its effectivity. This implementation is comprised of two elements: a centralized application that periodically computes and distributes the bad neighborhood blacklist, and an application at network edges that performs the actual scanning and classification.

5.1 Bad neighborhood blacklist generation

The blacklist is generated by combining these data sources:

1. Composite Black List (CBL) lists single IP addresses 'from very large spamtraps/mail infrastructures, and only lists IPs exhibiting characteristics which are specific to open proxies of various sorts (HTTP, socks, AnalogX, wingate etc.) and dedicated Spam BOTs which have been abused to send spam, worms/viruses that do their own direct mail transmission, or some types of trojan-horse or stealth spamware, dictionary mail harvesters etc.' [30].
2. Distributed Sender Black List (DSBL) lists single IP addresses of 'open relays and open proxies' [31]. The DSBL was disbanded in mid May 2008 and has only been used during prototype development and subsequently dropped.
3. Passive Sender Block List (PSBL) lists a single IP address: 'when it sends e-mail to a spamtrap, that e-mail is not identified as non-spam and the IP address is not a known mail server' [32].
4. Spamhaus Don't Route Or Peer List (DROP) lists 'stolen zombie netblocks and netblocks controlled entirely by professional spammers' [33].

These lists are downloaded nightly (using RSYNC or HTTP). Each individual listing in the CBL or PSBL is assigned a score of 1 and each netblock (minimum of /24) in the DROP is assigned a score of 128. These scores are aggregated at /24 and written to a file. Afterwards this file is enhanced for administrative purposes (metadata, revision number, cryptographic signature), compressed and transferred to a customized version of rbldnsd [34]. The data are then made available as a regular DNS blacklist (dnsbl.badhood.net).

DOI: 10.1002/nem

5.2 Mail scanning

Using the milter architecture for plug-ins for Sendmail and compatible mail servers we developed a simple mail-scanning plug-in (svmilter) that is capable of decoding common mail formats (MIME, Base64, Quoted-Printable, HTML encoding, URI encoding), extracting the URIs and sender IP address and classifying the mail according to our algorithm. svmilter features support for quarantining suspect messages, user-defined blocklists/thresholds and improperly constructed mail messages. The application has been used for 6 months for blocking spam at several sites prior to the validation phase of our research, to iron out any bugs and gain a solid understanding of real-world e-mail/spam traffic and the tricks used by spammers to circumvent detection.

6. EVALUATION

To answer our research question we compare an existing spam-filtering solution at a large network to our implementation of the previous paragraph. For this purpose we have selected the University of Twente, because of the diversity and magnitude of their e-mail traffic and the current use of SpamAssassin [4]. The setup compares five separate, randomly chosen test runs, each consisting of at least 1000 consecutive messages, by copying the e-mail traffic of the eight mail servers to a development server in real time. The development server captures the e-mail traffic using Wireshark [35] and writes the mail messages to disk using a custom plug-in. A separate program processes these messages and feeds them to svmilter using the milter protocol. Finally, the logs of the production mail servers are compared to the svmilter logs.

6.1 Test runs

Table 2 provides the following details of comparison of the test runs:

1. Run number, start date and total number of messages analyzed
2. Number of messages that both SpamAssassin and svmilter agree over the classification, marked as clean messages and spam messages
3. Number of messages that SpamAssassin and svmilter disagree over, marked between quotes
4. Human verification of the differences

Because of the number of e-mails processed and the privacy-invasive nature of human verification of the machine classification, this has only been performed for e-mails of which both software packages differ in classification. The following observations can be made from the the test runs:

1. Both software packages reach the same classification for over 95% of the total number of messages.
2. In all scenarios except for number 4, the false negative rate for svmilter was slightly better than SpamAssassin.

Run	1		2		3		4		5	
Date	Oct. 11 20:31:25		Oct. 12 14:21:34		Oct. 12 22:20:21		Oct. 13 13:04:45		Oct. 13 18:54:32	
#Messages	1291		1541		1301		1907		1308	
#Clean messages	126		145		211		630		140	
#Spam messages	1104		1362		1046		1201		1125	
#Differences	SA	svmilter	SA	svmilter	SA	svmilter	SA	svmilter	SA	svmilter
'clean'	36	25	21	13	30	14	25	51	18	23
of which false	36	20	21	11	30	12	24	37	18	10
'spam'	25	36	13	21	14	30	51	25	23	18
of which false	5	0	2	0	2	0	14	1	13	0

Table 2. Results of test runs

DOI: 10.1002/nem

Scenario	Univ. Twente	Development
RBL,PBL	78.46%	86.14%
BNBL	26.76%	31.01%
URIBL	92.98%	72.32%
RBL,PBL,BNBL	87.83%	96.91%
RBL,PBL,URIBL	99.10%	96.79%
BNBL,URIBL	95.68%	82.40%
RBL,PBL,BNBL,URIBL	100.00%	100.00%

Table 3. Results of scenarios

3. The svmilter software scores an exceptionally low false positive rate, both absolute and relative to SpamAssassin.[1]

Because the algorithmic complexity of svmilter is significantly lower than that of SpamAssassin, the resource usage (CPU, memory) is also lower. The test runs show an approximate increase of factor 20 in scanning throughput compared to the current SpamAssassin. Further quantification has not been performed since the gains are partly caused by implementation differences (SpamAssassin is interpreted code; svmilter is optimized for speed and memory usage).

6.2 Scenarios

Table 3 represents the cumulative results of the test runs split into seven different scenarios measuring the impact of each of the four scores for the core attributes. For comparison purposes the scores at one of the development sites are also included, because the effect of the URIBL is much more pronounced than expected and experienced during development. In particular, the novel bad neighborhood score is responsible for just 0.9% of all spam classifications at the University of Twente, compared to 3.2% at the development site. After further analysis this discrepancy can be explained by the use of pre-emptive blocking by sendmail at the university: for example, all hosts without a valid reverse DNS are blocked at the connection level.

7. CONCLUSION

The results of test runs in the previous section unequivocally answer our research question that a mail filter that does not analyze the contents of e-mail messages can rival a conventional state-of-the-art mail filter in performance. Furthermore, our approach shows that a simpler algorithm can, at the same time, lower the false positive rate, the false negative rate and the resource consumption, compared to a more complex algorithm like the reference SpamAssassin installation at the University of Twente.

8. FINAL REMARKS

Further tests are needed to determine the usefulness of our approach compared to, or in addition to, other conventional mail filters, for different kinds of e-mail traffic over longer periods of time. However, the very low false positive rate is very encouraging and a stable basis for further development. In particular, we think that our approach is well suited for use in a global sensor (telescope, honeypot) network that can feed data into traditional blocklists, but it could also be used as a separate front-end

[1]*The single false positive for svmilter in the test runs was later resolved to be an enduser submitting a message from a 'bad neighborhood' directly to the mail exchanger. This unusual scenario can be solved by requiring end-users to perform SASL authentication to the mail server.*

DOI: 10.1002/nem

filter (to reduce the load on more complex mail filters), an addition to regular SpamAssassin rules or even as a stand-alone, single line of defense against spam for a mail server.

Globally it remains important to have heterogeneous filters, not only to prevent weakest link failure, but also because these filters create a feedback loop. For example, the BNBL is composed of DNSBLs that might have been created using content filtering with SpamAssassin. Entries in the BNBL can then be used as an additional input for SpamAssassin.

During development we also encountered several issues—not research per se—to which the academic community is especially well suited for solving. The following three stand out, in order of importance:

1. List of registrable domains for URI normalization. No accepted method for composing such a list or source for distributing it exists, thereby affecting the effectiveness of mail URI extraction and pushing spammers to less well-known and more exotic domains.
2. URI blacklist. The art of creating an effective URI blacklist is much more difficult compared to creating RBL, since the risk of erroneously listing URIs is real. Currently only two reliable (near-zero false positive) blacklists (SURBL and URIBL) and one less reliable (some false positives) blacklist (SWINOG [36]) exist. Because URI detection is the single highest score for detecting spam in our test runs, creating another, independently managed URI blacklist is certainly worthwhile. Also, as an added feature, support for mailto URIs could be integrated, which is currently lacking from all three blacklists.
3. Realtime Blacklist. Although many DNS RBLs exist, only a very small number of these are reliable. Some of these are managed anonymously or by a company with a vested business interest and therefore could disappear overnight (like the DSBL did during development) or have access conditions changed.

Finally, the techniques should be bolstered to deal with shared hosting (e.g., blogspot.com), redirectors (e.g., tinyurl.com) and IPv6 (construction of the bad neighborhood list should be adopted to deal with different subnetting).

ACKNOWLEDGEMENTS

This work was supported in part by the European Network of Excellence for the Management of Internet Technologies and Complex Services (EMANICS), funded by the European Union under contract number FP6-2004-IST-026854-NoE. We would like to thank Peter Peters, Maarten Aertsen and Gert Vliek for their support while performing this research.

REFERENCES

1. Symantec. *State of Spam: A Monthly Report*, June 2009. http://www.symantec.com/ [9 June 2010].
2. Sophos. Only one in 28 emails legitimate, July 2008. http://www.sophos.com/pressoffice/news/articles/2008/07/dirtydozjul08.html [9 June 2010].
3. Spamhaus. *Effective Spam Filtering*. http://www.spamhaus.org/effective_filtering.html [July 2009].
4. The SpamAssassin Project. *SpamAssassin*. http://spamassassin.apache.org/ [July 2009].
5. Hoanca B. How good are our weapons in the SPAM wars. *IEEE Technology and Society magazine* 2006; **25**(1): 22–30.
6. Desika P, Srivastava J. Analyzing network traffic to detect e-mail spamming machines. In *Proceedings of the 2004 ICDM Workshop on Privacy and Security Aspects of Data Mining (PSDM'04)*, Brighton, UK, November 2004.
7. Schatzmann D, Burkhart M, Spyropoulos T. Inferring spammers in the network core. In *Proceedings of 10th International Conference on Passive and Active Network Measurement (PAM'09)*, Seoul, Korea, April 2009.
8. Ramachandran A, Feamste N, Vempala S. Filtering spam with behavioral blacklisting. In *Proceedings of the 14th ACM conference on Computer and Communications Security (CCS'07)*, Alexandria, VA, October 2007.

DOI: 10.1002/nem

9. Pathak A, Hu YC, Mao ZM. Peeking into spammer behavior from a unique vantage point. In *Proceedings of the 1st USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET'08)*, San Francisco, CA, April 2008.
10. Kreibich C, Kanich C, Levchenko K, Enright B, Voelker GM, Paxson V, Savage S. On the spam campaign trail. In *Proceedings of 1st USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET'08)*, San Francisco, CA, April 2008.
11. Zhuang L, Dunagan J, Simon DR, Wang HJ, Osipkov I, Hulten G, Tygar JD. Characterizing botnets from email spam records. In *Proceedings of 1st USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET'08)*, San Francisco, CA, April 2008.
12. Cheng B-C, Chen M-J, Chu Y-S, Chen A, Yap S, Fan K-P. A stateful and flow-based intrusion prevention system for email applications. In *Proceedings of the IFIP International Conference on Network and Parallel Computing (NPC'07)*, Dalian, China, September 2007.
13. Schatzmann D, Burkhart M, Spyropoulos T. *Flow-level characteristics of spam and ham*. TIK Report No. 291, Computer Engineering and Networks Laboratory, ETH, Zurich, August 2008.
14. Žádník M, Michlovský Z, *Is spam visible in flow-level statistic?* CESNET technical report, June 2008.
15. Sperotto A, Vliek G, Sadre R, Pras A. Detecting spam at the network level. In *Proceedings of the 15th EUNICE International Workshop (EUNICE'09)*, Barcelona, Spain, September 2009.
16. Vliek G. *Detecting spam machines, a Netflow–data based approach*. MSc thesis, University of Twente, Netherlands, February 2009. http://purl.org/utwente/e58583 [9 June 2010].
17. Spamhaus Project. ZEN. http://www.spamhaus.org/zen/ [July 2009].
18. IronPort Systems. SpamCop.net—SpamCop FAQ: What is the SpamCop Blocking List (SCBL)? http://www.spamcop.net/fomserve/cache/297.html [July 2009].
19. Makey J. Blacklists compared. http://www.sdsc.edu/~jeff/spam/Blacklists_Compared.html [July 2009].
20. Jung J, Sit E. An empirical study of spam traffic and the use of DNS black lists. In *Proceedings of the ACM SIGCOMM Internet Measurement Conference (IMC'04)*, Taormina, Sicily, October 2004.
21. Ramachandran A, Feamster N. Understanding the network-level behavior of spammers. In *Proceedings of the 2006 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM'06)*, Pisa, Italy, September 2006.
22. Ramachandran A, Dagon D, Feamster N. Can DNS-based blacklists keep up with bots? In *Proceedings of the 3rd Conference on Email and Anti-Spam (CEAS'06)*, Mountain View, CA, July 2006.
23. Rajab MA, Zarfoss J, Monrose F, Terzis 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 (HotBots'07)*, Cambridge, MA, April 2007.
24. Munroe R. Map of the Internet: the IPv4 space, 2006. http://www.xkcd.com/195/ [July 2009].
25. Irwin B, Pilkington N. High level Internet scale traffic visualization using Hilbert curve mapping. In *Proceedings of the Workshop on Visualization for Computer Security (VizSEC 2007)*, Sacramento, CA, October 2007.
26. Collins MP, Shimeall TJ, Faber S, Janies J, Weaver R, De Shon M, Kadane JB. Using uncleanliness to predict future botnet addresses. In *Proceedings of the ACM SIGCOMM Conference on Internet Measurement (IMC'07)*, San Diego, CA, October 2007.
27. Langelund P. PI assignment size, August 2006. http://www.ripe.net/ripe/policies/proposals/2006-05.html [9 June 2010].
28. SURBL. http://www.surbl.org/ [July 2009].
29. URIBL.COM. URIBL.COM: Realtime URI Blacklist. http://www.uribl.com/about.shtml [July 2009].
30. CBL. http://cbl.abuseat.org/ [July 2009].
31. DSBL.org. http://www.dsbl.org [November 2008].
32. van Riel R. Passive Spam Block List. http://psbl.surriel.com/about/ [July 2009].
33. Spamhaus Project. DROP. http://www.spamhaus.org/drop/ [July 2009].
34. Tokarev M. rbldnsd: small daemon for DNSBLs. http://www.corpit.ru/mjt/rbldnsd.html [July 2009].
35. Combs G. Wireshark. http://www.wireshark.org/ [July 2009].
36. ImproWare. Swinog URIBL. http://antispam.imp.ch/05uribl.php?lng=0 [July 2009].

AUTHORS' BIOGRAPHIES

Aiko Pras is an associate professor in the Departments of Electrical Engineering and Computer Science at the University of Twente, The Netherlands, where he is leading the Design and Analysis of Communication Systems

DOI: 10.1002/nem

Group. He received a PhD degree for his thesis entitled 'Network Management Architectures'. His research interests include network management technologies, network monitoring, measurements and security. He is chairing the IFIP Working Group 6.6 on 'Management of Networks and Distributed Systems' and is Research Leader in the European Network of Excellence on 'Management of the Internet and Complex Services' (EMANICS). He is steering committee member of several conferences, including IM/NOMS and manweek, and series/associate editor of ComMag and IJNM.

Ward van Wanrooij is a self-employed high-tech entrepreneur. In 2007 he received a MSc degree (cum laude) in computer science at the University of Twente. His work focuses on the practical application of research in distributed systems and has been published in several conference proceedings. He is currently involved as a consultant to several high-tech companies.

DOI: 10.1002/nem