Separating Wheat from the Chaff: A Deployable Approach to Counter Spam

Youngsang Shin, Minaxi Gupta, Rob Henderson
Computer Science Dept
Indiana University
Bloomington, IN, U.S.A.
{*shiny,minaxi,robh*}*@cs.indiana.edu*

Aaron Emigh
Radix Labs
Incline Village, NV, U.S.A.
aaron@radixlabs.com

Abstract

Unsolicited bulk email, aka spam is a persistent threat to the usefulness of the Internet. The fight against spam today relies solely on filtering at the recipient's mail server, which can delay mail delivery. We present a spam countering approach consisting of two complementary techniques. The first, *token-based authentication*, can identify emails from valid senders that a user expects to hear from. This identification prevents much of the good mail from being subjected to the filtering process. The second technique, *history-based prioritization*, is designed for the rest of the email. It utilizes past information about the sending mail servers and their domains to prioritize email filtration, thus reducing delays in the delivery of good email over spam. The proposed techniques do not rely on any infrastructure, deployed or otherwise, and any mail server can choose to deploy them independent of the choice made by the any other mail server.

1 Introduction

Some estimates claim that $60-80\%$ of emails today are spam. Spam has become such a huge problem that it threatens to render email itself unusable. Estimates on the cost of spam vary but they all are staggering. FTC estimates that an average person spends approximately 10 minutes each day dealing with spam. In fact, at the rate spam is growing the Radicati group estimates that it would cost businesses $198 billion by 2007! Further, an average spam is about 10KBytes in size. This translates into a huge amount of bandwidth and disk space that is wasted in delivering and storing spam respectively.

Beyond expensive *legal* approaches that seek to unmask the spammers with the goal of prosecuting them [7], the proposed and pursued approaches in the fight against spam can be classified in two broad categories: 1) *sender verification* approaches such as domain keys identified mail (DKIM) [4], and sender ID [9, 11] that aim to answer the question: "Did the sender actually send this email?" and can help in pinpointing the spammers and 2) *mitigation* approaches [2, 13, 12, 8, 14, 6, 5] that either attempt to deter spammers, or filter incoming or outgoing emails for spam. The sender verification approaches rely on being able to make infrastructural changes to the domain name system (DNS) which limits their immediate deployability. Also, their success hinges on the security of the DNS itself, for which solutions are still being worked on. In this work, we focus on the mitigation techniques due to their deployability. In particular, *the goal of the work presented in this paper is to develop an effective spam mitigation approach for organizations that is immediately deployable and does not require any cooperation from anyone outside the organization*. Our approach is complementary in nature to several of the existing mitigation techniques. In particular, the mitigation techniques that throttle spam at the senders [14], use greylisting [8][1], or impose quotas on senders through peer cooperation [2, 13] can be used in addition to our approach. On the other hand, other mitigation techniques that employ collaborative spam filtering based on social email networks [6], predict spam patterns [5], or prioritize email filtering for good mail over spam [12] are competing approaches to ours.

Our approach consists of two techniques. The first technique, *token-based authentication*, allows implementing sender's mail server to put a receiver specific *authentication token string* (referred to as *token* subsequently) in the header of every outgoing email. Replies to such emails would contain the token even if the recipient does not implement the technique if a standard field that gets copied in the reply is used to put the token[2]. Presence of such tokens in incoming mails is then used to identify valid emails which prevents much of the good mail from being subjected to the filtering process. This technique enables email sender's authentication only with simple token management and exchange unlike existing email authentication schemes such as

S/MIME [10] and pretty good privacy (PGP) [1] which require expensive operations and face deployability issues. The second technique we propose, *history-based prioritization*, complements the first technique and is designed for the rest of the incoming email that does not contain tokens. It builds on the email prioritization scheme proposed in [12] and utilizes past information about the sending mail servers, along with the information about their domains to prioritize email filtration. It reduces delays in the delivery of good email over spam because sophisticated spam filtration, which uses a combination of machine learning techniques such as Bayesian filtering [3], DNS-based blacklists of known offenders, and whitelists of good senders, can introduce substantial delays in mail delivery. The design of both the proposed techniques allows any mail server to deploy them independent of the choice made by the any other mail server in the Internet, making them immediately deployable. We evaluate the efficacy of the proposed techniques by analyzing 7 months of departmental email logs.

The rest of this paper is organized as follows. Section 2 describes the data used in analyzing the effectiveness of our approaches. Both our techniques, along with the analysis of each using logs, are presented in Section 3. Finally, Section 4 offers concluding remarks.

2 Data Used in Analyzing Our Approaches

In order to analyze the effectiveness of our approaches, we collected 7 months of Sendmail logs from Indiana University's Computer Science Department, starting April 10th, 2005. An entry in our log is created for each unique email message received and contains the timestamp, IP address and name (if the reverse DNS lookup by the department's mail server was successful) of the sending MTA, anonymized information about the senders and receivers of that email, and a status code for the mail. The status code indicates whether the mail was perceived as a spam or not according to the filtering program. In addition to the data on incoming SMTP connections, we also have anonymized information about all the outgoing mails in the department. Table 1 shows an overview of the data available to us. We exclude all mails where both the sender and receiver are local to focus only on emails that could potentially be spam.

3 Our Approach

Email filtration is a double edged sword: the more sophisticated a software becomes, the lower the false positives and negatives, but the higher the processing times. Table 2 shows the processing times for the $130,122$ emails received at our department's mail server for a week. These times range from 0.32 seconds to 3559.56 seconds. Due to the large range the mean, median, and standard deviation are 3.81, 1.48, and 21.63 seconds respectively. It is noteworthy that even though 76.8% of the emails took less than or equal to 3 seconds to finish the filtering process, 5.7% took greater than 10 seconds to process!

Table 1: Overview of the data.

Duration of logs	211 days
Number of *incoming* SMTP connections	3,415,219
Number of *outgoing* SMTP connections	806,215
Number of unique sending MTAs	879,670
Sending MTAs that are *DNS resolved*	505,443
Number of unique sending domains	25,760

Table 2: Processing time distribution.

Range (sec)	Number of messages	% of messages
0.0-1.0	29,094	22.4
1.0-2.0	55,892	43.0
2.0-3.0	14,817	11.4
3.0-4.0	9,448	7.3
4.0-5.0	4,406	3.4
5.0-6.0	2,825	2.2
6.0-7.0	2,041	1.6
7.0-8.0	1,646	1.3
8.0-9.0	1,374	1.1
9.0-10.0	1,104	0.8
> 10.0	7,475	5.7

Our approach aims to prioritize filtration of good emails over spam by using a divide and conquer strategy. We first separate bulk of the good mail using a token-based authentication scheme which we describe in Section 3.1. For the rest of the emails, we use history information about the sending MTA and its domain to predict the goodness/badness of the incoming mail.

3.1 Token-based Authentication

Token-based authentication is an authentication scheme to identify emails from valid senders that a user expects to hear from. It enables a mail server to deliver such mails to its users without subjecting them to the spam filtering process. An MTA deploying this scheme assigns a unique token for each (sender, receiver) pair. The token

is nothing but an alphanumeric string of a small number of bytes, say 64. For each outgoing email, the sending MTA, S, picks the relevant tokens for that sender. For example, if the mail was sent to just one receiver, the sending MTA puts the corresponding token for that receiver in the email header and if the mail is destined for multiple receivers, tokens for each individual receivers are put. Mailing lists can be allocated unique tokens to avoid increasing the email size substantially due to tokens.

This token is returned back to the sender if the receiver R chooses to reply to sender's email. Upon receiving such replies, S can infer that those are not spam by simple comparing the token in the mail to the one it has stored locally, thus delivering such mails immediately without subjecting them to the filtering process. Notice that the tokens accomplish more than what the sending MTA can accomplish simply by keeping track of the receivers for all outgoing mails. This is because sender information in incoming mails can be forged.

The token-based authentication scheme can be deployed even when receiver R does not implement the scheme. Today, this can be accomplished if an MTA uses the *Message-ID* field of the outgoing email header to insert the tokens. This field is copied by most replying MTAs into the *In-Reply-To* field of the reply messages. Thus, emails users expect to get can be separated from the others without any change to the receivers.

The above discussion assumes that the sending MTAs choose tokens for all the (sender, receiver) pairs and the same tokens are retained for all subsequent communications. This scheme can be misused by the spammers who have access to email logs over the Internet. Thus, the assigned tokens should be periodically changed to avoid the security vulnerability of a long lived token. Further, in order to keep the number of tokens bounded, some mechanism for expunging unused tokens would have to be devised.

Alternate schemes for token assignment are possible. For example, instead of having a token per (sender, receiver) pair, the token could be per sender, or per receiver. Both of these schemes reduce the number of tokens an implementing MTA has to keep track of but suffer from individual shortcomings. The sender-based token scheme prevents the senders from being able to specify which receivers they regularly communicate with, something our proposed scheme can easily be extended to do. Similarly, the receiver-based token scheme implies that there will be one token per receiver, irrespective of how many senders communicate with that receiver. This would also prevent senders from specifying which receivers they would like to maintain tokens with because even the same receiver could have diverse meanings to different senders.

3.1.1 Effect of Token-based Authentication

To see how many incoming emails would the token-based authentication scheme impact, we analyzed the data on incoming and outgoing SMTP connections (described in Section 2). Essentially, using the sender and receiver information contained in each incoming and outgoing connection, we extracted the information on unique (sender, receiver) pairs, and the pairs where both parties communicated with each other. Incoming mails with the latter would potentially contain tokens. Table 3 shows the number of pairs observed in our data and the number of SMTP connections that would be affected by the token-based authentication scheme.

Table 3: Incoming mails potentially benefited by token-based authentication.

Number of unique (sender, receiver) pairs	3,051,559
Unique pairs with 2-way communication	12,118
Percentage	0.4%
Total incoming SMTP connections	3,415,219
Incoming connections with tokens	105,891
Percentage	3.1%

Although it appears that the results in Table 3 are disheartening, such is not the case. This is because the small percentage of mails containing the tokens are likely to be the most important and urgent. And these are exactly the ones whose delays are avoided through the use of the token-based authentication scheme. Moreover, the total number of unique pairs are likely to be unusually high due to the use of distinct sender names in spam ($2,988,519$ of the total $3,051,559$ sender/receiver pairs were the kind where an incoming message was not responded to by the recipient), many of which are regularly forged. Further, our results are likely to be pessimistic due to common practice among the students of our department of forwarding emails to their other accounts, commonly the university-wide account that is accessible from anywhere through a Web-based interface.

Table 3 assumes that if a valid token is contained in an incoming mail, it is a good mail. In the case of incoming emails that contain multiple receivers, the above assumes that the mail would be regarded as good by all receivers. In reality, such emails may have to be subjected to the filtering process. To account for such cases, we also considered *total mails*, where an incoming SMTP connection containing n recipients is considered as n mails. Using this notion of total mails, we found that out of the total $4,468,806$ mails, $108,485$ mails (2.4%), would contain tokens and could be authenticated. Table 4 shows

the remaining data on incoming emails that we consider for the history-based analysis.

Table 4: Data after removing emails authenticated by tokens.

Duration of logs	211 days
Number of *incoming* SMTP connections	3,309,328
Number of unique sending MTAs	876,346
Sending MTAs that are *DNS resolved*	502,493
Number of unique sending domains	25,259

3.2 History-Based Prioritization

Even though the token-based authentication scheme is able to prioritize some good mails over others, it accounts for only 3.1% of the incoming SMTP connections. For the rest of the mails, we explore history-based prioritization whose goal is to use past behavior of mail servers and their domains to decide which mails should be prioritized by the spam filters. This allows the likely good mails to be delivered to the recipients faster than a first come first served (FCFS) filtering policy would allow.

The basic notion of history-based prioritization has been examined earlier. In [12], authors utilize the past history of the sending MTAs (referred to as *server history* subsequently) to prioritize which mails should be seen by the spam filters first. They consider an MTA worthy of prioritizing mails from if it sent at least 50% good emails in the past. Using this simple strategy on 69 days of email logs, they achieved accuracies of $74-80\%$ for good mails, $93-95\%$ for spam and viruses, and aggregate accuracies of at most 90%. Also, they found that maintaining a history of $100,000$ past SMTP connections was sufficient to achieve these accuracies.

We explore history-based prioritization in a more general sense with an aim to improve the prioritization accuracies for good mails. Also, while work in [12] focused only on mail servers that sent 10 or more emails, we wish to achieve good results for all MTAs, including the ones that sent fewer than 10 emails. In particular, we first seek the answers to the following questions to find out the parameters that can be used by a history-based prioritization scheme:

- Is the mail more likely to be a spam if the reverse DNS lookup fails on an IP address? We ask this question because many filtering programs compare the domain name contained in an email message to that obtained by the reverse DNS lookup and report it in the email header.
- Is the sending history of a domain indicative of whether the mail is likely to be a spam or not? How does its accuracy compare with the approach used in [12], where only MTA's sending history was utilized?
- Is the number of total mails or average mails per day sent by an MTA or a domain an indicator of whether they mostly send spam or not?
- Is the number of days an MTA is active for a predictor of what kind of mails it would send?
- Does the number of servers per domain predict what kind of emails would an MTA send? We are interested in this question because botnets are often comprised of compromised home machines, who belong to the domains of the service providers, which are relatively fewer in number compared to the total number of domains.

Based on the answers to the above questions, our goal is to incorporate the useful parameters into an algorithm that can then be used by an MTA to prioritize incoming emails for spam filtering. We now find out answers to the above questions one by one.

3.2.1 Reverse DNS Lookup

To explore if any relationship exists between the success of reverse DNS lookup on the name of the sending MTA and the probability of the mail being spam, we compared the good and junk mails sent by MTAs for whom the name resolution fails and the ones for whom the reverse DNS resolution succeeds. The data used in conducting this analysis is from Table 4 and the results are shown in Table 5.

Table 5: Good and junk mail statistics for MTAs.

		unique MTAs	%	SMTP connections	%
unresolved	good only	24,484	2.79	45,090	1.36
	junk only	340,287	38.83	829,779	25.07
	both	9,082	1.04	121,600	3.67
	subtotal	*373,853*	*42.66*	*996,469*	*30.11*
resolved	good only	42,616	4.86	255,774	7.73
	junk only	447,137	51.02	1,069,925	32.33
	both	12,740	1.45	987,160	29.83
	subtotal	*502,493*	*57.34*	*2,312,859*	*69.89*
	total	876,346	100	3,309,328	100

The results presented in Table 5 indicate that over 91% of MTAs for whom reverse DNS lookup fails (38.83% of all MTAs) actually established SMTP connections to

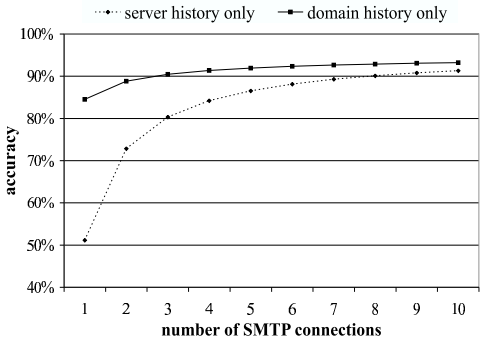

Figure 1: A comparison of using server history versus domain history for MTAs that sent 10 or more emails.

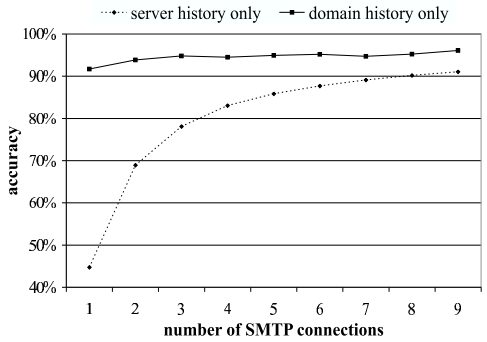

Figure 2: A comparison of using server history versus domain history for MTAs that sent less than 10 emails.

send only junk emails, where a junk email is either a spam or a virus. The corresponding SMTP connections are 83% of the total SMTP connections established by such MTAs. Alternately, only 6.5% of the MTAs for whom name resolution fails established SMTP connections to send only good emails, and only 2.43% of such MTAs actually sent emails that the spam filter perceived as both good and junk. *This implies that the failure of reverse DNS lookup is a strong indicator of what kind of email would an MTA send*.

3.2.2 Domain History

We now use the data in Table 4 to investigate if the history of sending MTA's domain, either by itself or in conjunction with server history, can be used to predict the nature of an incoming SMTP connection. Unlike [12], where the first email from an MTA was considered to be bad (leading to a higher accuracy for junk mails at the cost of good mails which are smaller in percentage), we consider the first incoming mail from an MTA to be bad only if the reverse DNS lookup fails. Otherwise, the incoming mail is considered good. We now observe the overall prediction accuracies when just the domain history and just the server history are used for prediction. The algorithm we use in computing history is: consider an incoming email to be bad if 50% of the past mails sent by the MTA or domain were bad, and good otherwise.

Figures 1 and 2 show the average prediction accuracy for incoming emails as more domain and server history are available (after seeing about 11 emails from a particular domain or MTA, the accuracy seems to stabilize). We split the data into servers that sent greater than or equal to 10 messages overall in our log and those that sent less than 10 mails to see the difference in accuracy results for servers that sent a small number of messages versus those who sent larger number of messages. This is because 91.6% of messages sent by MTAs in the latter set are spam (compared to 50% in the former case) and overshadow the aggregate accuracy results.

Figures 1 and 2 seem to indicate the domain history is a better predictor of the nature of an incoming mail. However, one needs to be careful before reaching this conclusion because the average prediction accuracy numbers are heavily biased by junk messages, which far outnumber the good emails. At this point, the only conclusion we can draw is that perhaps a combination of both server and domain history would be a better predictor of the nature of an email than just the server history, which was used in [12].

The first email from an MTA about which nothing is known deserves special attention. This is because such emails comprise 26.5% of the total incoming emails according to Table 4. While work in [12] assumes all such *first time emails* to be bad, we have so far used the results from our reverse DNS lookup failure analysis and assumed that only the first time emails from MTAs for whom the reverse DNS lookup fails are bad. We now explore if the use of domain history of the sending MTA, where available, helps predict the first time emails from unknown MTAs better. Table 6 compares the accuracy of first time emails for 1) assumption in [12] (all first time emails are junk), 2) our previous assumption (a first time email is bad if reverse DNS lookup fails), and 3) our first assumption along with the use of domain history where available. *The results show that using domain history improves the accuracy of good first time mails without compromising the overall prediction accuracy or the prediction accuracy of junk mails*.

3.2.3 Average Mails per Day and Total Mails

We now explore the relationship between the number of total and average mails sent by an MTA and the probability that the next mail from that MTA is spam. For each MTA, the average mails per day is computed by dividing the total mails it sent by the number of days it had been *active* for. We define active duration for an MTA to be

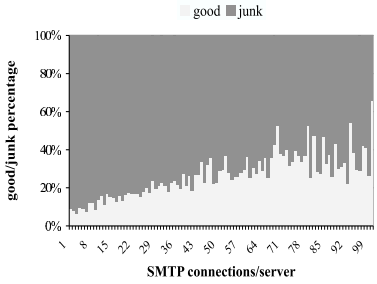

Figure 3: Relationship between total mails sent by an MTA to the percentage of good/junk mails it sent.

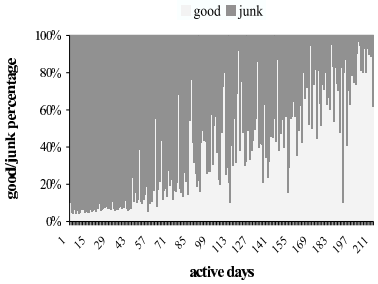

Figure 4: Relationship between active days of an MTA to the percentage of good/junk mails it sent.

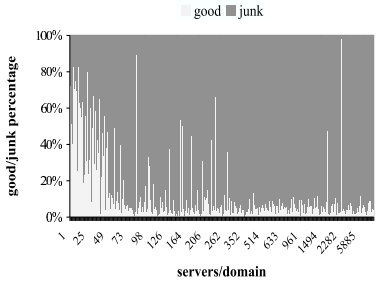

Figure 5: Relationship between number of MTAs per domain to the percentage of good/junk mails each sent.

Table 6: Prediction accuracy for first time emails from MTAs.

	All bad	Bad if DNS resolution failed	Use domain history where available
good mails	0%	62.54%	23.51%
junk mails	100%	43.18%	97.23%
average	90.89%	44.94%	92.00%

the number of days from when we witness a mail from it the first time to the last time it establishes an SMTP connection. *Figure 3 indicates that MTAs with relatively fewer total emails sent more spam than good mails*. The results for average mails per day were inconclusive and are omitted due to space constraints. Further, notice that using average and total mails in email prioritization is tricky because these numbers can change over time. We utilize the available past information to compute these numbers in the algorithm we design subsequently.

3.2.4 Active Days

A related parameter to the total number of mails is the duration for which a server stays active. Figure 4 shows the relationship between the number of days an MTA stays active in our log and the percentage of good/junk mail it sends. *We conclude that the longer an MTA stays active, the less spam it sends*.

3.2.5 Servers per Domain

Figure 5 shows the relationship between the number of MTAs in a domain and the percentage of good and junk mails. *It shows that the more the number of MTAs belonging to a domain, the more the likelihood that those MTAs will send more junk mails than good ones.* This is intuitive since much of the spam today is sent by compromised machines that are part of large botnets comprising of hundreds of thousands of machines. And many of the compromised machines belong to home users, who subscribe to handful of the popular big Internet service providers (ISPs).

3.2.6 Using Above Parameters in an Algorithm

With the goal of predicting incoming good emails as accurately as possible, we now use server history information and the above parameters to design a prediction algorithm for prioritizing emails for filtration. The algorithm presented in algorithm 1, computes the probability P_i that an incoming mail i is good in three separate cases: 1) when no server history information is available, 2) when server history is between 0.4 and 0.6 (implying the server sends both good and junk mails with close enough probabilities – this special case is required for servers like *yahoo.com* that aggregate mails from all kinds of users), 3) when server is either less than 0.4 (implying it sends junk mails most of the time) or greater than 0.6 (implying that it sends good mails most of the time). An incoming mail is considered to be good if $P_i \geq 0.5$. After the prediction of each incoming mail, the server and domain histories (referred to as *good mail probabilities*) for the sending MTA and its domain are updated according the following:

$$GMP(M_i) = \frac{N_{good}(M_i)}{N_{total}(M_i)} \quad (1)$$

$$GMP(D_i) = \frac{N_{good}(D_i)}{N_{total}(D_i)} \quad (2)$$

In equations 1 and 2, N_{good} and N_{total} are the numbers of good and total SMTP connections seen so far from MTA M_i and domain D_i respectively.

Parameters ρ, ϵ, and τ used in our algorithm are for total mails from mail i's server, relative days server for mail i was active for with respect to log duration at the time of measurement, and number of servers seen from the same domain as mail i's server respectively. They are chosen to be 10, 0.6, and 50 respectively based on our

Table 7: Performance comparison of our algorithm with the server history algorithm presented in [12].

	MTAs that sent 10 or more mails		MTAs that sent less than 10 mails		All MTAs	
	server history only	our algorithm	server history only	our algorithm	server history only	our algorithm
good	85.94%	95.39%	33.49%	35.37%	80.40%	90.10%
junk	87.43%	77.09%	99.04%	98.29%	93.49%	89.14%
average	86.42%	86.71%	92.34%	92.85%	89.39%	89.79%

Algorithm 1 Our algorithm.

if no history information available for M_i **then** {// case 1}
  if no domain history information available for D_i **then**
    if M_i's reverse DNS lookup failed **then**
      $P_i = 0.0$ {// mail is junk}
    else
      $P_i = 1.0$ {// mail is good}
    end if
  else
    $P_i = \gamma * GMP(D_i)$ {// γ allows tuning}
  end if
else if $0.4 \leq GMP(M_i) \leq 0.6$ **then** {// case 2}
  if the previous mail from M_i was good **then**
    $P_i = 1.0$
  else {// consider weighted average of server and domain histories}
    $P_i = \alpha * GMP(M_i) + \beta * GMP(D_i)$
    if $AD_i > \epsilon$ **then** {// consider days server is active for}
      $P_i = \lambda * P_i$ {// λ allows tuning}
    else if $SD_i > \tau$ **then** {// consider number of servers per domain}
      $P_i = \delta * P_i$ {// δ allows tuning}
    end if
  end if
else {// case 3}
  if $TM_i < \rho$ **then** {// consider weighted average if total mails from this server $< \rho$}
    $P_i = \alpha * GMP(M_i) + \beta * GMP(D_i)$
  else {// otherwise only consider server history}
    $P_i = GMP(M_i)$
  end if
end if

log. The tunable parameters γ, α, β, λ, and δ are chosen to be 0.7, 0.3, 0.7, 1.3, and 0.8 respectively. These are chosen to maximize the accuracy of prediction of good mails over junk mails, while ensuring that the overall prediction accuracy across all types of mails is not compromised. The latter is important to ensure that the spam filters do not end up dealing with much junk mail while processing the good mails.

Table 7 and Figure 6 show the accuracies of prediction resulting from our algorithm and compares it to the server history information algorithm used in [12]. Though we set the above mentioned parameters using the log, we do not explicitly use a training phase for either of the algorithms and instead evaluate their effectiveness as sending information becomes available. We show three view points in Table 7, 1) aggregate accuracies across all types of MTAs, 2) accuracies for MTAs that sent 10 or more emails, and 3) accuracies for MTAs that sent less than 10 emails. Figure 6 shows only the second view of point. Overall, our algorithm substantially outperforms the server history algorithm in prioritizing good emails for all MTAs, and for MTAs that sent 10 or more emails. In all other categories, we perform at least as well. In comparison, Return Path[3], a company that monitors email performance for online marketers, estimates that current spam filters misclassify nearly 19 percent of good email as spam.

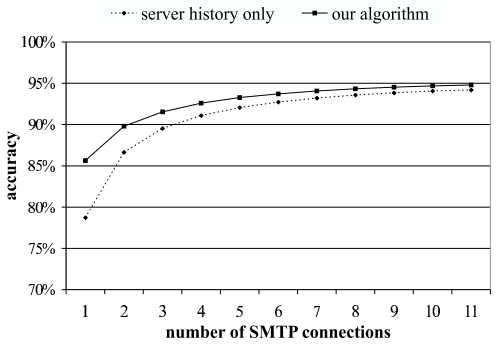

Figure 6: Comparison of prediction accuracy of our algorithm with algorithm presented in [12] as number of SMTP connections increases (for MTAs with greater than or equal to 10 SMTP connections).

4 Concluding Remarks

We have shown that the combination of token-based authentication and history-based prediction can help in delivering good mails to their recipients much faster than spam filtering alone. We believe that our history-based prediction algorithm does as best as an algorithm can do. The reason for this is that some MTAs (39% in our data) consistently sent both types of mails. These appear to be MTAs like *hotmail.com* that serve users with varying intentions, perhaps including spammers.

References

[1] ATKINS, D., AND ET AL. Pretty Good Privacy. RFC 1991, Aug. 1996.

[2] BALAKRISHNAN, H., AND KARGER, D. Spam-I-am: A Proposal for Spam Control using Distributed Quota Management. In *3rd ACM SIGCOMM Workshop on Hot Topics in Networks (HotNets)* (Nov. 2004).

[3] A plan for spam. `http://www.paulgraham.com/spam.html`.

[4] Domain Keys Identified Mail (DKIM). `http://mipassoc.org/dkim/`.

[5] GOMES, L. H., CASTRO, F., ALMEIDA, V., ALMEIDA, J. M., ALMEIDA, R. B., AND BETTENCOURT, L. Improving spam detection based on structural similarity. In *USENIX Workshop on Steps to Reducing Unwanted Traffic on the Internet* (July 2005).

[6] KONG, J., BOYKING, P., RCZACI, B., SARKAR, N., AND ROYCHOWDHURY, V. Scalable and reliable collaborative spam filters: harnessing the global social email networks. In *Conference on Email and Anti-Spam* (July 2005).

[7] KORNBLUM, A. Searching for John Doe: finding spammers and phishers. In *Conference on Email and Anti-Spam* (July 2005).

[8] LEVINE, J. Experiences with greylisting. In *Conference on Email and Anti-Spam* (July 2005).

[9] Microsoft's SenderID framework. `http://www.microsoft.com/mscorp/safety/technologies/senderid/default.mspx/`.

[10] S/MIME working group. `http://www.imc.org/ietf-smime/`.

[11] Sender Policy Framework. `http://spf.pobox.com/`.

[12] TWINING, R. D., WILLIAMSON, M. W., MOWBRAY, M., AND RAHMOUNI, M. Email prioritization: reducing delays on legitimate mail caused by junk mail. In *USENIX Annual Technical Conference (USENIX)* (July 2004).

[13] WALFISH, M., ZAMFIRESCU, J., BALAKRISHNAN, H., KARGER, D., AND SHENKER, S. Distributed Quota Enforcement for Spam Control. In *3rd USENIX Symposium on Networked Systems Design and Implementation (NSDI)* (May 2006).

[14] ZHONG, Z., HUANG, K., AND LI, K. Throttling outgoing spam for Webmail services. In *Conference on Email and Anti-Spam* (July 2005).

Notes

[1] Mail servers using greylisting temporarily reject any email from senders that they do not recognize under the assumption that legitimate mail servers will retry later.

[2] An example of such a field is the message identifier put by the mail server in the *Message-ID* field of the email header. This field is copied into the *In-Reply-To* field in the replies to this message by most mail clients.

[3] http://www.returnpath.net.