The Pennsylvania State University

The Graduate School

Department of Computer Science

EMAIL COMMUNITIES OF INTEREST AND THEIR APPLICATION

A Thesis in

Computer Science

by

Lisa Y Johansen

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Master of Science

May 2008

The thesis of Lisa Y Johansen was reviewed and approved* by the following.

Patrick McDaniel
Associate Professor of Computer Science and Engineering
Thesis Adviser

Wang-Chien Lee
Associate Professor of Computer Science and Engineering

Raj Acharya
Professor of Computer Science and Engineering
Head of the Department of Computer Science and Engineering

*Signatures are on file in the Graduate School.

Abstract

Email has become an integral and sometimes overwhelming part of users' personal and professional lives. Due to the number of emails and their wide range of importance, managing this communication medium has become a intensely researched topic. In communication networks, the identification of communities of interest (COIs) – groups of users that share a common bond – has proven highly applicable in automating various tasks. In this thesis, we measure the flow and frequency of user email toward the identification of COIs. Through this identification, we hope to enable the automation of some of the management tasks associated with email. We begin by analyzing a large corpus of university email in order to drive the development of algorithms for automatically determining COIs in email. We then validate the proposed algorithms by evaluating their ability to serve as an automated priority filter. Our analysis shows that the proposed algorithms correctly identify email as being sent from the human-identified COI with high accuracy. This indicates that a significant amount of information can be determined solely from the sender and receiver of an email.

Identification of COIs in email communication can be highly applicable in a variety of email-related applications. In the second part of our research, we look at a possible application for COI: an automated reputation service for use with DKIM (DomainKeys Identified Mail). We describe a COI-based domain reputation service, analyze its ability to identify relationships between users and domains and compare its characteristics to current reputation services. We discuss the benefit of employing the identification of communities of interest in email.

Table of Contents

List of Tables

List of Figures

Acknowledgments

I would like to thank Dr. Patrick McDaniel for giving me the idea for Communities of Interest in email, for introducing me to Perl, and for his comments, help, and support.

I would like to thank my parents and family for always supporting me no matter what I choose to do and receiving my successes with joy and pride.

Finally, I would like to thank Mike not only for being central to the research presented in this thesis, but for his unwaivering love and support.

Chapter 1

Introduction

Electronic mail has profoundly changed the nature of personal communication. It allows users to communicate with anyone, anywhere, at any time. It is easy to use, reliable, and fast. It is paradoxically asynchronous and immediate. Email is arguably the most influential and widely used application in existence. However, the technical community is only beginning to understand dynamics of its use.

In this thesis, we measure the flow and frequency of user email toward the identification of communities of interest. A *community of interest* (COI) is a set of entities that share a common bond [9]. These sets can be of interest for various group studies. COIs have been studied in systems such as the telephone system and computer networks [1, 9, 25]. These studies can provide highly utilitarian results. For example, COIs can be used to identify normal communication in end-user hosts and servers. Such techniques were shown to effectively suppress worm behavior within a LAN when used to automatically generate host-level firewall rules [25]. There are also a number of potential applications of COI in email. COIs have an obvious application to the problem of automated email organization, where online services prioritize and categorize incoming email as it arrives – thus aiding in the increasingly intractable problem of dealing with huge bodies of incoming email. Applications to spam filters, virus detection, workflow management, HCI, and sociology also exist.

In addition to these applications, another possible application of COI is to determine domain reputations for the DomainKeys Identified Mail (DKIM) service. We present a discussion of DKIM, propose a possible COI-based reputation service, analyze its ability to identify connections between users and domains and evaluate it based on its characteristics and the characteristics of current reputation services.

This is preliminary work and is the first to apply COIs to the characterization of email. We construct COIs by measuring features of past email traffic. The volume, directionality, and frequency of the email traffic are used to determine association between members. We build algorithms based on these email traffic features to determine the members in a COI. The algorithms are analyzed and validated over a large corpus of university email (more than 3 million messages spanning 4 months) by assessing their ability to predict the priority of email as indicated by the recipients. In addition, we examine how the relationships between email users may indicate further information about their COIs through transitive connections.

Unlike other characterizations based on content or external information [7, 24, 27, 26, 34, 6], e.g., email subject, body, address books, the only inputs to our algorithms are the email volume and frequency. That is, the algorithms develop each COI based solely on the senders, recipients, and features of email traffic. This departure from traditional email analysis is significant. It hypothesizes that *email traffic flow alone is highly reflective of social activities, and that those activities can be accurately modeled using the features of the traffic*.

We further found that our COI detection algorithms could correctly identify email priority with greater than 90% accuracy. This supports our hypothesis: COIs based on to whom, when and how frequently a user sends and receives email are highly reflective of their social connections.

We begin by first examining possible applications of COIs in email. We then present related work in Section 3. In Section 4.1 we study the underlying characteristics of a large set of email traffic. We then present measurement and analysis of these characteristics, namely volume and frequency. Section 4.2 defines algorithms to determine COIs based on the results of the analysis in Section 4.1. These algorithms are evaluated and validated in Section 4.3. Section 5 presents DKIM and a description and analysis of a COI-based reputation service. The work is concluded in Section 6.

Chapter 2

Applications of Email COI

Communities of interest have shown to be highly applicable in both phone and data networks. Communities of interest in phone networks lead to the identification of account delinquents and criminal accomplices [9]. In data networks, COIs were used as part of a security mechanism by detecting anomalous behavior and automatically setting firewall rules [25]. Due to the benefits seen in these areas, we believe that COIs will be equally beneficial in an email environment. Here we examine possible applications of identified communities of interest within email.

2.0.1 Email Filtering

Due in part to the overwhelming bombardment of spam [4, 10], spam filtering has been at the forefront of email research. Current email filters [31, 32] have been developed based on widely distributed blacklists [33], whitelisting, and content analysis. Social network dynamics have also been beneficial in improving spam filters [7]. While social dynamic based-approaches have weaknesses as stand-alone spam filters, our technique of identifying COIs could provide social dynamic characteristics to assist an existing spam filter. We leave the implementation and testing of such a system to future work.

Generic email filtering includes the automatic organizing of incoming email based on some user-identified feature [26]. This filtering can use content information, address books,

communication patterns, etc. to determine how to classify an email. Similar to the proposed use in spam filtering, information about COIs could also be used as input to these generic email filters. COIs provide an alternative mode of classifying activity to inference methodologies relying on naive Bayes classifiers, e.g., [15].

Automatic email prioritizing is one popular type of email filter. The ability to automatically sort email based on its priority level can significantly reduce the amount of time a user spends manually sorting through emails. We examine the application of COIs to priority-based email filtering in this paper. We believe that this application will best test the usefulness of COI within filters.

2.0.2 Guilt by Association

Identifying communities of interest within the telephone network proved highly applicable and beneficial in identifying fraudulent accounts [9]. Examination of the COI of a known fraudulent account could, with high probability, determine other fraudulent accounts as fraudulent users typically associate with each other. COIs in email could be used to determine similar behavior. For example, organizations could use COIs to identify accomplices in unauthorized behavior.

2.0.3 Malicious Email Identification

The application of COIs could also prove beneficial because of their ability to link users with relating email patterns. This can aid preventative and reactive measures such as those proposed by Stolfo et al. [34], and improve the forensic analysis of these events to identify the source of a virus and infected groups. We consider clustering methods for COI identification in

a similar manner to the methodologies employed by epidemiological researchers investigating the transmission of infection vectors, e.g., [30].

2.0.4 Automatic Group Generation

Identifying email COIs can also assist in automatic email mailing list generation. Currently, email groups or lists are generated statically. Because COIs automatically identify users associations, they could be applied to automatically generate lists of users that may need to be on a given email distribution list. This automatic generation could greatly reduce the work of a system administrator or other list manager.

2.0.5 Domain Reputation Service

Understanding COIs in email could assist in determining the trust of sending domains. This type of application is in high demand due to the recent push towards ”domain verification” in order to eliminate spoofing. We present an in-depth discussion and analysis of this COI application in Section 5.

Chapter 3

Related Work

Spam, the mass distribution of unsolicited email, has become a significant problem, wasting the time, money, and patience of ISPs and recipients. Various methods are being used to combat the growing quantity of spam. From a legal standpoint, known spammers are being arrested and prosecuted and new laws restricting the sending of spam are being created [10]. The option of restructuring the email system by adding certain constraints or payments has also been investigated. However, these legislative and restructuring methods take time to make a difference in the amount of spam a user sees daily. Thus, short term technical solutions have become quite popular to alleviate the daily influx of spam.

Email filters are the main defense against spam today. Filters take an email, or some part of email traffic, as input and, based on a certain strategy, determine what should be done with the email. There are a variety of known strategies that filters use to make their decision. The number of false negatives, spam messages that are determined to be legitimate emails, and false positives, legitimate emails determined to be spam messages, are used to measure the accuracy of a given filter strategy. Any filter can consist of one or more strategies, but there are a few basic strategies that are the basis of most of the filters used today.

Content based filters look at the subject line and, possibly, the body of an email for suspicious content to determine if it is spam or not. Suspicious content includes odd punctuation, words, phrases, or formatting that are not used in normal conversations. Content based filters can

easily be updated to catch any new forms of suspicious content. However, spammers are able to change the content of the emails they send just as easily which leads to the need for frequent updates of the filter.

Rule-based scoring filters are very similar to content based filters because they examine textual content of emails to determine if a given email is spam. However, they increase the accuracy of content based filters by assigning a point value to a given part of the content. An email is determined to be spam if its point value is greater than a specified threshold [28]. SpamAssassin uses a rule-based scoring strategy as the basis of its filter [31]. Even though the rule-based scoring filters improve on the accuracy of content based filters, they still have the same need for frequent updates to the filter, point values, and threshold.

Bayesian filters are trained to determine what is spam and what is not. They use the attributes from this training data to determine, with a given degree of confidence, whether future emails are legitimate or spam [29]. Some Bayesian filters continue to adapt to changes in the characteristics of email throughout use. This eliminates the need for the frequent filter updates, thus addressing the problem faced by content based filters. The Bayesian filter's down side is that it is limited by its training data in that it is only as good as the data with which it is trained.

Whitelisting and blacklisting are stringent techniques used in order to allow or disallow email from specified addresses. These techniques consist of keeping updated lists of allowed or blocked addresses, thus they have very little computational overhead and are easy to implement [28]. However, whitlelisting can lead to a large number of false positives while blacklisting leads to a large number of false negatives because of the difficulty of creating exhaustive lists of qualifying email addresses. Realtime blackhole lists are publicly maintained lists of known spammers which makes a blacklist more accurate, but this is still an insufficient spam filter by

itself [4]. Whitelists and blacklists are used in conjunction with other spam filtering strategies in order to easily eliminate known spam.

Challenge/response filters improve on the idea of the whitelist by sending any unknown email sender a challenge. If the receiver gets a response back from the sender, it adds the sender to the whitelist [28]. This helps by automatically updating the whitelist and by reducing the number of false positives in whitelisting, but challenge/response strategies take the time of both the sender and the receiver. This can be quite undesirable when receiving mail from a large number of legitimate senders.

Collaborative spam filtering is a fairly new technique that has become quite popular with large communities of email users. Each user determines which of their received emails are spam and the ”votes” are stored in a central email server. By collecting votes among a large group of users with common interests and using some type of spam filter, the filter can be adjusted based on which emails or types of emails a community deems as spam [8]. This method can be troublesome due to the fact that you need a large group of email users for it to be effective and not all of the users may agree on what is and what is not spam.

In addition to spam, there are a number of additional email management issues that exist today including automated prioritizing and email virus throttling. Understanding the social dynamics within the email communication medium can help in addressing these issues. Social dynamics have been studied in a variety of communication networks in order to gain a deeper understanding about the activity within the network; recent studies have focused on the telephone network [9], the Internet [21, 20], and data networks [1, 25]. Communities of interest and their applications have been a focus of many of these social dynamic studies. Our work has been the first to examine communities of interest within email networks.

A number of email-specific applications use elements of social network dynamics to improve their functionality. The use of these elements has been studied in spam filters [7], email classification [24, 26], and virus identification [34, 27]. While individual elements were chosen in these works on an application-specific basis, we have presented a general model of email network dynamics that can be used in a variety of applications. Additionally, past research has drawn on numerous information sources related to email traffic, including address books [27], email headers [7], message content [24], and users themselves [26, 34, 6, 12]. Unfortunately, most of these information sources are often difficult to obtain or store. Email log files, which we and other studies use [7, 34], are the most concise and easily accessible source of information about email traffic. Using only email log data for building COIs simplifies their development and deployment.

Clustering users based on email logs has been previously considered. Gomes et al. [16] uses clusters based on structural similarity of senders and recipients, rather than the communication connections that we employ. Additionally, Gomes et al. focus on spam detection. Tyler et al. [35] uses clusters in order to determine organizational communication characteristics and cluster leaders. Our work differs significantly as we present a more general cluster-based model and consider the temporal nature of email.

Chapter 4

Determining Communities of Interest in Email

4.1 Data Evaluation

In order to define a community of interest for an email user, we need to know with whom they associate. In this analysis, we want to find what information email traffic reveals about the associativity of email users. Specifically, we want to understand what the volume, direction, and frequency of email reveal about the association of email users. We analyze one month of email traffic data in order to gain information about these email features. Latter sections use the results from this analysis to determine email COIs.

4.1.1 Source Data

The data set used in our analysis is from the Computer Science and Engineering Department at Penn State. This network consists of nearly 3000 email accounts with diverse usage habits. The majority of user accounts actively send and receive emails on a daily basis. Over the course of four months, the log files captured more than 3 million emails. The information in the server email files consists of a unique message ID, *to* and *from* email addresses, a timestamp, and host names and addresses. The data contains no information about the subject or contents of the email. In order to preserve privacy, every email and host address was anonymized through a one way keyed hash. The anonymized email log file data was then pre-processed such that a sender, receiver, and timestamp identified each unique email.

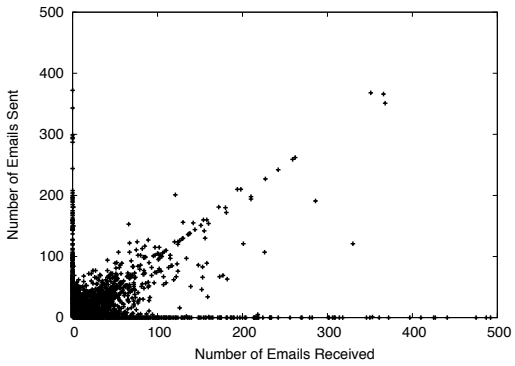


Fig. 4.1. Email communications between users

4.1.1.1 Outliers

After an initial evaluation of our data set, we found that there was some email activity that was atypical for a normal email user. Figure 4.1 shows a graph of the number of emails sent versus the number of emails received for all communicating pairs of email users. This graph indicates that there are some email users that communicate excessively with one other email user. Our research is being conducted based on typical email users such that the application of COIs can be beneficial to them. Much of the email activity seen in Figure 4.1 is fundamentally different from a typical user's email activity. This kind of activity is due to system admin emails, automated tools using tripwire, etc. The utility of these emails is very different from that of typical user emails and we are not concerned with characterizing this type of email activity. Thus, we remove this outlying data in order to gain a more complete understanding of COIs based on typical email user activity. The removed outliers comprise less than 0.5% of our overall data set.

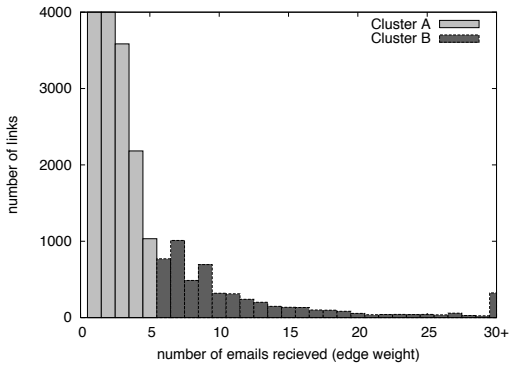


Fig. 4.2. k-means Clusterings of Inbound weights

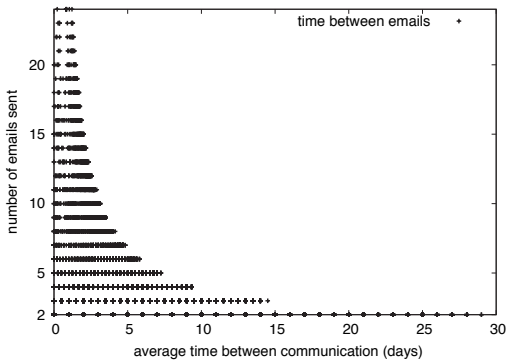


Fig. 4.3. Email Volume and Frequency

4.1.2 Email Volume

An analysis of our data reveals characteristics of email traffic that indicate attributes of an association among email users. The number of communications between two email users can be used to determine the existence of an association. For example, a large number of communications indicates an association whereas a fewer number of communications does not.

Our goal is to determine what volume of received emails indicates an association between a receiver and the sender. We determine this value through a partitioning of communication volumes: one partition includes values that indicate an association and the other partition includes volumes that do not. In order to create this partition, we perform k-means clustering on the number of emails sent between email users in our set of email traffic. This clustering groups communication volumes into k partitions such that the values in each partition have minimal variance from the other values within the partition. Because we are determining inclusion in a set, we use $k = 2$. The clustering is performed on inbound email volumes and separately on outbound email volumes. Our hypothesis is that the result from this clustering will indicate

the partition in the data between associative and non associative communication volumes. This hypothesis is confirmed in Section 4.3.

We performed clustering on the inbound communication volumes. The clustering was performed on three separate months of data. The sum of distances was minimized within each partition, and each clustering was repeated 100 times. We found that, in each of our tests, the k-means clustering of the inbound links arrived at the same partition of the data. Thus, because our analysis resulted in the same partition in three sets of non-intersecting data, we conclude that this is a specific characteristic of these data sets.

The resulting partition of the inbound communication volumes is depicted in Figure 4.2. The x-axis labels are the weights, or number of emails from one user to another. The number of pairs of users that share a given communication volume within our email traffic data are shown on the y-axis. Here we see that cluster A includes communication volumes with inbound emails from 1 to 5, while cluster B includes those with inbound emails greater than or equal to 6. Our intuition indicates that cluster B is a partition of the inbound email volumes which indicate association between nodes. Thus, our clustering indicates that 6 inbound emails is enough to indicate an association between two email users over the course of a month.

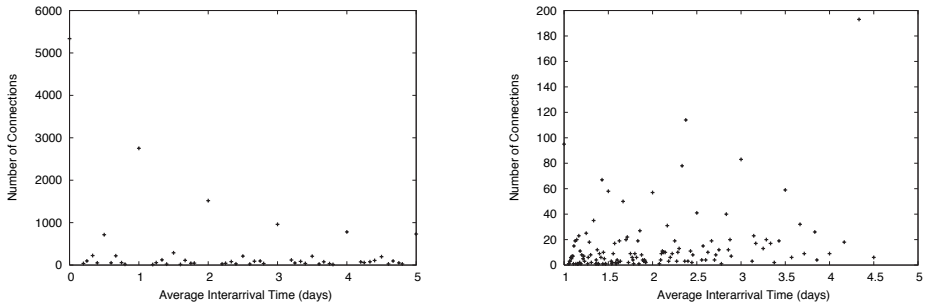
Outbound email traffic differs from inbound traffic. The number of users to which another user sends email is most often far less than the number of users from which he receives email. This fewer number of data points causes difficulty in clustering. When performing clustering the outbound email volumes, the results never stabilized around one specific partition. Thus, our tools were unable to converge on a specific partition of the data. This led us to examine the characteristics of outbound traffic. If an email user sends an email, he obviously has some common bond or interest with the recipient in that communication. By definition, user

association is indicated by communication between a sender and a receiver which indicates a shared bond. Thus, one outbound email is sufficient in indicating a relationship with a contact. This conclusion explains our inability to find clusters in outbound email traffic.

4.1.3 Email Frequency

We now investigate frequency as a determinant of an association. We measured the average interarrival time for received email for each email recipient from a given sender (connection). Due to the fact that 1 sent email indicates association, there is no need to evaluate the frequency of sent emails. This measurement was performed on one month of data[1].

Figure 4.3 shows the average interarrival time against the volume of email received by that user. In Section 4.1.2, we saw that 6 received emails within one month was the smallest number that would indicate association. In Figure 4.3, we see that the maximum frequency used for sending 6 emails is less than 6. This seems intuitive, as there are 30 days in a month.



(a) Interarrival Times, Volume < 6 (b) Interarrival Times, Volume ≥ 6

Fig. 4.4. Interarrival Times for Association and Non-Association Volume

[1]The experiment was repeated over the same three months of data on which volume clustering was performed. Each experiment resulted in nearly identical results.

To further study these interarrival times, we examined them separately; interarrival times of email volumes which indicate association (≥ 6), and those which do not (< 6). Figure 4.4 shows the graphs of the number of connections with a given interarrival time against the interarrival times. Comparing Figures 4.4(a) and 4.4(b), we see that Figure 4.4(b) clearly depicts the short interarrival times, or bursty nature, of email volumes which indicate association. Although this could be attributed to the large volume of the emails sent during a month, there is activity that cannot be explained by that alone. Thus, an association between two email users may be identified by email frequency. Because of this, we can look to frequency as an additional factor in determining association and thus, COI.

In summary, the measurement and analysis presented in this section indicates some fundamental characteristics of the associations as indicated by email traffic:

- Outbound email traffic is more indicative of an association than inbound traffic.
- Large email volume is indicative of an association.
- Frequent email is indicative of an association.

The measurements also indicate some initial results:

- 6 inbound emails indicate association
- 1 outbound email indicates association
- An email received within 5 days of a previously received email from the same sender indicates association

The algorithms defined in the following section are based on the resulting characteristics.

4.2 COI Detection Algorithms

In this section, we develop three algorithms of increasing complexity for determining COI: a *basic* algorithm based solely on communication volume and direction, a *frequency-based* algorithm that is sensitive to the frequency of email traffic, and a *decaying frequency-based* algorithm where COIs place priority on recent communications. The methods used to create the algorithms presented in this section are modeled after those in previous COI work [1, 9, 25].

The algorithms are each based on a value $C_{(a,b)}$ which captures characteristics of communication from one email user to another. This connection value would indicate to user a the "value of connection" with another user b. This value is adjusted based on email traffic flows. These adjustments are defined by the algorithms. If a connection value is above a certain threshold, τ, also defined within the algorithms, then user b is considered part of user a's COI.

4.2.1 Basic Algorithm

Our basic algorithm determines a COI solely on email volume. Each connection value, $C_{(a,b)}$, is adjusted every time there is email activity between user a and user b.

$$C_{(a,b)} = \begin{cases} C_{(a,b)} + 1 & \text{if email is received,} \\ C_{(a,b)} + \lambda & \text{if email is sent.} \end{cases} \tag{4.1}$$

We introduce the parameter λ as a way of weighting outbound and inbound emails. The value of λ is a ratio of the weight of outbound emails to inbound emails. If $\lambda > 1$, outbound

emails are more indicative of COI membership whereas if $\lambda < 1$ inbound emails are more indicative of COI membership. Based on our analysis in Section 4.1.2 sent emails should increase the value more than received emails, thus $\lambda > 1$.

In our basic algorithm, the threshold for determining COI membership, τ, should be the same as the number of received email required to indicate membership. This value is based on Equation 4.1 where reception of email results in a constant value increase.

The basic algorithm assigns weights to the edges independently of time. Thus, if user a receives 6 emails from user b over the course of a year, the weight on that edge assigned by the basic algorithm would be the same if those messages were received over the course of one day.

4.2.2 Frequency-Based Algorithm

To address effects of the frequency of email communication, we introduce a new algorithm that is dependent upon interarrival times (times between received emails). This algorithm determines COI membership based on outbound and inbound emails in a manner similar to the basic algorithm.

In order to define our connection values as dependent on time, we must consider how time affects the weight of email messages that are sent and received. If we refer back to Section 4.1.3 we note a contact should be included in a COI an email is received within 5 days of the previous received email. This will serve as the basis for the initial construction of our new connection values.

First we discuss the effect of time on outbound emails. In Section 4.1.2 we found that one sent email was enough to determine COI membership. Thus, send frequency is not a factor when considering sent emails.

When considering received emails, frequency is a factor. Based on the analysis in Section 4.1.3, the shorter the interarrival times, the greater the increase of the connection value should be. If user a receives an email from user b more than once every 5 days, the contribution to its weight should be more than 1. Accordingly, if user a receives emails from user b less often than every 5 days, its contribution should be less than 1. This value is denoted by μ.

$$C_{(a,b)} = \begin{cases} C_{(a,b)} + \mu & \text{if email is received,} \\ C_{(a,b)} + \lambda & \text{if email is sent.} \end{cases} \tag{4.2}$$

Both the frequency-based algorithm and the basic algorithm weight the graph edges with a monotonically increasing function. This implies that once a contact enters a user's COI, he will never be removed from it.

4.2.3 Decaying Frequency-Based Algorithm

A monotonically increasing COI may not be realistic when evaluating a user's email communications. In order to address this issue of permanent COI membership, we introduce a modification to the frequency-based algorithm. We extend the frequency-based algorithm such that the connection values decay over periods of inactivity. Our decay function will decrease the current connection value at the turn of each day with the following restrictions: 1) if the connection value is less than $\tau - 1$, then it should not decrease, and 2) for a connection value larger than $\tau - 1$, it should decrease faster for larger values. The reason for our first restriction is so a user that was once a part of a COI will not be completely forgotten and can be quickly reintroduced into that COI. Our second restriction ensures that someone in a COI with a small connection value will not be ejected too quickly and those with a large connection value will not

be forever in the COI. If $C_{(a,b)}$ is the current connection value, then at the increment of each day, we apply the following decay function

$$C_{(a,b)} = \begin{cases} C_{(a,b)} & \text{if } 0 \leq C_{(a,b)} \leq \tau - 1, \\ C_{(a,b)} - \frac{C_{(a,b)} - (\tau - 1)}{\delta} & \text{otherwise.} \end{cases} \tag{4.3}$$

We introduce δ as our decay coefficient, which can be varied depending on the purpose of the algorithm. It should be noted that the two parameters λ and δ will affect the speed at which a member of a COI will be removed. A larger value of δ would allow a user in a COI to stay longer without any communication, while a smaller value would eject the user after a shorter period of time. While our analysis from Section 4.1.2 implies that $\lambda \geq \tau$, it should be noted that an even larger value of λ will allow a user in a COI to stay longer without any communication. In Section 4.3, we examine the behavior of COIs under different values of δ.

4.2.4 Effect of Transitive COIs

The algorithms in the previous sections developed individual COIs in isolation from the knowledge of external COIs. We propose an extension to our algorithms such that COIs exploit transitivity. COI transitivity involves sharing COIs among associated nodes. For example, two users may not share a direct association; however, they may be indirectly linked through members of their individual COIs. We model transitivity by including COI "neighborhoods", e.g., transitivity including the COI members up to n hops away. We believe that this extra inclusion will result in more accurate identification of COI members due to the natural commonalities shared between users with associations. An evaluation of this extension is presented in Section 4.3.

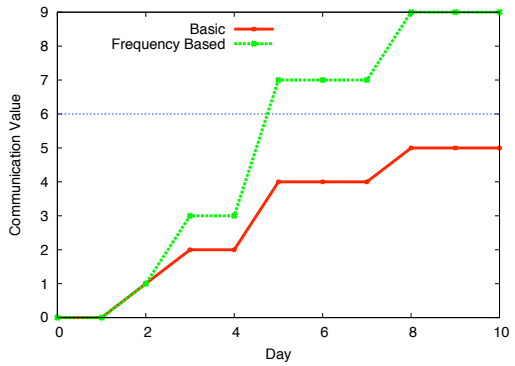


Fig. 4.5. Communication Values(τ=6)

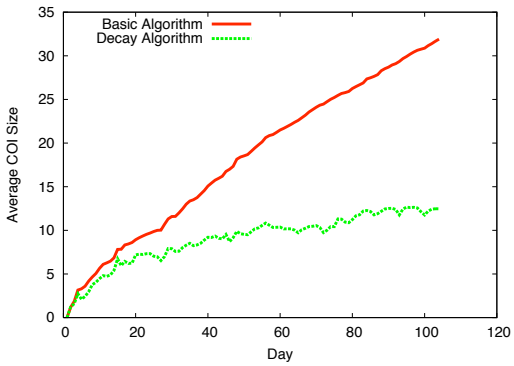


Fig. 4.6. Average COI Size

4.3 Algorithm Validation

An email user's community of interest consists of members with whom the user shares a common bond. Based on our definition, the emails sent between an email user and his COI members should be of interest to the user. The amount of interest that a receiver has in an email is commonly indicated by priority. Thus, the emails received from the members of a user's COI should be high priority emails. In order to validate the usefulness of our algorithms, we test their ability to correctly identify the senders of high priority email. We then present overall results from our email COI research.

4.3.1 Validation Data

Information about the priority of an email is determined solely by the receiver. Thus, in order to gain information about email priority, we performed a user study. Fifteen volunteers from our email network collected all of their received email for one month, which encompassed approximately 9,000 emails. At the end of the month, the volunteers labeled their data based

on high and low priority. The volunteers were not told how the information was used. This training data was then anonymized with the same one way keyed hash as the original server log data. These prioritized messages were then integrated with the server log data by labeling the messages with their assigned priority level.

We performed the experiments by considering each email received as a sequential trace, allowing the COI algorithms to incrementally update communication values. Volunteers prioritized a subset of the three million emails; both training and test data were used to create communication values and test the correctness of classifications made. This trace-based method allowed testing data to be updated concurrently with other training data, presenting us with the opportunity to observe the evolution of the resulting models. Emails belonging to the training subset (i.e., those labeled with a priority) were tested for inclusion in the COI, then used to update communication values.

During the testing, we measured four values: true positives, true negatives, false positives, and false negatives. True positives are high priority emails where the algorithm recognized the sender of the email as being included in the receiver's COI. True negatives are low priority emails that the algorithm did not recognize the sender of the email as being included in the receiver's COI. False positives occurred when the sender of low priority email is incorrectly recognized as a member of the receiver's COI, and false negatives occur when the sender of high priority email is incorrectly recognized as not being in receiver's COI.

We are most concerned about the ability of our algorithms to correctly identify the senders of high priority email thus, false negatives are the worst kind of false classification. We evaluate two statistics to highlight these results: correct identification of high priority email

senders (HIGH) and correct overall identification of both high and low priority senders (OVERALL).

4.3.2 Results

Our testing revealed that the COI algorithms were capable of successfully determining the priority of an email by its sender for over 90% of both high priority email (HIGH) and the full corpus of prioritized email (OVERALL). Table 4.1 shows the percentage of email correctly classified by our basic and frequency-based algorithms. The numbers in bold represent the percentage when the parameters used are those obtained from our analysis in Section 4.1: $\lambda = 6$ and $\tau = 6$.

Table 4.1 shows that our algorithms yield one of the highest percentages of correct overall and high priority classifications when $\lambda = 6$. This implies that if a receives 6 emails from b in a month, then b belongs in a's COI. It should also be noted that for larger values of λ our algorithm does not classify as well, implying that both sent and received emails contribute to association.

We also see that $\lambda = \tau$ yields one of the highest percentages of correct classifications. This would imply that a single email from user a to user b represents an inclusion in a COI. This observation suggests that our intuition of the association strength of sent mail was correct. Sent emails are a stronger indication of an association than received emails.

Figure 4.5 compares the basic and frequency-based algorithms for a given user. The y-axis represents the connection value of a given user a to another user. We see that our frequency-based algorithm introduces the user into a's COI faster than our basic algorithm. The results from the validation of frequency-based algorithm are shown in Tables 1(c) and 1(d). This suggests that introducing an algorithm dependent on time improves the classification of email, validating our

claims that time is an important criterion for evaluating email communication. Because the frequency-based results improve over the results from the basic algorithm, we can conclude that frequent emails are highly likely to indicate an association.

We further hypothesized in Section 4.2 that the COI of a user changes over time. To examine this change in COI, we implemented our basic algorithm on consecutive months of data and compared the COIs of each month. Our calculation showed that over two consecutive months only 25% of users were maintained in a COI. When comparing COIs calculated for months that were 4 months apart, only 17% of the users in a COI were the same. These numbers indicate that while our basic algorithm correctly adds members to a COI, the members are not indicative of the *current* COI because members of COIs enter and exit. Our decaying frequency-based algorithm captures this activity. This analysis indicates that COI membership is not stable over time.

In order to test our decaying frequency-based algorithm, we fixed $\lambda = 6$ and $\tau = 6$ to be consistent with the frequency-based algorithm tests. However, we test different decay parameters. A value of $\delta = 2$ will eject a current member of a COI with a low connection value after a 2 or 3 days, while $\delta = 7$ would eject after one week and $\delta = 30$ would eject after one month. For $\delta \geq 2$, our decay algorithm had the same success percentages as the frequency-based algorithm.

Figure 4.6 shows the average size of a COI over 100 days when using the basic algorithm and the decay algorithm (a decay coefficient of $\delta = 7$ was used). Using the basic algorithm, the average user in our graph had over 30 users in its COI. Using the decay algorithm, we saw that after 75 days the average COI size was only 12. Thus, even though a decay algorithm reduces

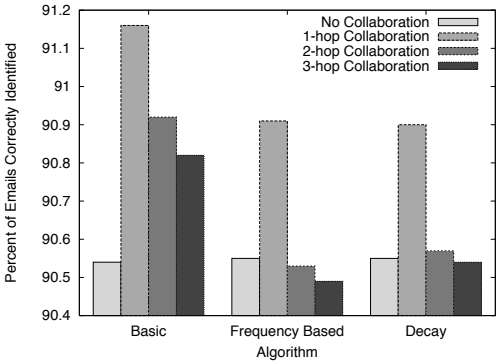


Fig. 4.7. Transitive COI Results

the number of COI members, it is still able to classify the priority of an email with significant accuracy. This indicates that users exit COIs frequently.

In Figure 4.7 we see that using transitive COIs improves the ability of the algorithms to correctly classify email by priority level. This validates our hypothesis that transitivity of COIs plays a significant part in classifying senders correctly. However, this increase in performance is only seen with one-hop collaboration. Extending the transitivity past a node's immediate neighbors results in a higher number of false positives, thus degrading performance. This implies that COI associations are transitive but only in restricted ways, i.e., a single hop.

4.3.3 Sensitivity Analysis

We validated our algorithms on one large set of data. We note in Section 4.2 that the parameters (λ, τ, and δ) can be changed for any data set from a certain domain or email server; a k-means analysis can be run to determine where the natural partition occurs in the data and δ can be chosen such that it would would model the desired decay function for the given domain.

However, because we were only able to validate our algorithms on one set of data, we were unable to test their robustness on other sets of data. Thus, we performed a sensitivity analysis to demonstrate the flexibility of our algorithms with different parameters over our data set. Because λ and τ are not mutually independent, we must perform the sensitivity analysis of both parameters simultaneously. Table 4.1 shows the parameters λ and τ varied from 2 to 10 and all show all combinations. As we stated in our results, we see that $\lambda = 6$ and $\tau = 6$ yields one of the highest percentages. We note that varying the parameters slightly would also yield high percentages. Thus, our algorithms are able to correctly classify email despite small variations in the parameters. This result leads us to believe that they will prove effective with other data sets.

In addition to λ and τ, we also experimented with the choice of the value of δ in our decay algorithm. The value δ is independent of the other parameters, thus its sensitivity analysis can be performed independently of λ and τ. Changing the value of δ should not affect the ability of our algorithms to correctly classify data. This is due to the rule of the decay algorithm: *old COI members should not be forgotten*. While the decay algorithm reduces the number of members in a COI to portray a current snapshot, it is able to quickly adjust when contacts re-introduce themselves. Thus, varying the decay parameter between 2 and 30, we still see the exact same results in every trial as we see in the frequency based algorithm. Based on the results of the change of the parameter δ in the decay algorithm, we conclude that it is will also be flexible and effective with other data sets.

4.3.4 Discussion

The methodology that we chose to validate our COI models had an inherent limitation. We used our models to prioritize email however, due to the personal and content components

of email, classifying email solely by traffic patterns (sender, quantity, and frequency) cannot capture the priority of an email. The semantics of an email lend a lot to its priority. For example, an email from a family member may notify the recipient of a family emergency while a different email from that same family member may be a chain letter. As discussed in the introduction, characterizations of email based on content, subject, etc. have been widely studied.

Despite only using traffic patterns to prioritize email, our validation results in $\sim$91% accuracy. The 9% of incorrectly classified email could very easily be contributed to missing semantics of an email. Thus, traffic patterns alone do indicate significant information about the characteristics of an email. This is a good indication of the usefulness of COIs as an additional tool to be used in prioritizing filters or other applications as described in Section 2.

(a) Basic High Results

$\tau \backslash \lambda$	2	4	6	8	10
2	0.9407	0.9407	0.9407	0.9407	0.9407
4	0.8983	0.9106	0.9106	0.9106	0.9106
6	0.8588	0.8829	**0.8952**	0.8952	0.8952
8	0.8296	0.8504	0.8728	0.8852	0.8852
10	0.8040	0.8371	0.8441	0.8674	0.8789

(b) Basic Overall Results

$\tau \backslash \lambda$	2	4	6	8	10
2	0.8948	0.8948	0.8948	0.8948	0.8948
4	0.8994	0.9055	0.9055	0.9055	0.9055
6	0.8879	0.8993	**0.9054**	0.9054	0.9054
8	0.8786	0.8887	0.8993	0.9054	0.9054
10	0.8695	0.8854	0.8888	0.8997	0.9055

(c) Frequency Based High Results

$\tau \backslash \lambda$	2	4	6	8	10
2	0.9407	0.9407	0.9407	0.9407	0.9407
4	0.9216	0.9216	0.9216	0.9216	0.9216
6	0.8983	0.9106	**0.9106**	0.9106	0.9106
8	0.8735	0.8906	0.9029	0.9029	0.9029
10	0.8586	0.8801	0.8829	0.8952	0.8952

(d) Frequency Based Overall Results

$\tau \backslash \lambda$	2	4	6	8	10
2	0.8948	0.8948	0.8948	0.8948	0.8948
4	0.9036	0.9036	0.9036	0.9036	0.9036
6	0.8994	0.9055	**0.9055**	0.9055	0.9055
8	0.8914	0.8996	0.9057	0.9057	0.9057
10	0.8878	0.8980	0.8993	0.9054	0.9054

Table 4.1. Results of Algorithm Sensitivity Analysis

Chapter 5

Supporting DKIM Domain Reputation with Communities of Interest

In this chapter we provide an introduction to the DomainKeys Identified Mail (DKIM) service and a brief discussion of the challenges facing the technology. In this discussion, we examine *domain reputation services*, why they are important to DKIM, and how our COI tool can be used in such a service.

5.0.5 Introduction to DKIM

Email spoofing is the typically malicious modification of email header fields used to make receivers believe that an email came from a different sender. Commonly used by spammers, email spoofing has made email classification incredibly difficult and plays a major part in devastating phishing attacks. There have been a number of proposed solutions for the spoofing problem including SPF [36], Sender ID [23], CSV [11], and DKIM which we focus on here.

DKIM was released as a standard in May 2007, however its benefits will will not be seen until it is adopted by users. Currently, there are a number of developers that are including DKIM signing and verifying services in their products [13] and a number of ISPs, email providers, and companies signing emails and verifying DKIM signatures [14, 5].

DKIM [2, 22] attempts to address the email spoofing problem by attaching cryptographic signatures to emails that can be used to verify the integrity and the originating sender domain of the email. When the sending domain's server receives an outbound email, it creates a hash of

the email (including some headers fields[1]), signs the hash with the domain's private DKIM key, and attaches the signed hash to the message. Upon receiving a signed email, a verifying server retrieves the sending domain's private key from a DNS server, hashes the email message, and then verifies the attached signature.

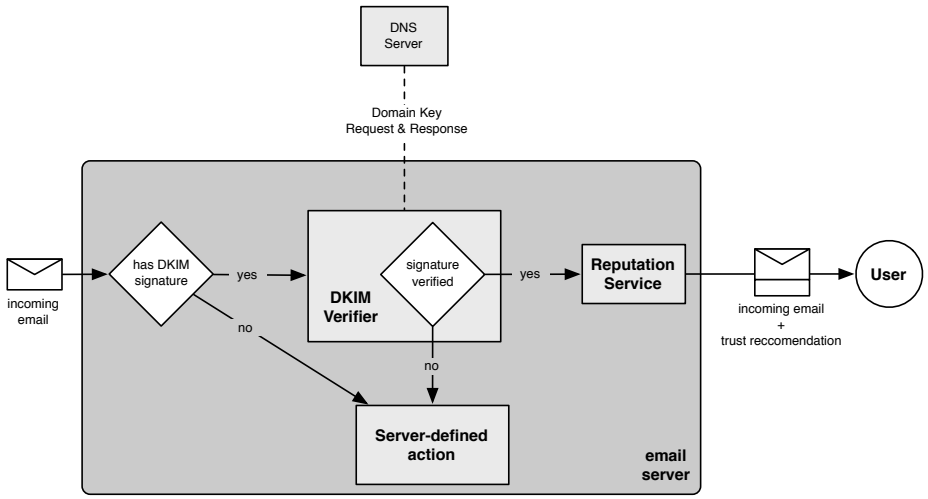


Fig. 5.1. Email server implementing DKIM verification

Unfortunately, DKIM does not completely solve the problems of spam or phishing. Alone, DKIM can be used to verify the sending domain and the integrity of an email. But once this verification passes or fails, it is up to the receiving mail agent to determine what to do with the email. This issue has been the focus of much of the current research associated with DKIM. Here we describe some of the situations that a verifying mail agent may face when it receives an email:

[1]Not all headers fields will be signed by the sending domain. Because header fields are the part of an email most likely to be altered in transit, it is best to limit the signed headers to the minimum necessary set, e.g. `from:`, `to:`, `subject:`.

1. **A message arrives unsigned:** If a message arrives unsigned, it obviously means that neither the origin nor the integrity of an email can be verified with DKIM. In this case is seems that the verifying agent would simply handle this email as it handled messages before the deployment of DKIM, e.g. send the email through a spam filter then forward it to the `to:` account. However, there is a complication. The issue of *sender signing processes* [3, 22] has been an important topic throughout the development of DKIM and a standard for these processes is under construction (and much debate). Sender signing processes are ways by which a signing domain can indicate its intentions to a verifying agent. For example, if a signing domain signs every email leaving its domain that it deems valid, that domain can include a message in the email header indicating this to a validating agent. Then, when a validating agent receives such a message, it knows that any email that it receives from that sending domain should be signed. At that point, if an email arrives unsigned from this domain, a verifying agent may actually handle that email much differently that it handles other unsigned emails.

2. **A message arrives signed and there is no key available for the sender:** If a message arrives with a DKIM signature but the verifying agent has no public key for the domain and is unable to retrieve a key from a DNS server one of two things may have happened: 1) the email may have been created maliciously or modified in transit such that the header `from:` domain does not match the key with which the email was signed or 2) the legitimate signer may have not uploaded his public key. Because it is unknown if this email is legitimate or not, the verifying agent does not have a lot of information to make an educated decision about how to handle the email. It could look at any domains' sender

signing processes that it has on file and, if there one matches the sending domain, could make a decision based on that. Otherwise, the handling of this email falls under the same category as an unsigned email.

3. **A message arrives signed, there is a key for the sender, and validation fails:** This situation is much like the previous situation. A validating agent receives an email with a signature, retrieves the sending domain's key from DNS, then attempts to verify the signature. An invalid signature could mean that 1) there was an error in the generation or validation of the signature, 2) a malicious entity is attempting to pass as a valid domain but cannot create the correct signature, or 3) the email has a legitimate signature, but some part or all of the content was altered enroute. In the event of case 1), because the deployment of DKIM is fairly new, the validating agent should verify that his system is working correctly. In case of a mis-generated signature, a DKIM signing domain can include a flag in the DKIM header that indicates that DKIM signing is a new procedure for the domain and if a signature arrives and doesn't validate, to simply dismiss the error. Additionally, the verifying agent should check its list of sender signing practices to see if any information can be found there. In cases 2) and 3), again, there is little that a verifying agent can do next to treating the email as if it were unsigned.

4. **A message arrives signed, there is a key for the sender, and validation succeeds:** In the case that DKIM serves it purpose and a sending domain can actually be validated by a verifying agent, the verifying agent is still in a position of determining how to handle the email. Again, DKIM is not a solution for spam or phishing, it is simply a way to be sure which domain a given email came from. The verifying agent must determine if it actually

trusts email coming from the validated domain. For example, say `citibank.com` is the domain that Citibank uses to send emails to current and potential customers. However, `citi-bank.com` is used by an attacker to launch phishing attacks. Both domains register public keys with DNS. If a verifying agent receives an email from the domains `citibank.com` and `citi-bank.com`, both of which come with valid signatures, how does it determine which one is the phishing attack and which one is a legitimate email? Currently *domain reputation* is a very challenging problem facing DKIM and its benefits. Various ideas have been proposed as *domain reputation services*, but nothing has emerged as a clear-cut solution. This problem is the focus of the remainder of our discussion.

5.0.6 Description of COI-based Reputation Service

Email communities of interest, as discussed in this research, show great promise to serve as a dynamic indicator of an individual's or group's relationships through email. Given these promising results found in our previous experiment, we believe that the use of COI in a DKIM reputation service would be extremely beneficial. In this section, we describe such a service.

A COI-based reputation service would run on an email server behind a DKIM signature verifier. The service would function as depicted in Figure5.2. When a validly signed email arrives, the filter would first update its "database" of user activity as seen in the original COI algorithms. It would then determine if the receiver's COI contains a sender with the same domain as the current email sender. If so, the email could be labeled as trusted or untrusted. This label could then be interpreted by a spam filter or could be displayed to the user through a reputation service plug-in for an email client.

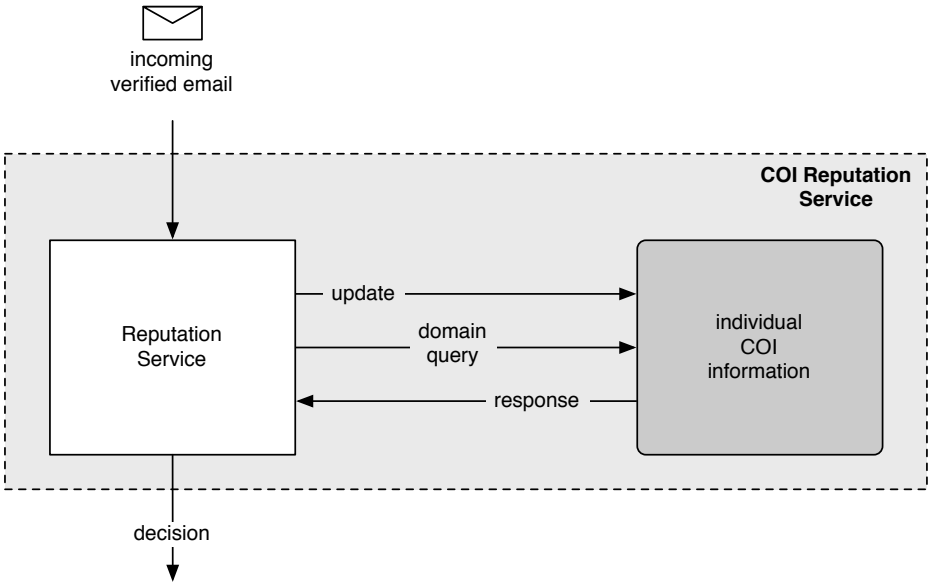


Fig. 5.2. Diagram of a COI-based Reputation Service

The main difference between this application and the priority-filter presented in Section 4.3 is that we are no longer dealing with individual email addresses. We must now examine trust on a domain level. This changes the way that the COI filter should function. However, COIs are based in the interactions between users, so we still depend on that as the basis of any decision.

Similar to the priority filter, COIs for a reputation service would be determined in the same manner. When a user sends or receives an email, his ”COI profile” is updated. However, in this case, we can only trust signed incoming messages as we can no longer be vulnerable to spoofing[2]. Based on the individual COIs, trusted domains can be determined by examining the domains included in the COI.

[2]In this case, all outbound messages are considered, signed or not as they would be handled by the same outgoing signature service.

5.0.7 Analysis

In order to determine the usefulness of COIs in reputation services, we must perform two tasks. First, we must demonstrate a COI-based reputation service's ability to correctly identify "trustable" domains. Second, we should examine the differences between a COI-based service and current reputation solutions to understand the strengths and weaknesses of our method.

5.0.7.1 COI Domain Analysis

In Section 4.3, we showed that COIs provide significant information about the priority of emails. In this analysis, we want to determine if COIs provide similar information about the trust of domains.

Similar to our original experiment, we must have data by which to compare the labeling of our COI-based reputation service. We use the original complete corpus of data as well as the priority labeled data. However, the goal in this experiment is to determine *if a domain can be trusted*. Thus, we use the priority labeled data and relabel it. Emails are relabeled with a the highest priority given to any email sent by the sending domain. For example, if a user receives an email from sender1@domain.com and labels it with a high priority "1" and another email from sender2@domain.com and labels it with a low priority "5", we relabel the second email with a priority of "1". As we are creating a reputation-based service for domains, not senders, if we trust one sender in a domain, then we should trust the entire domain, thus all senders. This relabeling encompasses this idea[3].

[3]There are other cases in which domains are not able be trusted in all cases e.g. Hotmail or gmail where the domain can consist of trusted or untrusted users. This is a problem facing DKIM in general.

Like the validation performed in Section 4.3, we performed this new test by considering each received or sent email as it was processed by the server as determined by our log files. The COI algorithms update each individual's COI values based on each email. In the case of a labeled email, we determined if the label indicated trust in a domain or not. We would then search the receiver's COI for any address with that domain. The labeled email then be classified as a *true positive*, *true negative*, *false positive*, or *false negative* as depicted in Figure 5.3. If an email was labeled as trusted and an address with a matching domain was found in the receiver's COI, the labeled email would be classified as a *true positive*, otherwise it would be classified as a *false negative*. If the email was labeled as untrusted, an email address with a matching domain was found in the receiver's COI, the email would be labeled as a *false positive*, otherwise it would be classified as a *true negative*.

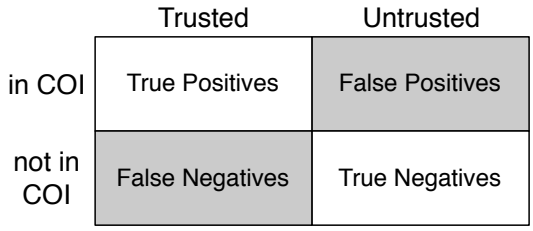


Fig. 5.3. Classification Matrix

After running the tests and classifying all of the labeled emails, we can examine the number of *true positives* and *false negatives* to determine how well a domain-based COI classifies trusted emails. In Table 5.4, we see that the number of correctly classified trusted emails is 4307 resulting in a 94.51% correct classification of trusted emails. This result is very promising as it is better than the result found in Section 4.3 for frequency based high priority. We also see in

Table 5.4 that the number of correctly classified non-trusted emails is 3377 resulting in a 87.83% correct classification of all email. This incorrect classification of untrusted emails is detrimental to our algorithm as we are creating a reputation service. Labeling emails as trusted when they should not be trusted is the worst case in this situation. In this case it would seem that we should simply use the single email based-COI algorithms as originally presented in Section 4.2.

However, our focus in this experiment is the correct classification of domains, not emails. Thus, we examine how well our COI algorithm correctly classifies domains as trusted or non-trusted. These results are presented in Table 5.4 in parentheses. We see that the number of correctly classified trusted domains is 139 resulting in a 47.77% correct classification. This is clearly a less than desirable number. However, we also witness 1710 correctly classified untrusted domains resulting in 91.76% overall correct classification of domains. There are a few clear answers for these results. First, the disappointing result in the trusted domains may be because of the inability to correctly identify a domain based upon seeing it for the first time. Second, there are very few trusted domains when compared to the number of domains that are untrusted. This makes the overall result of 91.76% very impressive. Thus, our domain-based COI algorithm does a respectable job as a reputation service because it can identify untrusted domains so well.

While the algorithm that we present and analyze is not perfect, it have very promising results. In fact, based on sender, receiver and date data alone, it does surprisingly well. With the demand for privacy and increased encrypted emails, the ability to do this analysis on this simple data is crucial. There are also some benefits to having a domain-level reputation service. In the next section, we discuss the benefits of a COI-based reputation service when compared to other plausible solutions.

	Trusted	Untrusted
in COI	4307 (139)	815 (41)
not in COI	250 (152)	3377 (1710)

Fig. 5.4. Results of Domain Algorithm Analysis

5.0.7.2 COI-based Reputation Service Analysis

DKIM is a fairly new idea as it was released as a standard less than a year ago. As with most new Internet technologies, it will take some time to catch on. Currently, some of the major email providers have been adopting DKIM and are actively participating in its development. Unfortunately, some spammers are also picking up the use of DKIM. However, while the spread of its use seems promising at this point, there is very little data available to support an informative experiment. For example, in nearly 7 million emails collected from a medium-sized ISP, about 100,000 messages were signed using DKIM signatures, about 1.5%. Of these signed messages, 60% were marked with test keys indicating that the signing domain is in the beginning phases of signing emails and not to reject their emails if a signature cannot be verified. Examining the domains, most of the signed messages received were from `gmail.com` and the rest resembled marketing or spam organizations.

Based on this available data, a thorough test of a COI-based reputation is not feasible and if it were, it seems as though COIs would only consist of `gmail.com` addresses which would lead to fairly meaningless results. Instead, the analysis of our COI-based reputation service will

focus on its advantages and differences when compared to current reputation services. We examine our proposed solution along with manual whitelisting/blacklisting, Vouch By Reference, Habeas, and SenderBase. A manually managed whitelist or blacklist is a simple approach involve creating a list of domains which you accept or deny emails signed with DKIM signatures. Vouch By Reference [18] is a protocol under development which involves a third party to vouch for a certain domain and the content that they approve to be in the emails that the domain sends. Habeas [17] is a company that offers subscription-based whitelists and blacklists as well as membership on the whitelist based on an evaluation of a domain. They also offer ”non-contractual” whitelists on which domains can be listed if they pass certain ”automated certification tests”. SenderBase [19], a product developed by IronPort, is similar to Haebeas in that it requires a subscription to use the service. However, it makes decisions about the validity of domains based on huge volumes of real time data and customer feedback in addition to various static characteristics that Habeas uses.

We examine these four different reputation services as well as our proposed COI-based reputation service based on four different criteria: how the services are updated, who makes the decision if a domain is trusted or not, whose input is included in the decision and who the decision effects, and the factors on which the decider bases its decision. Each service's characteristics are summarized in Table 5.1.

After examining the differences between the possibilities for DKIM reputation services, certain positives, and negatives are apparent. For example, dynamic update of the given service mechanism would be tremendously easier and perhaps more accurate than a static update. We then see that any of the proposed services would be preferred to the whitelist/blacklist scheme.

Although Habeas uses analysis of certain domains to statically update its reputation service, it also has some automated tools which play a large part and do dynamic updating.

The other apparent differences are the entity that makes the decision as to whether or not a given domain should be trusted, what data is used for a basis to make this decision, who utilizes the decision that is made, and the criteria for making the decision. In the case of VBR, Habeas, and SenderBase, an entity outside of the local domain makes the decisions about the reputation of domains. This could be viewed as a positive in that these services incorporate data from a number of users and verifiers creating a large body of information from which domain reputation can be determined. However, VBR is only able to provide reputation information about users that have subscribed to have their domain vouched for. On the other hand, COI and locally managed whitelists/blacklists make domain reputation decisions locally. While these decisions may not be based on large bodies of data, they may be similarly accurate because they are structured around local activity avoiding problems that may arise when a global decisions is made about a specific domain. Typically, local issues are easier to address than global ones. In addition, the domain reputation decisions made by Habeas and SenderBase are only made available to paid subscribers of their service while the other three presented services are offered for free.

Lastly, the quality of a reputation service lies largely in the basis of the decisions about domain reputation. While each of the presented reputation services use a wide array of information to make its decisions, we believe that because a COI-based reputation service would be very effective because it is based simply on individual relationships.

Service	Update	Decider	Inclusion	Basis of Decision
COI	dynamic	local domain	all sending domains; receiving domain users	individual relationships
Blacklist/ Whitelist	static	local domain	all sending domains; receiving domain users	administrator analysis
VBR	static	third party	subscribed sending domains; receiving domain users	third party knowledge
Habeas	both	Habeas	all sending domains; subscribed receiving domains/users	in-depth domain analysis, "automated certification tests"
SenderBase	dynamic	SenderBase	all sending domains; subscribed receiving domains/users	"global sending volume, complaint levels, spamtrap accounts, whether a senders DNS resolves properly and accepts return mail, country of origin, blacklist information, probability that URLs are appearing as part of a spam or virus attack, open proxy status, use of hijacked IP space, valid and invalid recipients"

Table 5.1. Comparison of Domain Reputation Services

Chapter 6

Conclusion

In this paper we investigated communities of interest and their existence in email. After examining a large body of data, we were able to construct three different algorithms, each increasing in complexity, to identify these communities of interest in email. We were able to validate their usefulness in email through a user-based study examining their potential in priority filtering. With this validation, we were able to determine that the individual relationships identified by our COI algorithms were very useful in identifying high and low priority email.

We furthered our research by looking closely at another possible application for COI in email, DomainKeys Identified Mail (DKIM). After understanding this new technology aimed at identifying spoofed emails, we determined that COIs could be used as part of an effective DKIM reputation service. We presented the idea for such a service and then analyzed such a service and compared it to existing services.

We have shown that communities of interest do exist in email, how to identify them, and that they do have meaningful applications, specifically as a domain reputation service for DKIM. As DKIM is a fairly new technology, we were limited in our ability to rigorously test a COI-based reputation service. As DKIM evolves, we hope to test the described reputation service and implement it for use by local email servers.

References

[1] William Aiello, Charles Kalmanek, Patrick McDaniel, Subhabrata Sen, Oliver Spatscheck, and Jacobus Van der Merwe. Analysis of communities of interest in data networks. *Lecture Notes in Computer Science*, 3431:83–96, 2005.

[2] E. Allman, J. Callas, M. Delany, M. Libbey, J. Fenton, and M. Thomas. DomainKeys Identified Mail (DKIM) Signatures. RFC 4871, Internet Engineering Task Force, 2007.

[3] E. Allman, M. Delany, and J. Fenton. DKIM Sender Signing Practices. Internet-Draft draft-ietf-dkim-ssp-00, Internet Engineering Task Force, 2007.

[4] Eric Allman. Spam, spam, spam, spam, spam, the ftc, and spam. *Queue*, 1(6):62–69, 2003.

[5] BITS. BITS Email Security Toolkit: Protocols and Recommendations for Reducing the Risks. Technical report, BITS: Financial Services Roundtable, 2007.

[6] Danah Boyd and Jeffrey Potter. Social network fragments: an interactive tool for exploring digital social connections. In *GRAPH '03: Proceedings of the SIGGRAPH 2003 conference on Sketches & applications*, pages 1–1, New York, NY, USA, 2003. ACM Press.

[7] P. Oscar Boykin and Vwani P. Roychowdhury. Leveraging social networks to fight spam. *Computer*, 38(4):61–68, 2005.

[8] John Canny. Collaborative filtering with privacy. *IEEE Symposium on Security and Privacy*, 2002.

[9] Corinna Cortes, Daryl Pregibon, and Chris Volinsky. Communities of interest. *Lecture Notes in Computer Science*, 2189:105–114, 2001.

[10] Lorrie Faith Cranor and Brian A. LaMacchia. Spam! *Commun. ACM*, 41(8):74–83, 1998.

[11] D. Crocker, D. Otis, and J. Leslie. Client SMTP Authorization (CSA). Internet-Draft draft-crocker-csv-csa-00, Interent Enginnering Task Force, 2005.

[12] Laura A. Dabbish, Robert E. Kraut, Susan Fussell, and Sara Kiesler. Understanding email use: predicting action on a message. In *CHI '05: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 691–700, New York, NY, USA, 2005. ACM Press.

[13] DKIM.org. DKIM Software and Services Deployment Reports. `http://www.dkim.org/deploy/index.htm`, 2008.

[14] DKIM.org. DKIM User Deployment. `http://www.dkim.org/deploy/users/index.htm`, 2008.

[15] Mark Dredze, Tessa Lau, and Nicholas Kushmerick. Automatically Classifying Email into Activities. In *Proceedings of the 2006 International Conference on Intelligent User Interfaces (IUI'06)*, Sydney, Australia, January 2006.

[16] Luiz H. Gomes, Fernando D. O. Castro, Virg'ilio A. F. Almeida, Jussara M. Almeida, Rodrigo B. Almeida, and Luis M. A. Bettencourt. Improving Spam Detection Based on Structural Similarity. In *Steps to Reducing Unwanted Traffic on the Internet Workshop (SRUTI'05)*, July 2005.

[17] Habeas. The Habeas Network. http://www.habeas.com/, 2008.

[18] P. Hoffman, J. Levine, and A. Hathcock. Vouch By Reference. Internet-Draft draft-hoffman-dac-vbr-02, Interent Enginnering Task Force, 2007.

[19] IronPort. The SenderBase Network. http://www.ironport.com/products/ironport_senderbase_network.html, 2008.

[20] Jon Kleinberg. The small-world phenomenon: an algorithm perspective. In *STOC '00: Proceedings of the thirty-second annual ACM symposium on Theory of computing*, pages 163–170, New York, NY, USA, 2000. ACM Press.

[21] Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, and Andrew Tomkins. The web and social networks. *Computer*, 35(11):32–36, 2002.

[22] Barry Leiba and Jim Fenton. Domainkeys identified mail (dkim): Using digital signatures for domain verification. In *Fourth Conference on Email and Anti-Spam*, 2007.

[23] J. Lyon and M. Wong. Sender ID: Authenticating E-Mail. RFC 4406, Interent Enginnering Task Force, 2006.

[24] Steve Martin, Anil Sewani, Blaine Nelson, Karl Chen, and Anthony D. Joseph. Analyzing behaviorial features for email classification. In *Conference on Email and Anti-Spam*, 2005.

[25] Patrick McDaniel, Shubho Sen, Oliver Spatscheck, Jacobus Van der Merwe, Bill Aiello, and Charles Kalmanek. Enterprise security: A community of interest based approach. In *Proceedings of Network and Distributed Systems Security*, 2006.

[26] Carman Neustaedter, AJ Bernheim Brush, Marc A Smith, and Danyel Fisher. The social network and relationship finder: Social sorting for email triage. In *Conference on Email and Anti-Spam*, 2005.

[27] M.E.J Newman, Stephanie Forrest, and Justin Balthrop. Email networks and the spread of computer viruses. In *Conference on Email and Anti-Spam*, 2005.

[28] M. Perone. An overview of spam blocking techniques. Technical report, Barracuda Networks, 2004.

[29] Mehran Sahami, Susan Dumais, David Heckerman, and Eric Horvitz. A bayesian approach to filtering junk E-mail. In *Learning for Text Categorization: Papers from the 1998 Workshop*, Madison, Wisconsin, 1998.

[30] P.M. Small, P.C. Hopewell, S.P. Singh, J. Parsonnet, D.C. Ruston, G.F. Schecter, C.L. Daley, and G.K. Schoolnik. The epidemiology of tuberculosis in San Francisco. A population-based study using conventional and molecular methods. *New England Journal of Medicine*, 330(9):1703–1709, June 1994.

[31] SpamAssassin. `http://spamassassin.apache.org/`, 2007.

[32] SpamBully. `http://www.spambully.com/`, 2007.

[33] Spamhaus. `http://www.spamhaus.org/sbl/`, 2007.

[34] Salvatore J. Stolfo, Shlomo Hershkop, Ke Wang, Olivier Nimeskern, and Chia-Wei Hu. A behavior-based approach to securing email systems. 2003.

[35] Joshua R. Tyler, Dennis M. Wilkinson, and Bernardo A. Huberman. Email as spectroscopy: automated discovery of community structure within organizations. pages 81–96, 2003.

[36] M. Wong and W. Schlitt. Semder Policy Framework (SPF) for Authorizing Use of Domains in E-Mail, Version 1. RFC 4408, Interent Enginnering Task Force, 2006.