Analyzing Group Communication for Preventing Data Leakage via Email

Polina Zilberman[+], Asaf Shabtai[*], Lior Rokach[*]
Deutsche Telekom Laboratories at Ben-Gurion University of the Negev
[+]Computer Science Department at Ben-Gurion of the Negev
[*]Information Systems Engineering Department at Ben-Gurion University of the Negev

Abstract

Modern business activities rely on extensive email exchange. Email "wrong recipients" mistakes have become widespread, and the severe damage caused by such mistakes constitutes a disturbing problem both for organizations and for individuals. Various solutions attempt to analyze email exchange for preventing emails to be sent to wrong recipients. However there is still no satisfying solution: many email addressing mistakes are not detected and in many cases correct recipients are wrongly marked as potential addressing mistake.

In this paper we present a new approach for preventing emails "slip-ups" in organizations. The approach is based on analysis of emails exchange among members of the organization and identification of groups of members that exchange emails with common topics. Each member's topics are then used during the enforcement phase for detecting potential leakage. When a new email is composed and about to be sent, each email recipient is analyzed. A recipient is approved if the email's content belongs to at least one of the topics common to the sender and the recipient.

We evaluated the new approach by comparing its detection performance to a baseline approach using the Enron Email dataset. Our evaluation results suggests that group communication analysis improves the performance of a baseline email classifier, which classifies a new email based only on emails exchanged in the past between the sender of the email and each of the recipients.

1 Introduction

Most of the people that intensely use email communication can confirm that at least once they have sent an email to the wrong person. Such a mistake can be very damaging. Inappropriate jokes may be sent to a supervisor, financial reports may be sent to a competitor or a broker, love letter to the wrong man or woman. An example of such incident was published onportfolio.com site on February 5th 2008[1]: "One of Eli Lilly & Co.'s sub-contracted lawyers at Philadelphia based Pepper Hamilton had mistakenly emailed confidential Eli Lilly's discussions to *Times* reporter Alex Berenson (instead of Bradford Berenson, her co-counsel), costing Eli Lilly nearly $1 billion."

When attempting to prevent such mistakes, the challenge is to correctly analyze emails sent between individuals. Various solutions to this problem are continuously emerging, however there is still no satisfying one.

In this paper we define an email leakage as an email that intentionally or unintentionally reaches a recipient it should not reach, i.e., a recipient that the email's sender has added intentionally or unintentionally. We present a new approach to be applied in organizations for preventing email leakage. According to the proposed approach we analyze the emails communicated between all members of the organization, extract the topics discussed in the organization via email exchange, and derive groups of members that share the same topic. Consequently, each member may belong to several topic groups, and a topic group may contain members that have never communicated before. When a new email is composed each recipient is classified as a potentially leak recipient or a legal one. The classification is based not only on the emails exchanged between the sender and the recipient, but also based on the topic groups which they belong to. It should be noted that partners of an organization are also considered as members of the organization for analysis purposes. We compare our method to the baseline classification approach that classifies an email considering only the emails exchanged between the sender and the recipients of the email, termed link-based email leakage detection.

Enron email dataset [1] is used for evaluating our approach. All emails are analyzed and split to training

[1]http://www.portfolio.com/news-markets/top-5/2008/02/05/Eli-Lilly-E-Mail-to-New-York-Times/

and testing sets. The older emails are used for the training and the generation of the classifiers, and the newer emails – for testing. Different types of simulated recipients are planted into the test emails. For each recipient, original and simulated, the classifiers determine whether the recipient is legal (i.e., original), or a potential leak (i.e., simulated).

Preliminary evaluation results show that the new approach outperformed the baseline approach in end cases where there exists no previous connection between the two. The results indicate that in order to gain the advantages of both approaches and eliminate their weak points, the approaches should be cascaded.

The rest of the paper is organized as follows. In section 2 we review state-of-the-art solutions. Section 3 presents the problem statement and overview of the proposed solution. In section 4 we describe in details the baseline and the proposed classification models. In section 5 we explain the performed evaluation. Section 6 concludes the paper.

2 Related works

We reviewed techniques for detecting recipients that were added by mistake or on purpose to an email they should not receive. We present summaries of the relevant studies. Our review includes advanced commercial products and the latest academic research results.

Commercial products, such as Symantec [2], Websense [3], McAfee [4], RSA, and Vericept [5], aim to prevent sensitive data from leaking via electronic communication channels. Their solutions are embedded at the network level where an email is inspected and a policy can be applied. A policy may define groups of users that are allowed to be exposed to certain contents. Therefore, it can identify recipients that should not receive the email. The commercial products address mainly emails that are sent out of the organization.

Google provides to its users an application the aims to prevent an email from reaching the "Wrong Bob". To the best of our knowledge Google's "Wrong Bob" is based on analyzing the groups of people a user usually exchange emails with, and alerting the user if an unexpected person has been added to the email. The disadvantage of Google's application is that it only works on group emails. "Got the Wrong Bob" won't know if a user has got the wrong Bob when he or she is sending an email to a single recipient [6]-[8].

Kalyan and Chandrasekaran [9] propose email pattern analysis to detect data leaks via email. The likelihood that an email has been sent by mistake is determined by analyzing attributes of emails previously exchanged between the sender and the recipients of the email. The attributes include time, attachment size, salutation and ending, existence of BCC recipients, etc. The proposed technique has been used on real-life emails, with detection accuracy close to 92%.

Carvalho and Cohen [10] predicted whether a sent email is a leak or not based on the textual content of the email, and how likely that the email recipient should receive a particular message. Messages sent to past recipients are modeled into <*message*,*recipient*> pairs, and a <*message*,*recipient*> pair is considered to be a potential leak if the message is sufficiently different from past messages sent to that recipient.

The solution of Carvalho and Cohen proposed two different techniques for leakage detection. The first technique relies strictly on the message's textual content. It measures the similarity between two vector-based representations of email messages. The first vector is a *TF-IDF* [11] representation of all previous messages from the user to the specific recipient (a different vector is created for each recipient). The second vector is a *TF-IDF* representation of the current message about to be sent. The distance between two vectors is measured using one of two suggested algorithms: *Cosine-Similarity* or *k-Nearest Neighbors* (*KNN*). If the computed similarity is smaller than a predefined threshold, then a warning message is issued to the user who is about to send the message. This comparison is done separately for each recipient of the message about to be sent.

The second technique is a classification-based method and has been implemented by using social network information (such as the number of received and sent messages, the number of times two recipients were addressed in the same message, etc.). The idea is to perform the leak prediction in two steps. In the first step, textual similarity scores are calculated using a cross-validation procedure in the training set. In the second step, network features are extracted and then a function is calculated, that combines these features with the textual scores.

In order to test their method, email leaks were simulated using the Enron email dataset [1]. The dataset was used to imitate realistic types of leaks, such as misspellings of email addresses, typos, similar first/last

names, etc. The method was able to detect email leaks in almost 82% of the test cases. The advantage of this approach is that it can be easily implemented for an email client and it does not use any information that is only available to the server.

In a later study by Carvalho et al. [12], the authors present a case study of the solution in [10] on the Mozilla Thunderbird. They also expanded the proposed solution not only to detect undesired recipients, but also to suggest recipients that the user has forgotten to address. The new method uses various machine learning and data mining techniques. These techniques study past email exchanges and suggest, according to the learned model, adding or removing a recipient. It has been proposed installing the solution as a plug-in to the Mozilla Thunderbird engine.

Participants in the study conducted by Carvalho et al. were required to write email using Thunderbird on a daily basis. The evaluation showed diverse results: more than 15% of the users reported that the email client prevented real cases of email leaks; more than 47% of the users accepted recommendations provided by the data mining techniques; and more than 80% of the users reported that they would permanently use this solution if a few improvements were added.

Stolfo et al. [13] demonstrate how the Email Mining Toolkit (EMT) [14] can detect the beginning of a viral propagation in emails without content-based or signature-based analysis. EMT is a data mining system that is applied online to email files gathered from email clients or server logs. It retrieves models of user email accounts and of groups of accounts, including the social cliques embedded in the user's email behavior patterns. EMT aggregates statistical information from groups of accounts and provides the means for detecting malicious users.

3 Problem statement and proposed solution

3.1 Problem statement

When classifying an email emanating from the computer of an individual, current academic solutions (that base on social interaction traffic analysis) focus on analyzing the emails sent and received by the individual. These solutions provide accurate analysis in most of the cases, however, there are cases in which the sole analysis of emails sent and/or received in the past by an individual is not enough for correctly classifying a new email the individual is about to send. For example, assume the members of a group *G* discuss topic *T*. Alice and Bob belong to *G*, but have never discussed topic *T* before (or even "worse", Alice and Bob have never communicated before). If Bob communicates content from topic *T* to Alice, current techniques may classify it by mistake as a potential leak (see Figure 1).

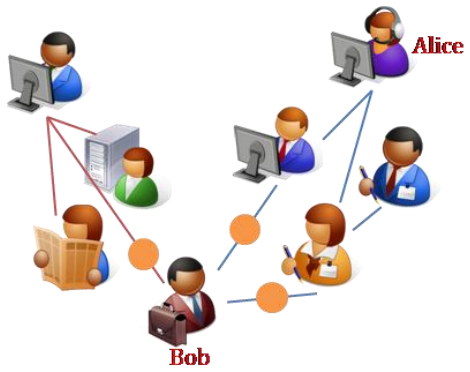

Figure 1. Orange circles represent the emails taken into account when classifying an email sent from Bob to Alice.

In commercial products it is possible to define groups of users that are allowed to receive emails with a specific content. These users are not necessarily communicating with each other. Thus, in cases as described above, sending an email from Bob to Alice won't be classified as a potential leak. However, identifying the groups in issue is a manual task, which begets a tremendous effort when considering a large organization with thousands of users.

3.2 Proposed solution

The proposed classification scheme is based on email exchange traffic among members of the organization (further on members of the organization will be referred to as *users*). The proposed scheme consists of two phases. In the first phase, groups of users that exchange emails with similar content, i.e., common topic, are identified. It is assumed that a user may belong to several groups working on distinct topics (typical for a manager). This phase will be further referred to as '*Group Communication Analysis*'.

In the second phase, each new email that is about to be sent is analyzed as follows: For each recipient of the email it is checked whether the recipient and the sender of the email belong to (at least one) common topic group. If such a group does not exist, it may be concluded that there is no common topic for the two users to discuss, and the aforementioned recipient is a wrong recipient. Otherwise, the content of the email is compared to the content of emails exchanged in the

group. If the similarity score is high enough, the recipient may be considered as a legal recipient. For example, assume Alice and Bob belong to the same group that communicates topic *T*, and Bob sends an email with content *T* to Alice. Alice won't be considered a wrong recipient, even if Alice and Bob have never exchanged communication with content *T* before (see Figure 2).

Group communication analysis provides additional information about potential connections between users who discuss similar topics, but do not necessarily communicate with each other. Thus, it better reflects the "real picture" of topics common to different users, than the sole analysis of the individual user communication with other users.

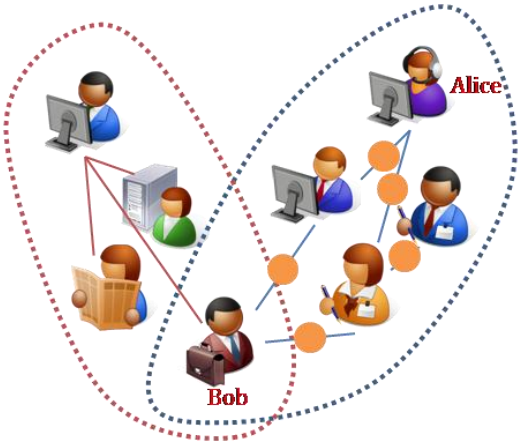

Figure 2: Orange circles represent the emails taken into account when classifying an email sent from Bob to Alice.

4 Classification model

A classification model consists of two phases; training phase (i.e., generating a model) and classification phase. The training is applied on a set of emails known to be "leak free" and the classification is applied on a newly composed emails represented as *queries*. Let a *query* refer to an email with content $\boldsymbol{c}$ that is about to be sent from a sender $\boldsymbol{s}$ to a recipient $\boldsymbol{r}$, and it is modeled as the triplet $\langle \boldsymbol{s}, \boldsymbol{r}, \boldsymbol{c} \rangle$ .Therefore, an email with $\boldsymbol{x}$ recipients defines $\boldsymbol{x}$ queries ($\langle \boldsymbol{s},\boldsymbol{r}_1,\boldsymbol{c} \rangle, \ldots, \langle \boldsymbol{s},\boldsymbol{r}_x, \boldsymbol{c} \rangle$).

The content of every email in the dataset is represented by a *TF-IDF* vector [11]. These vectors are computed as follows:

a. Extract terms from the emails in the training dataset after removing stop words and applying a stemming algorithm.
b. For every term, compute the Inverse Document Frequency (*IDF*) value: $\log \frac{|D|}{d}$, where $|D|$ is the number of emails in the training dataset and d is the number of emails in the training dataset containing the term.
c. For every email, compute the term frequency (*TF*) values for all terms extracted in step (a) where the *TF* refers to the term's frequency in the current email only.
d. For every email, compute its *TF-IDF* vector representation, where the vector components are the *TF*×*IDF* values of the terms extracted in step (a). This vector will be further denoted as *TF-IDF*(*c*), where *c* stands for the content of the email

Our goal is to show that group communication analysis improves the performance of the baseline email classifier. As a baseline email classifier we consider an email classifier that analyzes only the past emails exchange of the sender with the recipients.

4.1 Baseline classification model

Every two users that have exchanged emails in the past define a **link**, and all emails exchanged between these two users are associated with the link. During the training phase we compute:

(i) **centroid** of the link – the *TF-IDF* vector representation of the link, which will be further denoted as *TF-IDF*(*link*). The centroid represents a typical email associated with the link.

(ii) **threshold similarity score** of the link – determines how similar a new email should be in order to be associated with the link.
Setting the threshold score to such that considers all emails associated with the link as "normal", may result in high false-negative classifications (i.e., emails that do not belong to the link, falsely associated with the link since their similarity to *TF-IDF*(*link*) is high enough). Hence, we search for a compromised similarity threshold, which in turn may cause for false-positive classifications (i.e., emails that belong to the link, falsely not associated with it).

The training phase is performed as follows:
For every link we collect the *TF-IDF* vector representations of the emails associated with the link and perform the following steps:

1. Compute *TF-IDF*(*link*):
   a. Sum up *TF-IDF* vectors of the associated emails.

b. Divide the sum by the link's size, i.e., number of associated emails.

2. For every associated email with content *c*, compare *TF-IDF*(*c*) to *TF-IDF*(*link*) using Cosine similarity function, and sort the received similarity scores in ascending order. Store the scores in *Sorted Scores Array* (see an example in Figure 3).

0.1	0.2	0.25	0.4	0.4	0.5	0.7	0.76	0.8	0.9

Figure 3: Example of *Sorted scores array* indexed from 0 to 9

3. The system administrator sets a threshold parameter that controls the system sensitivity. The parameter's value is between 0 and 1. This value is translated into a threshold similarity score by each link. The link's threshold similarity score is the one whose index in the sorted scores array is such that:

   Index = |similarity scores| × (threshold parameter) - 1

   For example, assume the sorted similarity scores array given in Figure 3 and a threshold parameter that equals 0.3, then: score index = 10×0.3-1= 2. Hence, the threshold similarity score of the link is 0.25.

The threshold parameter represents the percentage of associated emails whose similarity scores are lower than the link's threshold score, i.e., the percentage of associated emails that are falsely excluded from the link.

Overview of the training phase flow is depicted in Figure 4.

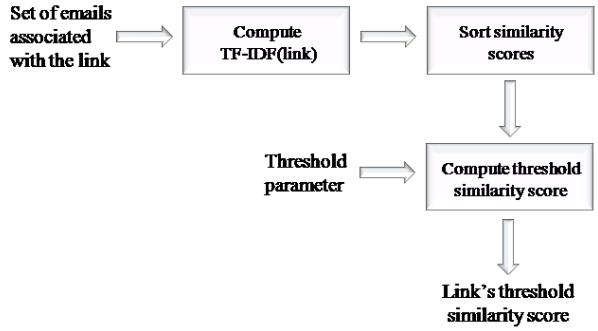

Figure 4: Link training phase

The classification phase of an email with content *c* sent from *s* to *r* represented by a query ⟨*s*, *r*, *c*⟩ is performed as follows:

1. Compute *TF-IDF*(*c*).
2. Retrieve the *TF-IDF*(*link*) of the link defined by the users *s* and *r*.
3. Compare, using Cosine similarity function, *TF-IDF*(*c*) and *TF-IDF*(*link*).
4. If the received similarity score is lower than the link's threshold similarity score, then sending *c* to recipient *r* is considered a potential leak.

If *s* and *r* exchanged no emails in the past, then no corresponding analyzed link exists, and sending any content *c* to recipient *r* is considered a potential leak.

Overview of the classification phase is depicted in Figure 5.

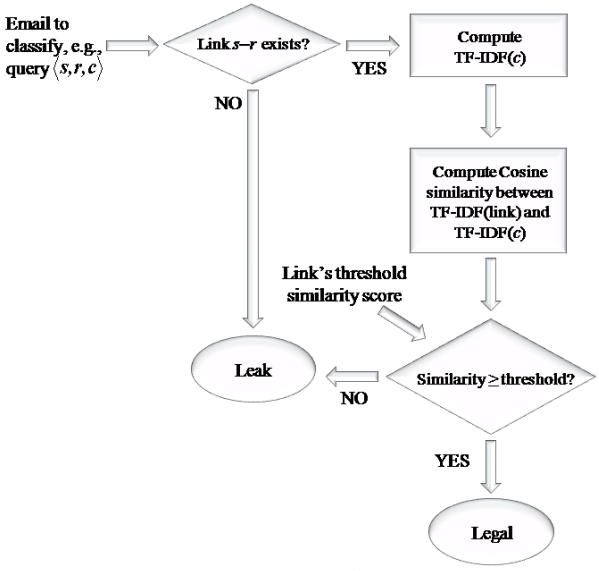

Figure 5: Link based classification phase

4.2 Proposed classification model

The training phase consists of grouping the users based on the emails they exchange. The members of each group exchange emails with similar content (i.e., topic). This phase is divided into two main sub-phases:

(i) **Identifying, via fuzzy clustering, the topics discussed in an organization**. For each identified topic compute:
   (a) cluster centroid – reflects the typical email associated with the topic; and,
   (b) threshold similarity score – determines how similar a new email should be in order to be associated with the topic.

(ii) **Projection of the different topics on the users**. This process includes counting for each user how many of his/her sent/received emails are associated with each one of the topics. Users that have emails associated with the same topic constitute a group of users with common topic. Note that a user may belong to several groups.

The training phase is performed as follows (see overview in Figure 6):

1. Compute the cluster centroids by applying *Fuzzy C-Mean Clustering Algorithm* [15] on the *TF-*

IDF(*c*) vectors in the training dataset. Each cluster centroid is denoted as *TF-IDF*(*cluster*).

2. The resulting *membership matrix* (see [15]) defines for each email in the training set the extent of relevance for each one of the derived clusters. Email *i* (represented by *TF-IDF*(*c*)) is associated with cluster *j* if:

   $\forall_{k \neq j}\, membership[i][j] \geq membership[i][k]$

   Thus, each email in the training set is associated with only one topic/cluster.

3. For every user *u* and cluster *j* compute the affinity vector, $affinity[u][j]$ which represents the number of emails that user *u* has sent or received and are associated with cluster *j*.

4. Users u_1 and u_2 belong to the same group g if:

   $\exists_i (affinity[u_1][g] \geq e\_t \wedge affinity[u_2][g] \geq e\_t)$,

   where *e_t* (emails threshold) is a threshold that defines the minimal number of emails associated with cluster g sent or received by a user.

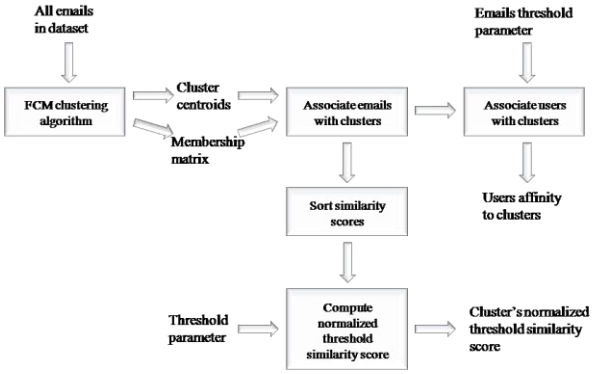

Figure 6: Group training phase

To compute the threshold similarity score of every cluster, derived during the Fuzzy C-Mean clustering algorithm, the following steps are performed (per cluster):

1. For every *cluster* associated email with content *c*, compare *TF-IDF*(*c*) to *TF-IDF*(*cluster*) using Cosine similarity function, and sort the received similarity scores in ascending order (store the scores in *Sorted_Score_Array*). The normalized similarity score is computed as follows:
   a. Compute the variance of the similarity scores.
   b. For every email's similarity score compute normalized similarity score:
      email's normalized score = (1-variance)× score
   c. Update the *Sorted_Score_Array* with the normalized scores.
2. Given a threshold parameter whose value is between 0 and 1, derive the cluster's normalized threshold similarity score. The cluster's normalized threshold similarity score is the one whose index in the *Sorted_Score_Array* is such that (analogous to link's threshold similarity score):
   Index = |similarity scores| × (threshold parameter) - 1
   Thus, normalized threshold similarity score = Sorted_Score_Array[Index].

The classification phase of an email with content *c* sent from *s* to *r* represented by a query $\langle s, r, c \rangle$ is performed as follows (see overview in Figure 7):

1. Retrieve all clusters for which holds: $affinity[s][j] > e\_t \wedge affinity[r][j] > e\_t$, where *j* is cluster index.
2. The normalized similarity score is computed as follows:
   a. For every cluster retrieved in the previous step, compare using Cosine similarity function *TF-IDF*(*cluster*) and *TF-IDF*(*c*). Normalize the similarity scores as described above.
   b. Find the cluster for which Cosine(*TF-IDF*(*cluster*), *TF-IDF*(*c*)) is maximal, denote this cluster as *maxCluster*.
   c. The normalized similarity score given to the query is:
      (1-variance)× Cosine(*TF-IDF*(*maxCluster*), *TF-IDF*(*c*))
3. Compute for *maxCluster* the normalized threshold similarity score.
4. If the received (normalized) similarity score is lower than the (normalized) threshold similarity score, then sending *c* to recipient *r* is considered a potential leak.

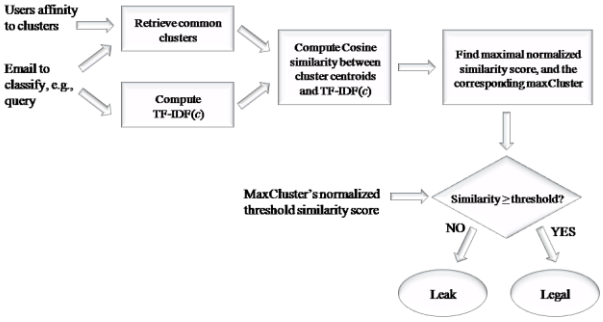

Figure 7: Group based classification phase

5 Evaluating the proposed model

5.1 Dataset

For implementation and evaluation purposes we have used the Enron email dataset [1]. It contains 517,430

emails organized into folders. The Enron email dataset has two known flaws that need to be addressed. The first is duplicate emails, which were removed. The second is Enron-users that use multiple email addresses. This has been partially addressed by mapping some of the Enron-users' email addresses into normalized email addresses [1]. In our implementation a User is defined by a distinct email address, which appears as a sender address or a recipient address in at least 20 emails.

For the purpose of preliminary evaluation 17 random Enron-user folders, containing 22,081 emails, were chosen as the dataset. As a first step we have loaded all emails and parsed their headers. Header fields which have been extracted are: 'From' (sender), 'To', 'Cc', 'Bcc' (recipients), and 'Date'. After removing duplicates, the emails have been sorted according to the chronological order of the sending date and time; from oldest email to the newest one. 90% (also a configurable parameter) of the emails (the oldest emails) were chosen as the training set, while the rest 10% of the emails (the newest emails) were used as the testing set.

The second step was to parse the email bodies of the training set. Terms were extracted from email subject and email body; stop words filtering and stemming was applied. *IDF* values were calculated for the extracted terms, and *TF-IDF*(*c*) was computed for every email. For each user we have computed its Address-Book, which includes the recipients in the emails sent by the user and the senders in the emails received by the user.

The third step was to parse the emails bodies of the testing set. Similarly to the parsing of training set emails, terms were extracted from email subject and email body, while filtering stop words and stemming the terms. *TF-IDF*(*c*) was computed for every email based on the *IDF* values derived from the training set.

5.2 Simulating wrong recipients

For each email in the testing set sent by user *u*, two randomly simulated recipients, which are not among the original recipients of the email, were "injected"; a recipient that appears in *u*'s Address-Book, and a recipient that does not appear in *u*'s Address-Book. If all the email addresses that appear in *u*'s Address-Book are among the original recipients of the test email, then only a simulated recipient of the second type is added. If the user *u* is not among the users derived from the emails in the training set, then the email is omitted from the testing set.

5.3 Parameters configurations

The experiment has been executed with different configurations of the parameters (described below). The parameters can be divided into two categories:

Parameters used in both classification models

1. Threshold similarity parameter – defines the threshold similarity score for a link or a cluster (as described in Section 4). The threshold parameter represents the percentage of emails, that constitute the link (or the cluster), who are excluded from the link (or the cluster). The parameter's values range from 0 to 1 with steps of 0.05.
2. Dataset splitting parameter – defines the percentage of emails that will be used as training set. The parameter's value is set to 0.9.

Group-based classification model parameters

Emails threshold parameter – defines the minimal amount of emails that a user should have sent or received and that are associated with a certain topic, so that the user will belong to the topic's group. The parameter's value is set to 1.

Fuzzy C-Mean Clustering algorithm parameters (for more details see [16]):

Parameter	Domain of values
Clusters #	10, 20
Fuzziness	2.0
Epsilon	0.005
Max. iterations #	100

Figure 8: Input params. for the FCM clustering algorithm

For every configuration of the parameters the experiment was executed 5 times, each time simulating new recipients to be "injected" into the test emails.

5.4 Execution

The link-based and group-based classifiers were trained and then tested. For every (non-omitted) test email, for each one of the recipients (original and simulated) we applied in turn the link-based and group-based classifiers. Every classifier performed the following:

1. Computed the similarity scores given to the email.
2. For every threshold similarity parameter, computed the threshold similarity scores.

Finally, for every classifier, we computed the accuracy results. For every similarity threshold parameter we computed the True-Positive Rate (TPR), i.e., how many leak (simulated) recipients were correctly classified as a leak, and the False-Positive Rate (FPR), i.e., how many legal (original) recipients were

falsely classified as a leak. The TPR and FPR were computed separately for "known" recipients – email addresses that appear in the Address-Book of the email's sender – and "unknown" recipients – email addresses that do not appear in the sender's Address-Book.

5.5 Results

The classifiers' accuracy results for every configuration of the parameters were averaged over 5 executions with different simulated recipients that were "injected" into the test emails.

There are two different configurations of the group-based classifier for every value of the threshold similarity parameter. For comparing the results of the link-based and group-based classifiers we first had to choose, for every value of the threshold similarity parameter, the best configuration of the group-based classifier.

Accuracy results of the link-based classifier are presented in Table 1. Table 2 presents the accuracy results of the group-based classifier for the following configuration: clusters# = 20, max. iterations# = 100, fuzziness = 2.0, and epsilon = 0.005.

Link-based classifier accuracy results

"Known" legal recipients were classified as leak (FP), because the "history" with the sender has been too short, e.g., not enough emails or too short emails. "Known" leak recipients were classified as legal (FN) because of the same reason – short history with the sender. "Unknown" legal recipients were classified as leak (FP), because they had no history at all with the sender.

Group-based approach VS. Link-based approach

"Known" and "Unknown" simulated recipients – the group-based approach has lower TPR. This is caused by the ability of group-based approach to identify potential topic-oriented connection between users that had no connection in the past, or had never directly discussed the specific topic before. However, since the recipient is a simulated recipient, it is considered to be a leak. Therefore, when a simulated recipient is classified as legal, it is considered to be a False-Negative FN, i.e., a leak (simulated) recipient that is falsely classified as legal.

"Known" and "Unknown" original recipients – the group-based approach has lower FPR. The reason for this is again the ability of group-based approach to identify potential topic-oriented connection between users.

Thresh. param.	TPR "known" recp.	TPR "unknown" recp.	FPR "known" recp.	FPR "unknown" recp.
0	0.11092	1	0	1
0.05	0.93429	1	0.32794	1
0.1	0.94014	1	0.34296	1
0.15	0.94595	1	0.39801	1
0.2	0.95266	1	0.42352	1
0.25	0.95633	1	0.48219	1
0.3	0.96065	1	0.49994	1
0.35	0.96152	1	0.57531	1
0.4	0.96476	1	0.59371	1
0.45	0.97856	1	0.61987	1
0.5	0.99502	1	0.65626	1
0.55	0.99611	1	0.66274	1
0.6	0.99632	1	0.67129	1
0.65	0.99739	1	0.68139	1
0.7	0.99804	1	0.69162	1
0.75	0.99848	1	0.7012	1
0.8	0.99913	1	0.71001	1
0.85	0.99935	1	0.71416	1
0.9	0.99935	1	0.71804	1
0.95	0.99956	1	0.72037	1
1	1	1	1	1

Table 1: Link-based classifier accuracy results

Apparently, cascading the group-based and link-based classifiers will take advantage of the "strong" points of both classifiers, and eliminate their "weak" points. A simple cascading is described by a flow chart in Figure 9. In this cascading, a known recipient is classified by the link-based classifier while an unknown recipient is classified by the group-based classifier.

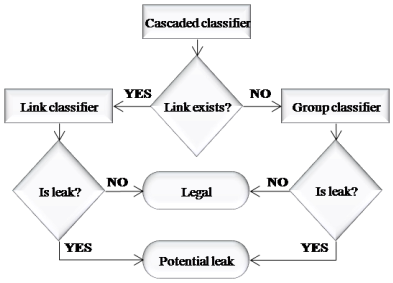

Figure 9: Cascaded classifier flow chart

To conclude, the cascading may generate a system with TPR=0.945945952 and FPR=0.39800543 (highlighted in Table 1) for "known" recipients, and TPR=0.506623386 and FPR=0.40047506 (highlighted in Table 2) for "unknown" recipients.

Thresh. param.	TPR "known" recp.	TPR "unknown" recp.	FPR "known" recp.	FPR "unknown" recp.
0	0	0.0001299	0	0
0.05	0.05505	0.0523377	0.04486	0.0157957
0.1	0.0769	0.0801299	0.06502	0.0331354
0.15	0.09943	0.1042857	0.08432	0.045962
0.2	0.12173	0.1301299	0.10403	0.0745843
0.25	0.15184	0.1605195	0.12392	0.0922803
0.3	0.17474	0.1853247	0.13752	0.1090261
0.35	0.20089	0.2103896	0.15216	0.1374109
0.4	0.23129	0.2393507	0.17029	0.1562945
0.45	0.25032	0.2619481	0.18656	0.172209
0.5	0.27214	0.2883117	0.20321	0.2028504
0.55	0.2947	0.3154545	0.22632	0.2356295
0.6	0.33118	0.3502597	0.24761	0.2522565
0.65	0.36324	0.3835065	0.26761	0.2720903
0.7	0.40108	0.4249351	0.29727	0.3119953
0.75	0.43932	0.4661039	0.32693	0.3469121
0.8	0.4743	0.5066234	0.36726	0.4004751
0.85	0.52614	0.5564935	0.40813	0.4429929
0.9	0.58632	0.618961	0.45971	0.5704276
0.95	0.65717	0.687013	0.53415	0.8668646
1	1	1	1	1

Table2: Group-based classifier accuracy results for clusters#=20, fuzziness=2.0, epsilon=0.005, and max. iterations#=100

6 Conclusions

Modern communication between individuals and organizations rely on extensive email exchange. Email "wrong recipients" mistakes have become a disturbing problem for all those who use email. Although various solutions exist, there is still no "silver-bullet" solution.

In this paper we present a new approach for preventing email recipients "slip-ups" in organizations. The approach is based on analysis of emails exchange among members of the organization and identification of groups of members that share common topics. When a new email is about to be sent, its content topic is verified against topics that both the sender and the recipient of the email are associated with.

We evaluated the new approach by comparing its detection performance to a baseline approach using the Enron Email dataset. Our evaluation results show that group communication analysis reduces the amount of original (legal) recipients that are wrongly marked as potential addressing mistake. These results indicate that in order to gain the advantages of the proposed approach and the baseline approach – the two should be cascaded. A preliminary cascading approach is described. Future research should focus on designing, implementing, and evaluating a more advanced cascading approach that will reduce the FP rate.

7 References

[1] W. W. Cohen, Enron Email Dataset Webpage, http://www.cs.cmu.edu/~enron/

[2] Symantec, http://www.symantec.com/business/products/family.jsp?familyid=data-loss-prevention

[3] Websense Data Security Solutions, White Paper.

[4] McAfee Host DLP, http://www.sans.org/reading_room/analysts_program/McAfee_Total_Protection_Jun09.pdf

[5] Vericept Protect, http://www.entrepreneur.com/tradejournals/article/173709078.html

[6]http://www.telegraph.co.uk/technology/google/6334469/Google-Mails-Got-The-Wrong-Bob-tool-aims-to-stop-email-errors.html

[7] http://www.paklinks.com/gs/rss-downloads/360830-gmails-got-the-wrong-bob-helps-avoid-misfired-emails.html

[8] http://gmailblog.blogspot.com/2009/10/new-in-labs-got-wrong-bob.html

[9] Kalyan C., Chandrasekaran K., "Information Leak Detection in Financial E-mails Using Mail Pattern Analysis under Partial Information", *Proc. of the 7th Conf.on 7th WSEAS International Conf.on Applied Informatics and Communications*, pp. 104-109, 2007.

[10] Carvalho V., Cohen W., "Preventing Information Leaks in E-mail", *SIAM International Conf.on Data Mining 2007*, 2007, http://www.cs.cmu.edu/wcohen/postscript/sdm-2007-leak.pdf).

[11] Salton G., BuckleyC., "Term-weighting approaches in automatic text retrieval", *Information Processing and Management*, Vol. 24, No. 5, pp. 513-523, 1988.

[12] Carvalho V.R., Balasubramanyan R., "Information Leaks and Suggestions: A Case Study using Mozilla Thunderbird", *Proc. of 6th conf. on email and anti-spam*, 2009.

[13] Stolfo S. J., Hershkop S. Hu C.W., Li W.J., Nimeskern O., Wang K., "Behavior-based modeling and its application to Email analysis", *ACM Trans. Internet Technol.*, Vol. 6, No. 2, pp. 187-221, 2006.

[14] Hershkop S., Wang K., Lee W., Nimeskern O., Creamer G., Rowe R., "Email Mining Toolkit Technical Manual", Department of Computer Science, Columbia University New York, NY, June 2006, (http://sneakers.cs.columbia.edu/ids/emt/software/EMT TechManualv 3 6 8.pdf), (http://sneakers.cs.columbia.edu/ids/emt/).
[15] Gath I., Geva A. B., "Unsupervised Optimal Fuzzy Clustering", *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol. 11, No. 7, pp. 773-780, 1989.