A Behavior-based Approach To Securing Email Systems

Salvatore J. Stolfo, Shlomo Hershkop, Ke Wang, Olivier Nimeskern, and Chia-Wei Hu

450 Computer Science Building
Fu Foundation School of Engineering & Applied Science
Computer Science Dept., Columbia University, USA
{sal, shlomo, kewang, on2005, charlie }@cs.columbia.eduz

Abstract. The Malicious Email Tracking (MET) system, reported in a prior publication, is a behavior-based security system for email services. The Email Mining Toolkit (EMT) presented in this paper is an offline email archive data mining analysis system that is designed to assist computing models of malicious email behavior for deployment in an online MET system. EMT includes a variety of behavior models for email attachments, user accounts and groups of accounts. Each model computed is used to detect anomalous and errant email behaviors. We report on the set of features implemented in the current version of EMT, and describe tests of the system and our plans for extensions to the set of models.

1. Introduction

The *Email Mining Toolkit* (EMT) is an offline data analysis system designed to assist a security analyst compute, visualize and test models of email behavior for use in a MET system [0]. In this paper, we present the features and architecture of the implemented and operational MET and EMT systems, and illustrate the types of discoveries possible over a set of email data gathered for study.

EMT computes information about email flows from and to email accounts, aggregate statistical information from groups of accounts, and analyzes content fields of emails without revealing those contents. Many previous approaches to "anomaly detection" have been proposed, including research systems that aim to detect masqueraders by modeling command line sequences and keystrokes [0,0].

MET is designed to protect user email accounts by modeling user email flows and behaviors to detect misuses that manifest as abnormal email behavior. These misuses can include malicious email attachments, viral propagations, SPAM email, and email security policy violations. Of special interest is the detection of polymorphic virii that are designed to avoid detection by signature-based methods, but which may likely be detected via their behavior.

The finance, and telecommunications industries have protected their customers from fraudulent misuse of their services (fraud detection for credit card accounts and telephone calls) by profiling the behavior of individual and aggregate groups of customer accounts and detecting deviations from these models. MET provides behavior-based protection to Internet user email accounts, detecting fraudulent misuse and policy violations of email accounts by, for example, malicious viruses.

A behavior-based security system such as MET can be architected to protect a client computer (by auditing email at the client), an enclave of hosts (such as a LAN with a few mail servers) and an enterprise system (such as a corporate network with many mail servers possibly of different types).

The principle behind MET's operation is to model baseline email flows to and from particular individual email accounts and sub-populations of email accounts (eg., departments within an enclave or corporate division) and to continuously monitor ongoing email behavior to determine whether that behavior conforms to the baseline. The statistics MET gathers to compute its baseline models of behavior includes groups of accounts that typically exchange emails (eg., "social cliques" within an organization), and the frequency of messages and the typical times and days those messages are exchanged. Statistical distributions are computed over periods of time, which serve as a training period for a behavior profile. These models are used to determine typical behaviors that may be used to detect abnormal deviations of interest, such as an unusual burst of email activity indicative of the propagation of an email virus within a population, or violations of email security policies, such as the outbound transmission of Word document attachments at unusual hours of the day.

EMT provides a set of models an analyst may use to understand and glean important information about individual emails, user account behaviors, and abnormal attachment behaviors for a wide range of analysis and detection tasks. The classifier and various profile models are trained by an analyst using EMT's convenient and easy to use GUI to manage the training and learning processes. There is an "alert" function in EMT which provides the means of specifying general conditions that are indicative of abnormal behavior to detect events that may require further inspection and analysis, including potential account misuses, self-propagating viral worms delivered in email attachments, likely inbound SPAM email, bulk outbound SPAM, and email accounts that are being used to launch SPAM.

EMT is also capable of identifying similar user accounts to detect groups of SPAM accounts that may be used by a "SPAMbot", or to identify the group of initial victims of a virus in a large enclave of many hundreds or thousands of users. For example, if a virus victim is discovered, the short term profile behavior of that victim can be used to cluster a set of email accounts that are most similar in their short term behavior, so that a security analyst can more effectively detect whether other victims exist, and to propagate this information via MET to limit the spread and damage caused by a new viral incident.

2. EMT Toolkit

MET, and its associated subsystem MEF (the Malicious Email Filter) was initially conceived and started as a project in the Columbia IDS Lab in 1999. The initial research focused on the means to statistically model the behavior of email attachments, and support the coordinated sharing of information among a wide area of email servers to identify malicious attachments. In order to properly share such information, each attachment must be uniquely identified, which is accomplished through the computation of an MD5 hash of the entire attachment. A new generation of polymorphic virii can easily thwart this strategy by morphing each instance of the attachment that is being propagated. Hence, no unique hash would exist to identify the originating virus and each of its variant progeny. (It is possible to identify the progenitor by analysis of entry points and attachment contents as described in the Malicious Email Filter paper [0].)

Furthermore, by analyzing only attachment flows, it is possible that benign attachments that share characteristics of self-propagating attachments will be incorrectly identified as malicious (e.g., a really good joke forwarded among many friends).

Although the core ideas of MET are valid, another layer of protection for malicious misuse of emails is warranted. This strategy involves the computation of behavior models of email accounts and groups of accounts, which then serve as a baseline to detect errant email uses, including virus propagations, SPAM mailings and email security policy violations. EMT is an offline system intended for use by security personnel to analyze email archives and generate a set of attachment models to detect self-propagating virii. User account models including frequency distributions over a variety of email recipients, and typical times of emails are sent and received. Aggregate populations of typical email groups and their communication behavior intended to detect violations of group behavior indicative of viral propagations, SPAM, and security policy violations.

Models that are computed by EMT offline serve as the means to identify anomalous, atypical email behavior at run time by way of the MET system. MET thus is extended to test not only attachment models, but user account models as well. It is interesting to note that the account profiles EMT computes are used in two ways. A long term profile serves as a baseline distribution that is compared to recent email behavior of a user account to determine likely abnormality. Furthermore, the account profiles may themselves be compared to determine subpopulations of accounts that behave similarly. Thus, once an account is determined to behave maliciously, similar behaving accounts may be inspected more carefully to determine whether they too are behaving maliciously.

The basic architecture of the EMT system is a graphical user interface (GUI) sitting as a front-end to an underlying database (eg., MySQL [0]) and a set of applications operating on that database. Each application either displays information to an EMT analyst, or computes a model specified for a particular set of emails or accounts using selectable parameter settings. Each is described below. By way of an example, Fig. 3 displays a collection of email records loaded into the database. This section allows an analyst to inspect each email message and mark or label individual messages with a class label for use in the supervised machine learning applications described in a later section. The results of any analyses update the database of email messages (and may generate alerts) that can be inspected in the messages tab.

2.1 Attachment Statistics and Alerts

EMT runs an analysis on each attachment in the database to calculate a number of metrics. These include, birth rate, lifespan, incident rate, prevalence, threat, spread, and death rate. They are explained fully in [0].

Rules specified by a security analyst using the alert logic section of MET are evaluated over the attachment metrics to issue alerts to the analyst. This analysis may be executed against archived email logs using EMT, or at runtime using MET. The initial version of MET provides the means of specifying thresholds in rule form as a collection of Boolean expressions applied to each of the calculated statistics. As an example, a basic rule might check for each attachment seen:

*If its **birth rate** is greater than specified threshold T AND **sent from** at least X number of users.*

2.2 Account Statistics and Alerts

This mechanism has been extended to provide alerts based upon deviation from other baseline user and group models. EMT computes and displays three tables of statistical information for any selected email account. The first is a set of stationary email account models, i.e. statistical data represented as a histogram of the average number of messages sent over all days of the week, divided into three periods: day, evening, and night. EMT also gathers information on the average size of messages for these time periods, and the average number of recipients and attachments for these periods. These statistics can generate alerts when values are above a set threshold as specified by the rule-based alert logic section.

We next describe the variety of models available in EMT that may be used to generate alerts of errant behavior.

2.3 Stationary User Profiles

Histograms are used to model the behavior of a user's email accounts. Histograms are compared to find similar behavior or abnormal behavior within the same account (between a long-term profile histogram, and a recent, short-term histogram), and between different accounts.

A histogram depicts the distribution of items in a given sample. EMT employs a histogram of 24 bins, for the 24 hours in a day. (Obviously, one may define a different set of stationary periods as the detect task may demand.) Email statistics are allocated to different bins according to their outbound time. The value of each bin can represent the daily average number of emails sent out in that hour, or daily average total size of attachments sent out in that hour, or other features defined over an of email account computed for some specified period of time.

Two histogram comparison functions are implemented in the current version of EMT, each providing a user selectable distance function as described below. The first comparison function is used to identify groups of email accounts that have similar usage behavior. The other function is used to compare behavior of an account's recent behavior to the long-term profile of that account.

2.3.1 Histogram Distance Functions

A *distance function* is used to measure histogram dissimilarity. For every pair of histograms, h_1, h_2, there is a corresponding distance $D(h_1, h_2)$, called the distance between h_1 and h_2. The distance function is non-negative, symmetric and 0 for identical histograms. Dissimilarity is proportional to distance. We adapted some of the more commonly known distance functions: simplified histogram intersection (L1-form), Euclidean distance (L2-form), quadratic distance [0] and histogram Mahalanobis distance [0]. These standard measures were modified to be more suitable for email usage behavior analysis. For concreteness,

$$\text{L1-form: } D_1(h_1,h_2) = \sum_{i=0}^{n-1} | h_1[i] - h_2[i] | \tag{1}$$

$$\text{L2-form: } D_2(h_1,h_2) = \sum_{i=0}^{n-1} (h_1[i] - h_2[i])^2 \tag{2}$$

$$\text{Quadratic: } D_3(h_1,h_2) = (h_1 - h_2)^T A (h_1 - h_2) \tag{3}$$

where n is the number of bins in the histogram. In the quadratic function, A is a matrix where a_{ij} denotes the similarity between bins i and j. In EMT we set $a_{ij} = | i - j | + 1$, which assumes that the behavior in neighboring hours is more similar. The Mahalanobis distance is a special case of the quadratic distance, where A is given by the inverse of the covariance matrix obtained from a set of training histograms. We will describe this in detail.

2.3.2 Abnormal User Account Behavior

The histogram distance functions are applied to one target email account. (See Fig. 4.) A long term profile period is first selected by an analyst as the "normal" behavior training period. The histogram computed for this period is then compared to another histogram computed for a more recent period of email behavior. If the histograms are very different (i.e., they have a high distance), an alert is generated indicating possible account misuse. We use the weighted Mahalanobis distance function for this detection task.

The long term profile period is used as the training set, for example, a single month. We assume the bins in the histogram are random variables that are statistically independent. Then we get the following formula:

$$D_4(h_1, h) = (h_1 - h)^T A(h_1 - h) \tag{4}$$

$$A = B^{-1},\ b_{ii} = Cov(h[i], h[i]) = Var(h[i]) = \sigma_i^2 \quad B = \begin{pmatrix} \sigma_0^2 & 0 & \cdots & 0 \\ 0 & \sigma_1^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \\ 0 & \cdots & 0 & \sigma_{n-1}^2 \end{pmatrix} \tag{5}$$

Then we get:

$$D_4(h_1, h) = \sum_{i=0}^{n-1} ((h_1[i] - h[i])^2 / \sigma_i^2) \tag{6}$$

Vector h represents the histogram of the (eg., one month) profile period, while h_1 represents the recent profile period (eg., one week). σ_i describes the dispersion of usage behavior around the arithmetic mean. We then modify the Mahalanobis distance function to the weighted version. First we reduce the distance function from the second degree function to the first degree function; then we assign a weight to each bin so that the larger bins will contribute more to the final distance computation:

$$D_4(h_1, h) = \sum_{i=0}^{n-1} w_i (h_1[i] - h[i]) / \sigma_i \tag{7}$$

$$\text{Weight } w_i = h_1[i] / \sum_{j=0}^{n-1} h_1[j] \tag{8}$$

When the distance between the histogram of the selected recent period and that of the longer term profile is larger than a threshold, an alert will be generated to warn the analyst that the behavior "might be abnormal" or is deemed "abnormal". The alert is also put into the alert log of EMT.

2.3.3 Similar Users

User accounts that may behave similarly may be identified by computing the pair-wise distances of their histograms (eg., a set of SPAM accounts may be inferred given a known or suspect SPAM account as a model). Intuitively, most users will have a pattern of use over time, which spamming accounts will likely not follow. (SPAMbots don't sleep or eat and hence may operate at times that are highly unusual.)

The histogram distance functions were modified for this detection task. First, we balance and weigh the information in the histogram representing hourly behavior with the information provided by the histogram representing behavior over different aggregate periods of a day. This is done since measures of hourly behavior may be too low a level of resolution to find proper groupings of similar accounts. For example, an account that sends most of its email between 9am and 10am should be considered similar to another that sends emails between 10am and 11am, but perhaps not to an account that emails at 5pm. Given two histograms representing a heavy 9am user, and another for a heavy 10am user, a straightforward application of any of the histogram distance functions will produce erroneous results.

Thus, we divide a day into four periods: morning (7am-1pm), afternoon (1pm-7pm), night (7pm-1am), and late night (1am-7am). The final distance computed is the average of the distance of the 24-hour histogram and that of the 4-bin histogram, which is obtained by regrouping the bins in the 24-hour histogram.

Second, because some of the distance functions require normalizing the histograms before computing the distance function, we also take into account the volume of emails. Even with the exact distribution after normalization, a bin representing 20 emails per day should be considered quite different from an account exhibiting the emission of 200 emails per day.

In addition to find similar users to one specific user, EMT computes distances pair-wise over all user account profiles, and clusters sets of accounts according to the similarity of their behavior profile. To reduce the complexity of this analysis, we use an approximation by randomly choosing some user account profile as a “centroid” base model, and then compare all others to this account. Those account profiles that are deemed within a small neighborhood from each other (using their distance to the centroid as the metric) are treated as one clustered group. The cluster so produced and its centroid are then stored and removed, and the process is repeated until all profiles have been assigned to a particular cluster.

The histograms described here are stationary models; they represent statistics time frames. Other non-stationary account profiles are provided by EMT, where behavior is modeled over sequences of emails irrespective of time. These models are described next.

2.4 Non-Stationary User Profiles

Another type of modeling considers the changing conditions of an email account over sequences of email transmissions. Most email accounts follow certain trends, which can be modeled by some underlying distribution. As an example of what this means, many people will typically email a few addresses very frequently, while emailing many others infrequently. Day to day interaction with a limited number of peers usually results in some predefined groups of emails being sent. Other contacts communicated to on less than a daily basis have a more infrequent email exchange behavior. These patterns can be learnt through the analysis of a user’s email archive over a bulk set of sequential emails. For some users, 500 emails may occur over months, for others over days.

Every user of an email system develops a unique pattern of email emission to a specific list of recipients, each having their own frequency. Modeling every user's idiosyncrasies enables the EMT system to detect malicious or anomalous activity in the account. This is similar to what happens in credit card fraud detection, where current behavior violates some past behavior patterns.

Figure 5 and 6 are screenshots of the non-stationary model features in EMT. We will illustrate the ideas of this model referencing specific details in the screenshots.

2.4.1. Profile of a user

The Profile tab in Figure 11 provides a snapshot of the account's activity in term of recipient frequency. It contains three charts and one table.

The "Recipient Frequency Histogram" chart is the bar chart in the upper left corner. It displays the frequency at which the user sends emails to all the recipients communicated to in the past. Each point on the x-axis represents one recipient, the corresponding height of the bar located at any point on the x-axis measures the frequency of emails sent to this recipient, as a percentage.

This bar chart is sorted in decreasing order, and usually appears as a nice convex curve with a strong skewedness; a long low tail on the right side, and a very thin spike at the start on the left side. This frequency bar chart can be modeled with either a Zipf function, or a DGX function (Discrete Gaussian Exponential function), which is a generalized version of the Zipf distribution. This distribution characterizes some specific human behavioral patterns, such as word frequencies in written texts, or URL frequencies in Internet browsing [2]. In brief, its main trait is that few objects receive a large part of the flow, while many objects receive a very small part of the flow.

The rank-frequency version of Zipf's law states that $f(r) \propto 1/r$, where $f(r)$ is the occurrence frequency versus the rank r, in logarithmic-logarithmic scales. The generalized Zipf distribution is defined as $f(r) \propto (1/r)^{\theta}$, where the log-log plot can be linear with any slope. Our tests indicate that the log-log plots are concave, and thus require the usage of the DGX distribution for a better fit [2].

The "Recipient List" is the table in the upper right corner in Fig. 5. This table is directly related to the previous histogram. It lists in decreasing order every recipient's email address and their frequency of receiving messages. It is used as a reference to the Recipient Frequency Histogram, as the email address corresponding to each bar of the histogram can be found in the table in the same order.

"Total Recipient Address List size over time" is the chart in the lower left-hand side in Fig. 5. The address list is the list of all recipients who received an email from the selected user between two dates. This list is a cumulative distribution that grows over an increasing number of emails analyzed, as new distinct recipients are added to the list. It starts empty, then each time the selected user sends an email to a new distinct email address not yet in the list, that address is added to the list. The list size is the number of elements in the list at a given point in the set of bulk emails analyzed. (Note, the address book is the list of email addresses a user maintains in their mail client program for easy reference, unlike the address list, which is the list of all recipients a user may send to eg., in reply to messages from accounts not recorded in the user's address book).

The plots labeled "# distinct rcpts & # attach per email blocks" appear in the chart in the lower right-hand side of Figure 11. It contains five plots that visualize the variability of the user's emission of emails. The green plot is the number of distinct recipients per block of 50 emails sent. The program uses a rolling window of 50 emails to calculate this metric. What it means is, the higher its value (and thus the closer to 50), the wider the range of recipients the selected user sends emails to, over time. On the other hand, if the metric is low, it means that the user predominantly sends messages to a small group of people. The moving average of this metric (using 100 records) is plotted as a blue overlapping curve, indicating the trend.

The plot in yellow is based on the same criterion, but with a window of 20 emails instead of 50. This metric is chosen, so that it will have a faster reaction to anomalous behavior, while the previous one using blocks of 50 shows the longer-term behavior. The short-term profile can be used as the first level of alert, the longer-term one acting to confirm it. The 100 record average of this metric is displayed in blue as well.

Finally the plot in red is the number of messages with attachment(s), per block of 50 emails. It shows the average ratio of emails with attachment(s) versus emails without attachments, and any sudden spike of emails sent with attachment(s) will be detected on the plot.

The profile window displays a fingerprint of the selected user's email frequency behavior. The most common malicious intrusion can be detected very fast by the metrics. For instance, a Melissa type virus would be detected since the five plots in the chart will jump up to 50, 20 and 50 respectively (if the virus sends an attachment of itself, polymorphic or not, the red plot will jump up). See section 3.4.4 for more details on virus detection simulations.

Spammers too have a typical profile, easily distinguishable from a normal user's profile. The metrics in Figure 11 will be very high most of the time, as a SPAMbot generates emails to a very wide range of recipients. The address list size will likely grow much faster then a typical user. If a spammer sends many emails with attachments, the red metric will be close to 50 most of the time, unlike the general intuitive case of normal user behavior.

2.4.2 Chi Square Test of User Histograms

The purpose of the Chi Square window shown in Fig. 6 is to test the hypothesis that the recipient frequencies are identical (for a given user) over two different time frames. Obviously, recipient frequencies are not constant over a long time horizon, as users will add new recipients and drop old ones. It can be informative for behavioral modeling though, to analyze the variability of frequencies over two near time frames.

The window is composed of two histograms and two tables. They are constructed in the same manner as what is done in the Profile window, but here two time periods of activity for the same user are compared. The idea is to treat the first period, the "Training range" at the top, as the true distribution corresponding to the user under normal behavior, while the second time period, the "Testing range" at the bottom, is used to evaluate if frequencies have changed, and if any malicious activity is taking place.

By default, the 200 past emails are selected as the Testing range, while the previous 800 are the Training range, thus operating under the usual 1/5 - 4/5 ratio between testing and training sets, using the past 1000 messages as total set. These ranges are modifiable; a live version would use a much shorter testing period in order to provide fast alerts.

The scales on the x-axis for both Recipient Frequency histograms are the same, and are based on the sorted frequencies from the Training range. It implies that the addresses appearing only in the testing range (but not in the training range) are nevertheless present in both histograms, but with frequency zero in the training range histogram. Conversely, they have a non-zero frequency in the lower histogram and are located on the extreme right side. (One can see a jump in frequency on this side, as the sorting is based on the top histogram, where they had zero frequency.) As each recipient address is at the same location on the x axis on both histograms, the lower one does not appear to be sorted, as the order has changed between training and testing ranges. This shows how some frequencies drop while others spike between the two periods.

Finally, at the bottom of the window, a Chi Square statistic, with its p-value and degrees of freedom are displayed in blue. The Chi Square is a statistic that can be used, among other things, to compare two frequency tables. Assuming that the observed frequencies corresponding to the first, longer time frame window are the true underlying frequencies, the Chi Square statistic enables us to evaluate how likely the observed frequencies from the second time frame are to be coming from that same distribution [18]. The Chi Square formula is:

$$Q = \sum_{i=1}^{k} (X(i) - np(i)) / np(i) \tag{9}$$

Where $X(i)$ is the number of observations for recipient (i) in the testing range, $p(i)$ is the true frequency calculated from the training range, n is the number of observations in the testing range, and k is the number of recipients. There are (k-1) degrees of freedom.

The p-value represents the probability that the frequencies in both time frames come from the same multinomial distribution. In order to get an idea of the variability of the frequencies under real conditions, we used a sample of 37,556 emails from 8 users. We run two batches of calculations. First, we used a training period size of 400 emails and a testing period size of 100 emails; for each user, we started at the first record, calculated the p-value, then translated the two windows by steps of 10 records until the end of the log was reached, each time calculating the p-value. Secondly, we reproduced the same experiment, but with a training period size of 800 emails, and a testing period size of 200 emails. We thus collected a total of 7,947 p-values, and their histogram is shown in Fig. 1.

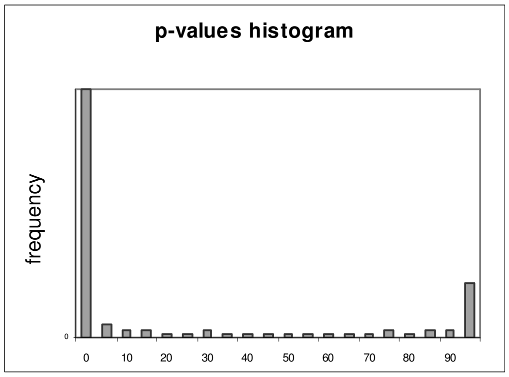


Fig. 1. P value plot

Under the hypothesis that the frequencies are constant, the histogram is expected to be a flat line. On the contrary, this histogram is characterized by a very large concentration of p-values between 0 and 5%, and a large (but less large) concentration between 95 and 100%, while p-values in the range of 5 to 95% are under-represented. Our intuitive explanation of this histogram (also based on our domain knowledge) is the following:

Most of the time, frequencies change significantly (in a statistical sense) between two consecutive time frames; this is why 60% of the p-values are below 5% (as a low p-value indicates a very high chance that the frequencies have changed between two time frames). Emails users tend to modify their recipient frequencies quite often. On the other side, there are non-negligible times when those frequencies stay very stable (as 13% of the p-values are above 95%, indicating strong stability). As the frequencies have been found to be so variable under normal circumstances, the Chi Square itself could not be used to detect an intrusion. Instead we explore a related metric, which will be more useful for that purpose.

2.4.3 Hellinger Distance

Our first tests using the Chi-square statistic revealed that the frequencies cannot be assumed constant between two consecutive time frames for a given user. What is specific to every user though is, how variable frequencies are over time. We try to assess this by calculating a measure between the two frequency tables.

We are using the Hellinger distance for this purpose. Its formula is:

$$HD(f_1[], f_2[]) = \sum_{i=0}^{n-1} (\sqrt{f_1[i]} - \sqrt{f_2[i]})^2 \tag{10}$$

Where $f_1[]$ is the array of frequencies for the training set, $f_2[]$ for the testing set, ***n*** the total number of distinct recipients during both periods. “Hellinger distance size” is a text field set at 100 by default. It represents the length of the testing window and can be changed to a different value, while the length of the training window equals 4 times the length of the testing window. Fig. 2 is shown an example for a normal user.

The Hellinger distance plot shows the distance between training and testing sets plotted over the entire email history of the user. For example, if a user has 2500 outbound emails, the plots starts at the 500th record, and measures the distance between the frequencies corresponding to the first 400 records, versus the emails corresponding to the next 100 records; these two windows, of 400 and 100 records, respectively, are then rolled forward over the entire email history of the user, by steps of one record. At each step, a Hellinger distance is calculated between the given training window of 400 records, and the corresponding testing window of 100 records.

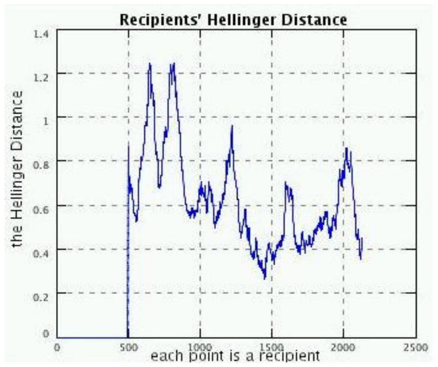


Fig. 2. Normal User

What this plot tells us is that when a burst occurs, the recipient frequencies have been changing significantly. This can be either a normal event, as we know from the previous section, or an intrusion. Thus the plot can be included in our list of detection tools. Formal evaluation of its efficiency, as well as of some of the metrics presented along section 3.4, is described in the next section.

2.4.4 Tests of simulated virii

As data with real intrusions are very difficult to obtain [19], EMT's menu includes the creation of simulated email records. Using a database of archived emails, EMT can generate the arrival of a "dummy" virus, and insert corrupted records into a real email log file. A set of parameters introduces randomness in the process, in order to mimic real conditions: the time at which the virus starts, the number of corrupted emails sent by the virus and its propagation rate. Each recipient of such a "dummy" corrupted email is picked randomly from the address list of the selected user. These recipients can be set to be all distinct, as most virii, but not all, would only send an email once to each target recipient account.

By design, each email generated by a simulated virus contains one attachment, but no information about the attachment is provided or used, other than the fact that there is one attachment. Our assumption is that virii propagate themselves by emitting emails with one attachment, which may or may not be a copy of themselves, so that the tests encompass polymorphic virii as well. Virii can also propagate through HTML content, and testing such scenario would follow the same logic as this test. The results are expected to be very similar, as most calculations would be identical. This new test would for the most part consists in just replacing one boolean random variable ("attachment") with another ("html").

Our experiment used a combination of three plots, the "Hellinger distance" plot, the "# distinct recipients" plot, and the "# attachment" plot, as detailed in the above sections. Our intuition is that when a virus infiltrates itself, it causes each plot to burst. We have tested three types of thresholds used to determine when a burst occurs:

a simple moving average (MA), a threshold proportional to the standard deviation of the plots (TH), and a heuristic formula evaluating when a change of trend occurs (HE).

The dataset for the test was an archive of 16 users, totaling 20,301 emails. The parameters that were randomly generated at each simulation were the time of the intrusion and the list of receiving recipients (taken from the address list of each selected user). The parameters that were controlled were the propagation rate (ranging from 0,5 message per day to 24) and the number of corrupted emails sent (ranging from 20 corrupted emails sent to 100). In total, about 500,000 simulations were made, and for each type, we determined the false alarm rate and the missing alarm rate. The results are summarized in the table 1.

No optimization was attempted, and we ran the experiment only once, as we did not want to appear to use the best out of several runs. In summary, we achieved very reasonable results, given the lack of optimization, and the simplicity of the threshold formulae. Thus, this method can be expected to be quite efficient once optimized and used in combination with other tools. As expected, a slower propagation rate makes detection harder, as in such a case, each corrupted email becomes less "noticeable" among the entire email flow (as can be seen in the table, each method gets worse results as the propagation rate decreases). A smaller number of emails sent by the virus would also make it harder to be detected (this can be seen by comparing the results between 20 and 50 corrupted emails; in the case of 100 emails sent, the results get lower for heuristic reasons only).

Propagation Rate:		24			2			1			0.5		
#corrupted email		20	50	100	20	50	100	20	50	100	20	50	100
Method	MA	41/9	27/9	39/9	56/10	51/11	59/12	60/10	54/12	61/12	63/11	53/12	55/12
	TH	49/4	41/4	60/4	69/4	67/5	73/6	73/5	69/6	73/6	76/5	65/5	67/6
	HE	25/8	21/8	48/8	48/9	54/11	63/12	59/10	63/13	67/13	69/11	73/12	67/13

Table 1. Each cell contains the missing alarm rate and the false alarm rate (for example 27/9 means that the missing alarm rate is 27% and the false alarm rate is 9%). The results for the three methods (MA, TH, HE) are shown for various propagation rates and number of emails emitted by the virus.

2.5 Group Communication Models: Cliques

In order to study the email flows between groups of users, EMT provides a feature that computes the set of *cliques* in an email archive.

We seek to identify clusters or groups of related email accounts that frequently communicate with each other, and then use this information to identify unusual email behavior that violates typical group behavior. For example, intuitively it is doubtful that a user will send the same email message to his spouse, his boss, his "drinking buddies" and his church elders all appearing together as recipients of the same message. A virus attacking his address book would surely not know the social relationships and the typical communication pattern of the victim, and hence would violate the user's *group behavior profile* if it propagated itself in *violation* of the user's "social cliques".

Clique violations may also indicate internal email security policy violations. For example, members of the legal department of a company might be expected to exchange many Word attachments containing patent applications. It would be highly unusual if members of the marketing department, and HR services would likewise receive these attachments. EMT can infer the composition of related groups by analyzing normal email flows and computing cliques (see Figure 5), and use the learned cliques to alert when emails violate clique behavior (see Figure 7).

EMT provides the clique finding algorithm using the branch and bound algorithm described in [0]. We treat an email account as a node, and establish an edge between two nodes if the number of emails exchanged between them is greater than a user defined threshold, which is taken as a parameter (Figure 7 is displayed with a setting of 100). The cliques found are the fully connected sub-graphs. For every clique, EMT computes the most frequently occurring words appearing in the subject of the emails in question which often reveals the clique's typical subject matter under discussion.

(The reader is cautioned not to confuse the computation of cliques, with the maximal Clique finding problem, that is NP-complete. Here we are computing the set of all cliques in an email archive which has near linear time complexity.)

2.5.1 Chi Square + cliques

The Chi Square + cliques (CS + cliques) feature in EMT is the same as the Chi Square window described in section 3.4.2, with the addition of the calculation of clique frequencies.

In summary, the clique algorithm is based on graph theory. It finds the largest cliques (group of users), which are fully connected with a minimum number of emails per connection at least equal to the threshold (set at 50 by default). For example if clique_1 is a clique of three users A, B and C, meaning that A and B have exchanged at least 50 emails; similarly B and C, and A and C, have exchanged at least 50 emails. The Clique Threshold field can be changed from this window, which will recalculate the list of all cliques for the entire database, and concurrently the metrics in the window are automatically readjusted accordingly.

In this window, each clique is treated as if it were a single recipient, so that each clique has a frequency associated with it. Only the cliques to which the selected user belongs will be displayed. Some users don't belong to any clique, and for those, this window is identical to the normal Chi Square window.

If the selected user belongs to one or more cliques, each clique appears under the name clique_i, i:=1,2,..., and is displayed in a cell with a green color in order to be distinguishable from individual email account recipients. (One can double click on each clique's green cell, and a window pops-up with the list of the members of the clique.)

For any connection between the selected user and a given recipient belonging to a clique, the algorithm implemented in EMT allocates 60% of the email flow from user to recipient to the given cliques, and the rest to the recipient. This number was chosen to reflect the fact that if a user and a recipient belong to the same clique, most of the email flow between the two is assumed to belong to the clique.

In some cases, two or more cliques may share the same connection between a user and a recipient. For example if A, B and C belong to clique$_1$, and A, B and D belong to clique$_2$, the connection between A and B is shared among the two cliques. In that case, half of the 60% allocated to cliques will be split between clique$_1$ and clique$_2$ in order to calculate the frequency, from A to B, say. If 100 messages were sent from A to B, 40 are assigned to B, 30 to clique$_1$ and 30 to clique$_2$.

Cliques tend to have high ranks in the frequency table, as the number of emails corresponding to cliques is the aggregate total for a few recipients. Let's for example assume that clique$_1$ = {A, B, C, D}, and that clique$_1$ shares no connection with other cliques. If A sent 200 messages to B, 100 to C and 100 to D, the number of messages allocated, respectively, to B is 80, to C is 40, to D is 40, and to is 240. Thus, the clique will get a large share of the flow, and this is expected, as they model small groups of tightly connected users with heavy email traffic.

2.5.2 Enclave cliques vs. User cliques

Conceptually, two types of cliques can be formulated. The one described in the previous section can be called *enclave cliques* because these cliques are inferred by looking at email exchange patterns of an enclave of accounts. In this regard, no account is treated special and we are interested in email flow pattern on the enclave-level. Any flow violation or a new flow pattern pertains to the entire enclave. On the other hand, it is possible to look at email traffic patterns from a different viewpoint altogether. Consider we are focusing on a specific account and we have access to its outbound traffic log. As an email can have multiple recipients, these recipients can be viewed as a clique associated with this account. Since another clique could subsume a clique, we defined a user clique as one that is not a subset of any other cliques. In other words, *user cliques* of an account are its recipient lists that are not subsets of other recipient lists.

To illustrate the idea of both types of cliques and show how they might be used in a spam detection task, two simulations are run. In both cases, various attack strategies are simulated. Detection is attempted based on examining a single attack email. Final results are based on how well such detection performs statistically.

In the case of enclave cliques, the following simulation is performed. An enclave of 10 accounts is created, with each account sending 500 emails. Each email has a recipient list that is no larger than 5 and whose actual size follows Zipf distribution, where the rank of the size of recipient lists is in decreasing order of the size; i.e. single-recipient emails have a rank of 1 and 5-recipient emails have a rank of 5. Furthermore, for each account, a random rank is assigned to its potential recipients and this rank is constant across all emails sent. Once the recipient list size an email is determined, the actual recipients of that email is generated based on generalized Zipf distribution, with theta = 2. Finally, a threshold of 50 is used to qualify any pair of accounts to be in a same clique.

In terms of attack strategies used, 5 different ones are tested. The first is to send to all potential recipient addresses, one at a time. The second, third and fourth attack strategies are to send 20 attack emails, each with 2, 3 and 5 randomly chosen addresses. The value, 20, is of no concern here, since detection is based solely on each email. The last strategy is to send a single email to all potential email addresses. During the detection phase, each attack email is examined and compared with previously classified enclave cliques. An alarm is triggered if the recipient list of the email is not a subset of any of the known enclave cliques. Finally, 10 replications of simulation are run and the resulting statistics are listed below. As expected from intuition, it is easier to detect a potential anomaly if the size of the recipient list of the attack email is large.

In the case of user cliques, the following simulation is performed. 200 emails are sent from an account to 10 potential recipients according to some rules. Each email has a recipient list size that is no larger than 5 and whose actual size is determined based on Zipf distribution, where the rank of the size of recipient lists is in decreasing order of the size; i.e. single-recipient emails have a rank of 1 and 5-recipient emails have a rank of 5. Furthermore, a random rank is assigned to its potential recipients and this rank is constant across all emails sent. Once the recipient list size of an email is determined, the actual recipients of that email is generated based on generalized Zipf distribution. Finally, a threshold of 50 was used to qualify any pair of accounts to be in a same clique.

Five different attack strategies were simulated and tested. The first strategy sends a distinct email to all potential recipient addresses, one at a time. The second, third and fourth attack strategies send 20 attack emails, each with 2, 3 and 5 randomly chosen addresses. The value, 20, is of no concern here, since detection is based solely on each email. The last strategy sends a single email to all potential email addresses. During the detection phase, each attack email is examined and compared with previously classified enclave cliques. An alarm is triggered if the recipient list of the email is not a subset of any of the known enclave cliques. Finally, 10 replications of the simulation were run and the resulting statistics are listed below. As expected from intuition, it is easier to detect a potential anomaly if the size of the recipient list of the attack email is large.

Attack Strategy	Detection Rate
Send to all addresses, one at a time	0
Send many emails, each containing 2 random addresses	7 %
Send many emails, each containing 3 random addresses	17 %
Send many emails, each containing 5 random addresses	30 %
Send 1 email, containing all addresses	90 %

Table 2: Simulation of enclave cliques, with 5 attack strategies.

In terms of attack strategies used, the same 5 strategies are used, as in the case of enclave cliques. During the detection phase, each attack email is examined and compared with previously classified user cliques. An alarm is triggered if the recipient list of the email is not a subset of any of the known user cliques. Finally, 30 replications of simulation are run and the resulting statistics are listed below. As is also expected from intuition, it is easier to detect a potential anomaly if the size of the recipient list of the attack email is large.

Attack Strategy	Detection Rate
Send to all addresses, one at a time	0
Send many emails, each containing 2 random addresses	13 %
Send many emails, each containing 3 random addresses	49 %
Send many emails, each containing 5 random addresses	96 %
Send 1 email, containing all addresses	100 %

Table 3: Simulation of user cliques, with 5 attack strategies.

3. Supervised Machine Learning Models

In addition to the attachment and account frequency models, EMT includes an integrated supervised learning feature akin to that implemented in the MEF system previously reported in [12].

3.1 Modeling Malicious Attachments

MEF is designed to extract content features of a set of known malicious attachments, as well as benign attachments. The features are then used to compose a set of training data for a supervised learning program that computes a classifier. Figure 2 displays an attachment profile including a class label that is either “malicious” or “benign”.

MEF was designed as a component of MET. Each attachment flowing into an email account would first be tested by a previously learned classifier, and if the likelihood of “malicious” were deemed high enough, the attachment would be so labeled, and the rest of the MET machinery would be called into action to communicate the newly discovered malicious attachment, sending reports from MET clients to MET servers.

The core elements of MEF are also being integrated into EMT. However, here the features extracted from the training data include content-based features of email bodies (not just attachment features).

The Naïve Bayes learning program is used to compute classifiers over labeled email messages deemed interesting or malicious by a security analyst. The GUI allows the user to mark emails indicating those that are interesting and those that are not, and then may learn a classifier that is subsequently used to mark the remaining set of unlabeled emails in the database automatically.

A Naïve Bayes [5] classifier computes the likelihood that an email is interesting given a set of features extracted from the set of training emails that are specified by the analyst. In the current version of EMT, the set of features extracted from emails includes a set of static features such as domain name, time, sender email name, number of attachments, the MIME-type of the attachment, the likelihood the attachment is malicious, the size of the body, etc. Hot-listed "dirty words" and n-gram models and their frequency of occurrence are among the email message content-based linguistic features supported. We describe these next.

EMT is also being extended to include the profiles of the sender and recipient email accounts, and their clique behavior, as features for the supervised learning component.

3.2 Content-based Classification of Emails

In addition to using flow statistics about an email to classify an email message, we also use the email body as a content-based feature. There are two choices we have explored for features extracted from the contents of the email. One is the n-gram [16] model, and the other is a calculation of the frequency of a set of words [17].

An N-gram represents the sequence of any *n* adjacent characters or tokens that appear in a document. We pass an n-character wide window through the entire email body, one character at a time, and count the number of occurrences of each n-gram. This results in a hash table that uses the n-gram as a key and the number of occurrences as the value for one email; this may be called a *document vector.*

Given a set of training emails, we use the arithmetic average of the document vectors as the *centroid* for that set. For any test email, we compute the cosine distance [16] against the centroid created for the training set. If the cosine distance is 1, then the two documents are deemed identical. The smaller the value of the cosine distance, the more different the two documents are.

The formula for the cosine distance is:

$$D(x,y)=\sum_{j=1}^{J} x_j y_j /(\sum_{j=1}^{J} x_j^2 \sum_{k=1}^{J} y_k^2)^{1/2}=\cos\theta_{xy} \tag{11}$$

Here J is the total number of possible n-grams appearing in the training set and the test email. x is the document vector for a test email, and y is the centroid for the training set. x_j represents the frequency of the $\boldsymbol{j}^{\mathbf{th}}$ n-gram (the n-grams can be sorted uniquely) occurring in the test email. Similarly y_{k} represents the frequency of the k$^{\mathrm{th}}$ n-gram of the centroid.

A similar approach is used for the words in the documents instead of the n-grams. The classification is based on Naïve Bayes learning [17]. Given labeled training data and some test cases, we compute the likelihood that the test case is a member of each class. We then assign the test email to the most likely class.

These content-based methods are integrated into the machine learning models for classifying sets of emails for further inspection and analysis.

Using a set of normal email and spam we collected, we did some initial experiments. We use half of the labeled emails, both normal and spams, as training set, and use the other half as the test set. The accuracy of the classification using n-grams and word tokens varies from 70% to 94% when using different part as training and testing sets.

It's challenging to figure out spam using only content because some of the spam content are really like our normal ones. To improve the accuracy we also use weighted key words and stopwords techniques. For example, the spams also contain the words: free, money, big, lose weight, etc. Users can empirically give some key words and give higher weight to them when count their frequency. To avoid counting those common words user can give a stopwords list to delete those words first, which is a common approach in NLP. We are still doing further experiments and analysis on this content-based part.

Recent studies have indicated a higher accuracy using weighted naïve bayes on email contents. But as these methods become common, the spam writers are finding new and unusual ways to circumvent content analysis filters. These include, pasting in encyclopedia entries into the non rendered html section of the email, using formatting tricks to break apart words, using images, and redirection.

4. Discussion

4.1 Deployment and Testing

It is important to note that testing EMT and MET in a laboratory environment is not particularly informative, but perhaps suggestive of performance. The behavior models are naturally specific to a site or particular account(s) and thus performance will vary depending upon the quality of data available for modeling, and the parameter settings and thresholds employed.

It is also important to recognize that no single modeling technique in EMT's repertoire can be guaranteed to have no false negatives, or few false positives. Rather, EMT is designed to assist an analyst or security staff member architect a set of models whose outcomes provide evidence for some particular detection task. The combination of this evidence is specified in the alert logic section as simple Boolean combinations of model outputs; and the overall detection rates will clearly be adjusted and vary depending upon the user supplied specifications of threshold logic. In the following section, several examples are provided to suggest the manner in which an EMT and MET system may be used and deployed.

To help direct our development and refinement of EMT and MET, we deployed MET at two external locations, and the version of EMT described herein at one external location. (Because of the nature of the sensitivity of the target application, email analysis, both organizations prefer that their identities be left unknown; a topic we address in section 6.) It is instructive, however, to consider how MET was used in one case to detect and stop the spread of a virus.

The particular incident was an inbound copy of the "Hybris" virus, which appeared sometime in late 2000. Hybris, interestingly enough, does not attack address books of Window's based clients, but rather takes over Winsock to sniff for email addresses that it targets for its propagation. The recipient of the virus noticed the inbound email and a clear "clique violation" (the sender email address was a dead giveaway). Fortunately, the intended victim was a Linux station, so the virus could not propagate itself, and hence MET detected no abnormal attachment flow. Nonetheless, the intended victim simply used the MET interface to inspect the attachment he received, and noticed 4 other internal recipients of the same attachment. Each were notified within the same office and subsequently eradicated the message preventing the spread among other Windows clients within the office.

Simply having a tool to peer into the email behavior of the enclave provided a quick and effective response by one observer. The value of this type of technology to a large organization is noteworthy. A large enterprise that may manage 10's of thousands of hosts with several large NOC's would benefit greatly from early observation and detection and a quick response to avoid saturation of the enterprise. Indeed, the cost in time of staff to eradicate a virus on so many hosts makes a MET-like system a sensible protection mechanism, with demonstrable ROI. This, however, illuminates issues regarding response.

4.2 Response

There is also an interesting issue to consider regarding response to detected errant email events. MET is designed to detect errant email behavior in real time to prevent viral propagations (both inbound and outbound) from saturating an enterprise or enclave, to eliminate inbound SPAM, detect Spam bots utilizing (compromised) machines within an organization and other detectable misuses of email services. The behavior models employed compare recent email flows to longer-term profiles. The time to gather recent behavior statistics in order to detect an errant email event provides a window of opportunity for a viral or spam propagation to succeed in spreading to some number of hosts, until sufficient statistics have been gathered and tested in order to generate an alarm and subsequent corrective action taken. The number of successfully infected hosts targeted by the emails that leak out prior to detection, will vary depending upon a number of factors including network connectivity (especially the density of the cliques of the first victim).

This notion is discussed in a recent paper [15] that considers essentially propagation rates through a network defined by address book entries and the "social network links" defined therein among members of a network community, and subsequent strategies for managing and protecting address book data from the prying eyes of malicious code. It is important to note that viruses, such as Hybris, that do not attack address book data may follow a completely different propagation strategy not inferable from address book data. In such cases, EMT's account behavior models would likely detect viral spreads simply by noting abnormal account or attachment behaviors.

An alternative architectural strategy for a deployed MET/EMT system, reminiscent of router rate limiting technology to limit congestion, is to delay delivery of suspicious emails until such time as sufficient statistics have been gathered in order to determine more accurately whether a propagation is ongoing, or not. A recent paper by Williamson [14] notes the same strategy. Here, suspicious emails may be those deemed possibly anomalous to one or more of the behavior models computed by EMT and deployed in MET. Delaying delivery time would naturally reduce the opportunity for a propagation to saturate an environment, preventing fewer emails from leaking out. If no further evidence is developed suggesting an errant email event, the mail server would simply forward held emails. In the case where sufficient evidence reveals a malicious email event, such emails may be easily quarantined, followed by informative messages sent back to the client that originated those messages.

Even if the system incorrectly deems an email event as abnormal, and subsequently incorrectly quarantines emails, a simple means of allowing users to release these can be provided as a corrective action. Clearly, this may raise the "annoyance" level of the system forcing users to validate their email transmissions from time to time. The issue is of course complex requiring a careful design to balance between the level of security and protection an enterprise or enclave desires, the cost of repair of damage to errant email events, with the potential annoyance internal users may experience to prevent that damage.

We plan to study such issues in our future work after deploying MET in different environments.

5. Tradeoffs: Security, Privacy and Efficiency

For wide deployment of an MET/EMT system, users and providers must carefully study a number of tradeoffs regarding security and privacy of the user, and the value and cost of the protection mechanism such a system may provide.

The MET system can be configured to extract features from the contents of emails without revealing private information (e.g., aggregate statistics such as size, number of attachments, distribution of letters in the body, or even the number of occurrences of certain hot listed "dirty words".). Identity information may be hashed using a one way hash.

EMT's analysis may be entirely performed by hashing the identity of email accounts and thus hiding personally identifiable information while still providing the core functionality of MET. However, a large service provider may incur additional expense of its email services if fielding an MET system to protect their customers. The current set of models computed by EMT are a first cut using techniques that are as light weight to implement and as informative as possible. More advanced modeling techniques will likely incur additional cost. The cost may be offloaded from the provider's servers by pushing much of the computation and profiling mechanisms to the user's client machine further protecting the user's privacy. This would offload expense, while also limiting the information the email service provider may see (although of course they may still have the means of doing so).

Even so, ultimately the privacy of users must be considered, and is a matter of trust between the email service provider and the user (perhaps ultimately governed by law and regulation). The same trust relationship exists between banks and credit card users, and telecommunication companies and their customers, and such trust has benefited both parties; credit card fraud, for example, limits the liability and financial damage to both parties due to fraud. We believe a similar trust relationship between email providers and users will also benefit both parties.

As for security considerations, if EMT's analyses are performed at a user's client machine, the profile data of the user would be subject to attack on the client, allowing clever virus writers to thwart many of the core protection mechanisms EMT and MET provide. For example, if a clever virus writer attacks the EMT user profile data on the client platform (which is quite easy for some platforms), the virus may propagate itself in a manner that is entirely consistent with the user's normal behavior, thus, interestingly, hijacking or spoofing the user's behavior, rather than spoofing the user's identity. EMT and MET would therefore accomplish nothing. This would therefore argue for service providers to provide the needed resources in their email service to protect their customers (and hence to pass along these costs in their service fees; user's may be willing to pay more for security).

Deployment of an MET/EMT system in a corporate enterprise or enclave environment would likely have a completely different risk and cost assessment dynamic when considering the potential damage that may be incurred corporate-wide by virus attacks, or security policy violations (where corporate proprietary information may be revealed). It is apparent that the tradeoffs between privacy and security and cost are non-trivial, and ultimately are a matter of risk and benefit assessment between users and providers who will likely ultimately find an optimal balance.

6. Concluding Remarks

We are actively gathering large amounts of email data from volunteers within the Computer Science Department of Columbia University to test our hypotheses. We are aiming to develop a fully deployed system and further develop the range of analyses appropriate for such a behavior-based protection system.

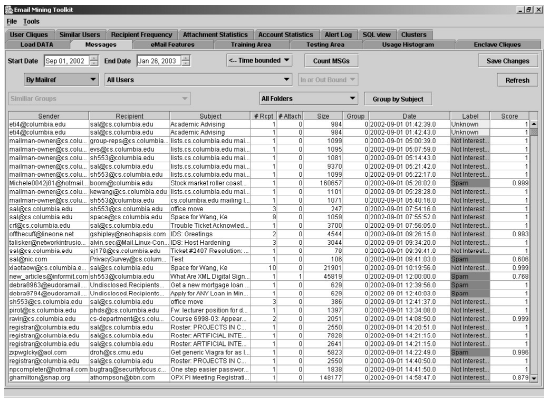
There are other uses of an MET/EMT system worth noting, metering of email and application layer profiling. There is some discussion in the popular press about the costs of spam emails and the burden placed upon ordinary users. Some analysts have proposed that email may one day not be a free service provided by ISP's, but rather a fee- or usage-based service provided by ISP in order to create an economic hurdle to spammers, and to of course recover their costs in handling massive amount of unwanted emails. Hence, each outbound email sent by a user account may incur a charge. If such schemes are indeed implemented, it likely would not be long before inbound received emails would also incur a charge (akin to "airtime" charges for inbound cell phone calls). Obviously, a system based upon MET/EMT would therefore become very useful for managing billing information and service charges (since a record of each email is maintained by MET), and is thus a natural extension to its core functionality.

It is also interesting to note that EMT also serves as a general strategy for an *Application-level Intrusion Detection System*. Email is after all an application. The same principles of behavior modeling and anomaly detection may also be applied to an arbitrary application. Thus, we are also exploring the application of a general form of EMT to model the behavior of large-complex distribution applications to detect errant misuses of the application.

7. References

1. M. Bhattacharyya, S. Hershkop, E. Eskin, and S. J. Stolfo. ``*MET: An Experimental System for Malicious Email Tracking.''* In Proceedings of the 2002 New Security Paradigms Workshop (NSPW-2002). Virginia Beach, VA, September, 2002.
2. Zhiqiang Bi, Christos Faloustos, Flip Korn. *"The DGX Distribution for Mining Massive, Skewed Data"*, 2001
3. C. Bron, J. Kerbosch, *Finding all cliques of an undirected graph*, Comm. ACM 16(9) (1973) 575--577.
4. E. Eskin, A. Arnold, M. Prerau, L. Portnoy and S. J. Stolfo. ``*A Geometric Framework for Unsupervised Anomaly Detection: Detecting Intrusions in Unlabeled Data*'', To Appear in Data Mining for Security Applications. Kluwer 2002.
5. George H. John and Pat Langley. Estimating continuous distributions in bayesian classifiers. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*. Pages 338-345, 1995
6. Wenke Lee, Sal Stolfo, and Kui Mok. ``Mining Audit Data to Build Intrusion Detection Models" *In Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining (KDD '98)*, New York, NY, August 1998
7. Wenke Lee, Sal Stolfo, and Phil Chan. ``*Learning Patterns from Unix Process Execution Traces for Intrusion Detection" AAAI* Workshop: AI Approaches to Fraud Detection and Risk Management*, July 1997*
8. MySQL, www.mysql.org, 2002.
9. W.Niblack et al. *"The QBIC project: querying images by content using color, texture, and shape".* In *Proceedings of the SPIE*, February 1993
10. Procmail,www.procmail.org,2002.
11. Sendmail,www.sendmail.org,2002.
12. Matthew G. Schultz, Eleazar Eskin, and Salvatore J. Stolfo. ``*Malicious Email Filter - A UNIX Mail Filter that Detects Malicious Windows Executables*." Proceedings of USENIX Annual Technical Conference - FREENIX Track. Boston, MA: June 2001.
13. J.R.Smith. *Integrated Spatial and Feature Image Systems: Retrieval, Compression and Analysis*. PhD thesis, Columbia University, 1997.
14. M.M. Williamson, *"Throttling viruses: Restricting propagation to defeat malicious mobile code",* Prof. ACSAC Security Conference, Las Vegas, NV, 2002.
15. M.E. Newman, S. Forrest and J. Balthrup, "*Email networks and the spread of computer viruses",* The American Physical Society, 2002.
16. Damashek, M.. *Gauging similarity with n-grams: language independent categorization of text*. Science, 267(5199):843--848, 1995.
17. Mitchell, Tom M. *Machine Learning*. McGraw-Hill, 1997, pg. 180-183.
18. R.V. Hogg, A.T. Craig, "Introduction to Mathematical Statistics", Prentice Hall, 1994, pg 293-301
19. M. Schonlau, W. DuMouchel, WH Ju, A.F.Karr, M theus and Y. Vardi, "Computer Intrusion Detecting Masquerades", Statistical Science, Vol. 16, 2001

Apendix A: ScreenShots of EMT.

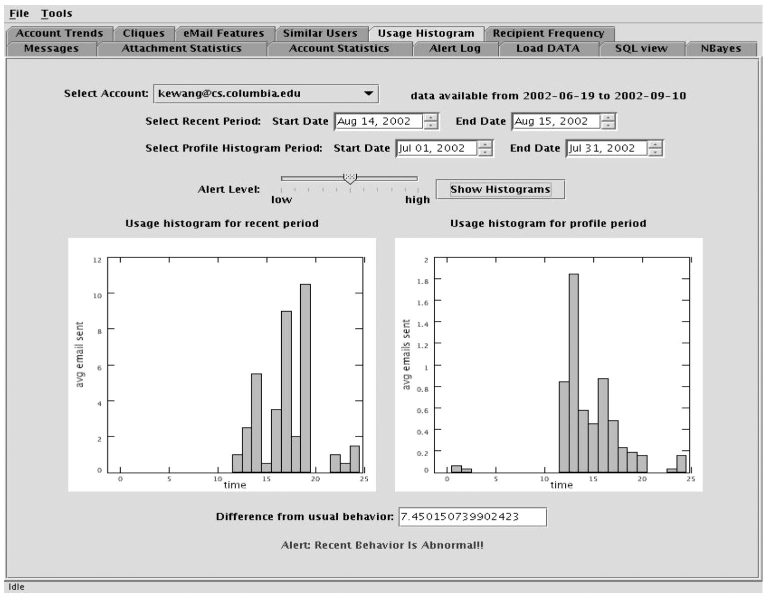


Start Date Sep 01, 2002 End Date Jan 26, 2003

By Mailref | All Users | All Folders

Sender	Recipient	Subject	# Rcpt	# Attach	Size	Group	Date	Label	Score
eti4@columbia.edu	sal@cs.columbia.edu	Academic Advising	1	0	984	0	2002-09-01 01:42:39.0	Unknown	1
eti4@columbia.edu	sal@cs.columbia.edu	Academic Advising	1	0	984	0	2002-09-01 01:42:43.0	Unknown	1
mailman-owner@cs.colu...	group-reps@cs.columbia...	lists.cs.columbia.edu mai...	1	0	1099	0	2002-09-01 05:00:39.0	Not Interest...	1
mailman-owner@cs.colu...	evs@cs.columbia.edu	lists.cs.columbia.edu mai...	1	0	1095	0	2002-09-01 05:07:59.0	Not Interest...	1
mailman-owner@cs.colu...	sh553@columbia.edu	lists.cs.columbia.edu mai...	1	0	1081	0	2002-09-01 05:14:43.0	Not Interest...	1
mailman-owner@cs.colu...	sal@cs.columbia.edu	lists.cs.columbia.edu mai...	1	0	9370	0	2002-09-01 05:21:42.0	Not Interest...	1
mailman-owner@cs.colu...	sh553@cs.columbia.edu	lists.cs.columbia.edu mai...	1	0	1099	0	2002-09-01 05:22:17.0	Not Interest...	1
Michele0042j81@hotmail...	boom@columbia.edu	Stock market roller coast...	1	0	160657	0	2002-09-01 05:28:02.0	Spam	0.999
mailman-owner@cs.colu...	kewang@cs.columbia.edu	lists.cs.columbia.edu mai...	1	0	1101	0	2002-09-01 05:28:28.0	Not Interest...	1
mailman-owner@cs.colu...	sh553@columbia.edu	cs.columbia.edu mailing l...	1	0	1071	0	2002-09-01 05:40:16.0	Not Interest...	1
sal@cs.columbia.edu	sh553@cs.columbia.edu	office move	3	0	247	0	2002-09-01 07:54:16.0	Not Interest...	1
sal@cs.columbia.edu	space@cs.columbia.edu	Space for Wang, Ke	9	0	1059	0	2002-09-01 07:55:52.0	Not Interest...	1
crf@cs.columbia.edu	sal@cs.columbia.edu	Trouble Ticket Acknowled...	1	0	3700	0	2002-09-01 07:56:05.0	Not Interest...	1
offthecuff@lineone.net	gshipley@neohapsis.com	IDS: Greetings	2	0	4544	0	2002-09-01 09:26:15.0	Not Interest...	0.993
talisker@networkintrusio...	alvin.sec@Mail.Linux-Con...	IDS: Host Hardening	3	0	3044	0	2002-09-01 09:34:20.0	Not Interest...	1
sal@cs.columbia.edu	sj178@cs.columbia.edu	Ticket #2407 Resolution: ...	1	0	78	0	2002-09-01 09:39:41.0	Not Interest...	1
sal@nic.com	PrivacySurvey@cs.colum...	Test	1	0	106	0	2002-09-01 09:41:03.0	Spam	0.606
xiaotaow@cs.columbia.e...	sal@cs.columbia.edu	Space for Wang, Ke	10	0	21901	0	2002-09-01 10:19:56.0	Not Interest...	0.999
new_articles@informit.com	sh553@columbia.edu	What Are XML Digital Sign...	1	1	45819	0	2002-09-01 12:00:00.0	Spam	0.768
debra8963@eudoramail....	Undisclosed.Recipients...	Get a new mortgage loan ...	1	0	629	0	2002-09-01 12:39:56.0	Spam	1
debra9794@eudoramail....	Undisclosed.Recipients...	Apply for ANY Loan in Min...	1	0	629	0	2002-09-01 12:40:03.0	Spam	1
sh553@cs.columbia.edu	sal@cs.columbia.edu	office move	3	0	386	0	2002-09-01 12:41:37.0	Not Interest...	1
pirot@cs.columbia.edu	phds@cs.columbia.edu	Fw: lecturer position for d...	1	0	1397	0	2002-09-01 13:34:08.0	Not Interest...	1
ravir@cs.columbia.edu	cs-department@cs.colu...	Course 6998-03: Appear...	2	0	2051	0	2002-09-01 14:08:50.0	Not Interest...	0.999
registrar@columbia.edu	sal@cs.columbia.edu	Roster: PROJECTS IN C...	1	0	2550	0	2002-09-01 14:20:51.0	Not Interest...	1
registrar@columbia.edu	sal@cs.columbia.edu	Roster: ARTIFICIAL INTE...	1	0	7828	0	2002-09-01 14:21:15.0	Not Interest...	1
registrar@columbia.edu	sal@cs.columbia.edu	Roster: ARTIFICIAL INTE...	1	0	2641	0	2002-09-01 14:21:15.0	Not Interest...	1
zxpwglcky@aol.com	droh@cs.cmu.edu	Get generic Viagra for as l...	1	0	5823	0	2002-09-01 14:22:49.0	Spam	0.996
registrar@columbia.edu	sal@cs.columbia.edu	Roster: PROJECTS IN C...	1	0	2550	0	2002-09-01 14:40:50.0	Not Interest...	1
npcompleter@hotmail.com	bugtraq@securityfocus.c...	One step easier passwor...	1	0	1838	0	2002-09-01 14:41:50.0	Not Interest...	1
ghamilton@snap.org	athompson@bbn.com	OPX PI Meeting Registrati...	1	0	148177	0	2002-09-01 14:58:47.0	Not Interest...	0.879

Fig. 3. The Message Tab



Select Account: kewang@cs.columbia.edu data available from 2002-06-19 to 2002-09-10

Select Recent Period: Start Date Aug 14, 2002 End Date Aug 15, 2002

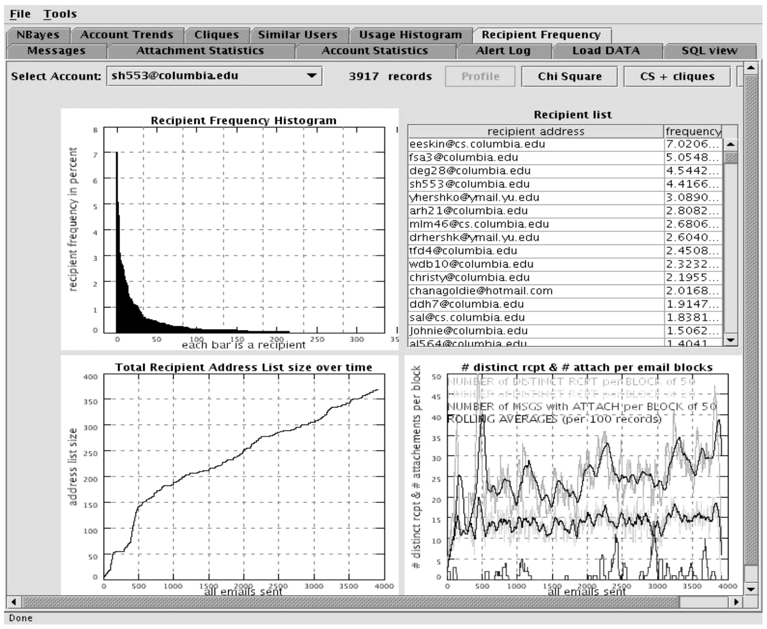
Select Profile Histogram Period: Start Date Jul 01, 2002 End Date Jul 31, 2002

Alert Level:

Difference from usual behavior: 7.450150739902423

Alert: Recent Behavior Is Abnormal!!

Fig. 4. Abnormal Behaviour Detected

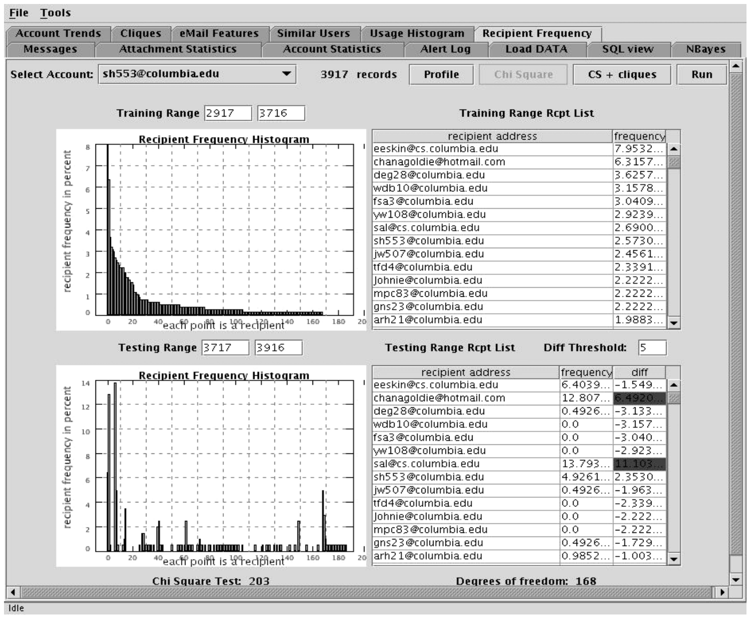

File Tools
NBayes
Account Trends
Cliques
Similar Users
Usage Histogram
Recipient Frequency
Messages
Attachment Statistics
Account Statistics
Alert Log
Load DATA
SQL view
Profile
Chi Square
CS + cliques
Recipient Frequency Histogram
recipient frequency in percent
each bar is a recipient
Total Recipient Address List size over time
address list size
all emails sent
# distinct rcpt & # attach per email blocks
# distinct rcpt & # attachements per block
NUMBER of DISTINCT RCPT per BLOCK of 50
NUMBER of MSGS with ATTACH per BLOCK of 50
ROLLING AVERAGES (per 100 records)
all emails sent
Done

Select Account: sh553@columbia.edu 3917 records

Recipient list

recipient address	frequency
eeskin@cs.columbia.edu	7.0206...
fsa3@columbia.edu	5.0548...
deg28@columbia.edu	4.5442...
sh553@columbia.edu	4.4166...
yhershko@ymail.yu.edu	3.0890...
arh21@columbia.edu	2.8082...
mlm46@cs.columbia.edu	2.6806...
drhershk@ymail.yu.edu	2.6040...
tfd4@columbia.edu	2.4508...
wdb10@columbia.edu	2.3232...
christy@columbia.edu	2.1955...
chanagoldie@hotmail.com	2.0168...
ddh7@columbia.edu	1.9147...
sal@cs.columbia.edu	1.8381...
Johnie@columbia.edu	1.5062...
al564@columbia.edu	1.4041

Fig. 5. Recipient Frequency Plots



Select Account: sh553@columbia.edu 3917 records

Training Range 2917 3716

Training Range Rcpt List

recipient address	frequency
eeskin@cs.columbia.edu	7.9532...
chanagoldie@hotmail.com	6.3157...
deg28@columbia.edu	3.6257...
wdb10@columbia.edu	3.1578...
fsa3@columbia.edu	3.0409...
yw108@columbia.edu	2.9239...
sal@cs.columbia.edu	2.6900...
sh553@columbia.edu	2.5730...
jw507@columbia.edu	2.4561...
tfd4@columbia.edu	2.3391...
Johnie@columbia.edu	2.2222...
mpc83@columbia.edu	2.2222...
gns23@columbia.edu	2.2222...
arh21@columbia.edu	1.9883...

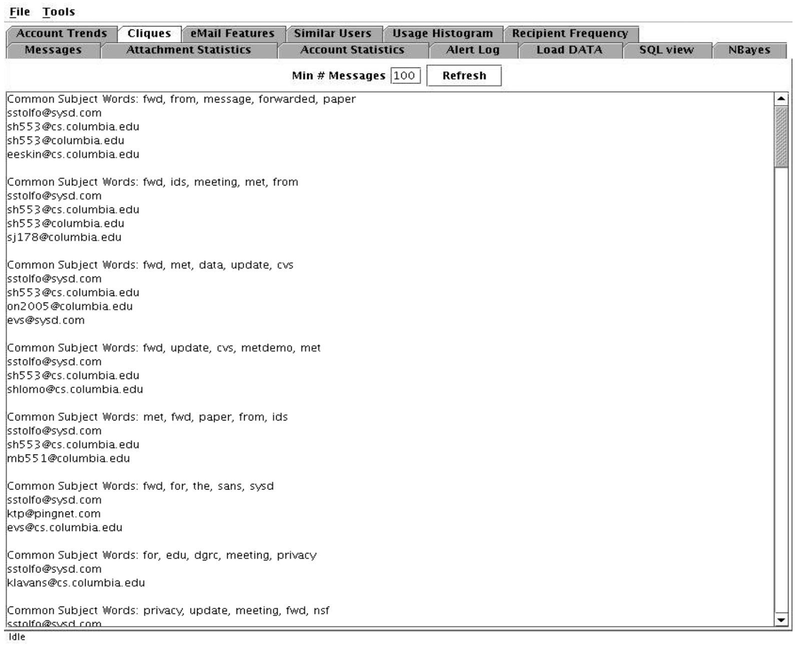
Testing Range 3717 3916

Testing Range Rcpt List Diff Threshold: 5

recipient address	frequency	diff
eeskin@cs.columbia.edu	6.4039...	-1.549...
chanagoldie@hotmail.com	12.807...	6.4920...
deg28@columbia.edu	0.4926...	-3.133...
wdb10@columbia.edu	0.0	-3.157...
fsa3@columbia.edu	0.0	-3.040...
yw108@columbia.edu	0.0	-2.923...
sal@cs.columbia.edu	13.793...	11.103...
sh553@columbia.edu	4.9261...	2.3530...
jw507@columbia.edu	0.4926...	-1.963...
tfd4@columbia.edu	0.0	-2.339...
Johnie@columbia.edu	0.0	-2.222...
mpc83@columbia.edu	0.0	-2.222...
gns23@columbia.edu	0.4926...	-1.729...
arh21@columbia.edu	0.9852...	-1.003...

Chi Square Test: 203 Degrees of freedom: 168

Fig. 6. Chi Square Test of recipient frequency



Min # Messages 100

```
Common Subject Words: fwd, from, message, forwarded, paper
sstolfo@sysd.com
sh553@cs.columbia.edu
sh553@columbia.edu
eeskin@cs.columbia.edu

Common Subject Words: fwd, ids, meeting, met, from
sstolfo@sysd.com
sh553@cs.columbia.edu
sh553@columbia.edu
sj178@columbia.edu

Common Subject Words: fwd, met, data, update, cvs
sstolfo@sysd.com
sh553@cs.columbia.edu
on2005@columbia.edu
evs@sysd.com

Common Subject Words: fwd, update, cvs, metdemo, met
sstolfo@sysd.com
sh553@cs.columbia.edu
shlomo@cs.columbia.edu

Common Subject Words: met, fwd, paper, from, ids
sstolfo@sysd.com
sh553@cs.columbia.edu
mb551@columbia.edu

Common Subject Words: fwd, for, the, sans, sysd
sstolfo@sysd.com
ktp@pingnet.com
evs@cs.columbia.edu

Common Subject Words: for, edu, dgrc, meeting, privacy
sstolfo@sysd.com
klavans@cs.columbia.edu

Common Subject Words: privacy, update, meeting, fwd, nsf
sstolfo@sysd.com
```

Fig. 7. Clique generation for 100 Messages