Detecting Viral Propagations Using Email Behavior Profiles

SALVATORE J. STOLFO, WEI-JEN LI,
SHLOMO HERSHKOP, KE WANG, CHIA-WEI HU, OLIVIER NIMESKERN
Columbia University

The Email Mining Toolkit (EMT) is a data mining system that computes *behavior profiles or models* of user email accounts. These models may be used for a variety of forensic analyses and detection tasks. In this paper we focus on the application of these models to detect the early onset of a viral propagation without "content-based" (or signature-based) analysis in common use in virus scanners. We present several experiments using real email from 15 users with injected simulated viral emails and describe how the combination of different behavior models improves overall detection rates. The performance results vary depending upon parameter settings, approaching 99% true positive(TP) (percentage of viral emails caught) in general cases and with 0.38% false positive(FP) (percentage of emails with attachments that are mislabeled as viral). The models used for this study are based upon volume and velocity statistics of a user's email rate and an analysis of the user's (social) *cliques* revealed in their email behavior. We show by way of simulation that virus propagations are detectable since viruses may emit emails at rates different than human behavior suggests is normal, and email is directed to groups of recipients that violates the user's typical communication with their social groups.

Categories and Subject Descriptors: Security and privacy
General Terms: Email Virus Propagations, Behavior Profiling
Additional Key Words and Phrases: Anomaly Detection

---

1. INTRODUCTION

Email is a common method of choice for the propagation of viruses and worms. Typically, a virus will extract email addresses in an infected computer and send copies of itself to some or all of these addresses. These addresses may be obtained from many sources, such as the address book, socket-layer sniffing, or a locally stored email archive. Virus scanners are signature-based, which means that they use a collection of byte sequences or embedded strings to identify known malicious attachments. If a virus scanner's signature database does not contain a signature for a malicious program, the virus scanner is unable to detect or protect against that malicious program. In general, to mitigate against this *false negative problem*, virus scanners require frequent updating of signature databases, otherwise the scanners become useless to detect new attacks. Similarly, security patches protect systems only when they have been written, distributed and applied to host systems in response to known attacks. Until then, systems remain vulnerable and attacks are able to propagate widely and rapidly over the internet.

For example, the SoBig.F propagation that occurred in the late summer of 2003 spread rapidly across the internet using a high speed spam-based propagation strategy. It took several days before an effective signature was available for distribution to locally

update virus scanners to stop this virus. During this period of time no signature-based filters were available and SoBig.F flooded the internet causing remarkable damage and expense. It is this case of a new viral attack that is the subject matter of this paper.

Furthermore, virus writers have demonstrated their continual cleverness by thwarting virus scanners with strategies that defy signature-based detection. Stopping a polymorphic virus that uses several points of entry, and that also "morph" the contents of the virus in various ways, can be a daunting task using traditional signature-based virus scanning methods alone.

Our core premise is that viral propagations fundamentally behave differently than typical human user email behavior. Thus, the idea is to train or learn a detector that models a user's email behavior, and then to apply these models to the email flow of a user's account to detect abnormal or anomalous email flows that may indicate a viral propagation has been initiated. Much prior work on anomaly detection systems has been reported in the literature to solve the false negative problem of signature-based detection systems in intrusion detection systems. Here, we apply this methodology to email.

In this paper, we describe the Email Mining Toolkit (EMT), a data mining system that computes a large number of different profiles over email archives, and demonstrate how its model outputs are used as an email anomaly detection system. We believe EMT demonstrates a solution to raise the bar of protection to detect and extinguish viral propagations as early as possible until new signatures are developed and deployed. Consider the following observations.

First, viral email propagations involve an email sent to or from a victim email account with either an attachment or with something equivalent to an HTML page in the text body. In the former case, the user will have to run the executable that launches a virus directly, or invoke a program that uses the seemingly innocent data file that exploits the weakness of the program that makes use of it. In the latter case, the user may simply click on an innocent appearing URL that may start the download and execution of malware.

Second, it is highly unlikely a virus will propagate itself with only one or a few emails. This is because usually viruses are designed to infect as many computers as possible in a short period of time. Otherwise, they would be stopped long before they have a chance to inflict damage to many systems. Creating many copies ensures the virus will propagate quickly and widely. We conjecture that the frequency of emissions of emails during a viral propagation will be substantially different than the victim user's typical email rate (both inbound and outbound).

Finally, a virus is ignorant of their victim's behavior, in the sense that it does not know the relationship between a user and those with whom they communicate. For example, a user would be unlikely to send an email, or many copies of an email, to a large number of recipients among the user's separate *social cliques.* Instead, a virus may use simple hard-coded rules in deciding whom to propagate to, violating the user's typical behavior in sending emails among his/her social cliques. These observations suggest that viral propagations may be detected by profiling email behavior and using the user's behavior models to detect the onset of a propagation.

Behavior-based detection is not a new concept. Credit card fraud detection [40] is perhaps the best example of a widely deployed security system that depends upon profiling behavior of credit card users. We posit that a similar approach directed towards "email transactions" will provide comparable broad-based security without requiring a complete overhaul of email protocols and server implementations.

By measuring behavior of individual email users over time using different statistics and profiling techniques, and the probabilities associated with these statistics, we wish to correlate as much evidence from multiple models to accurately detect errant or malicious email while minimizing false alarms.

Three types of behavior-based models are examined in detail: *user cliques* and the *Hellinger distance and cumulative distribution* models. The user clique model profiles a user's communication groups that naturally occur in their email communication history (for example, colleagues, family members, friends, etc). The Hellinger distance model profiles the distribution of the frequency of communication of the user, and the variability of that frequency, between a user and his/her correspondents. (Interestingly, the analysis we have performed on the email archives of many volunteer email users reveals that email communication behavior follows a Zipf distribution, the same distribution that models the naturally occurring frequency distribution of words in natural language.)

The cumulative distribution model profiles the (daily) rate at which a user sends emails to distinct parties in sequential order. These three models are more or less orthogonal to each other and they are combined together to form a hybrid model that yields very good detection performance.

We describe a number of experiments using an email archive collected from 15 volunteers. Experiments were performed by injecting this archive with viral emails using the virus' propagation strategies. The viruses were not run to avoid potential damage (and litigation).

To measure and compare the detection rate of the combined behavior models, there is no baseline to study other than typical COTS virus scanners. We take the point of view that a virus scanner will have a 100% True Positive rate and 0% False Positive rate for any virus for which a signature exists; but it will also exhibit a 0% TP rate for any "new" virus for which a signature has not yet been developed and deployed. It is these "new" viruses that cause damage, and that we use under simulation to test the performance of EMT. Of particular importance here is the tradeoff between EMT's TP rate (detecting new viruses) and its FP rate, i.e. the percentage of emails deemed viral by EMT but which are indeed non-viral. We demonstrate this performance using ROC curves and evaluate the "annoyance rate" EMT may exhibit in generating false alarms.

The results show that EMT's behavior models are an effective detection system. Its best performance in detecting inbound viral propagations over all users is 99% TP and 0.38% FP, while its best performance for detecting outbound viral propagations from an account is 99% TP and 0.9% FP. EMT also exhibited its worst performance for inbound viral detection at 70% TP and 0.38% FP (outbound detection is 60% TP and 0.9% FP) if the viral propagation is a very slow, stealthy propagation with one viral email delivered every 5 days.

Thus, fast propagations are easy to detect by observing anomalous email flows that are inconsistent with a user's normal email behavior. Slow and stealthy propagations are, however, hard to detect. In all these cases, detection is achieved by observing email flows and user behaviors, not by analyzing the contents of the email traffic.

The rest of this paper is organized as follows. Section 2 describes related research on anomaly detection in intrusion detection systems from which this work was originally conceived. Section 3 provides an overview of EMT's implementation and database schema. We then describe the particular models supported by EMT and used in this study. Section 4 describes the technique to compute user's (social) cliques revealed in their email behavior, and an independent test of this model shows that viral emails violate a user's email clique behavior. Section 5 details a frequency-based model of the user's typical recipients, and likewise a simulated test showing how this model may detect viral email propagations. Section 6 presents an experiment where these two models are combined for better detection performance and reduced false alarm rates. Section 7 introduces another statistical distribution and a final performance evaluation for all three models producing our best results. Section 8 concludes the paper with a discussion of future research.

2. RELATED REASEARCH

EMT is a data mining and profiling system applied to email data to detect anomalous email behavior. Our prior work on the Malicious Email Tracking (MET) system focused on modeling the behavior of attachments, and attachment flows in email [3] among participating sites either within an enclave or across sites within an enterprise. The concept behind MET is to measure the statistics of attachment flows across a mail server and to detect a viral propagation as an anomaly (e.g. a "burst" or a "high host saturation") in this attachment flow. Thus, MET is best viewed as an anomaly detector for flows.

Anomaly detection systems were first proposed by Denning [8] for intrusion detection, and later implemented in NIDES [15] to model normal network behavior in order to detect deviant behavior that may correspond to an attack against a network computer system. W. Lee et al. describe a framework and system for auditing and data mining and feature selection for intrusion detection. This framework consists of classification, link analysis and sequence analysis for constructing intrusion detection models. [19, 21]

A variety of other work has appeared in the literature detailing alternative algorithms to establish normal profiles, applied to a variety of different audit sources, some specific to user commands for masquerade detection [29], others specific to network protocols and LAN traffic for detecting denial of service attacks [24, 34] or Trojan execution, or application or system call-level data for malware detection [13, 36], to name a few.

A variety of different modeling approaches have been described in the literature to compute baseline profiles. These include probabilistic or statistical distributions over temporal data [2, 9, 39], supervised machine learning [11, 20] and unsupervised cluster-based algorithms [10]. Some approaches consider the correlation of multiple models [11, 36].

In general, in the case that an audit source is a stream or temporally ordered data, a variety of models may be defined for an audit source and a detector may be computed to generate an alarm if a violation is observed based upon volume and velocity statistics. Volume statistics represent the amount of data observed per unit of time, while velocity statistics model the changes in frequency and magnitude of the data over time. In our EMT work, for example, we compute volume statistics, such as the "number of distinct recipients of emails" and the "cumulative number of emails with attachments" sent sequentially. EMT also computes the Hellinger distance of the recipient frequency as an example of velocity statistics. These two kinds of statistics represent one aspect of a user's behavior profile and are used to detect the abnormal behavior indicative of virus and spam emails.

We are not aware of any prior work devoted to anomaly detection applied to email audit streams other than MET. However, recent work by Forrest [27] and HP [38] and social scientists at Columbia University [37] analyze email account connectivity for various purposes. In Forrest's case, they consider email accounts linked in a graph as defined by address books to measure network density specifically to provide guidance on address book management. They note that viral propagations will spread fast among accounts whose address books are deemed "dense" from a graph theoretic point of view. The HP and Columbia social science work are similar to our work on cliques. In these two pieces of work, the communication density and flow within an organization is studied to understand the effectiveness of communication with an organization. In the case of the Columbia social science work, they seek to answer the question whether 6 levels of indirection indeed separate any two people within email communication.

3. EMT – EMAIL MINING TOOLKIT

In order to develop behavior-based methods for email security, we have implemented the Email Mining Toolkit (EMT) [32]. This toolkit is useful for report generation and summarization of email archives, as well as for detecting email security violations when incorporated with a real-time violation detection system, such as the MET system [3]. EMT is presently implemented as a forensic analysis tool applied to email logs; it is being upgraded to allow uploading of its computed models for real-time detection either as a client-side detector (when integrated with a mail program) or within MET as a server-side detector.

3.1 Schema

The main database embedded within EMT consists of three tables: the Email, Message, and Kwords tables, which represent email in the database. Below is a brief overview of these tables.

3.1.1 Email Table

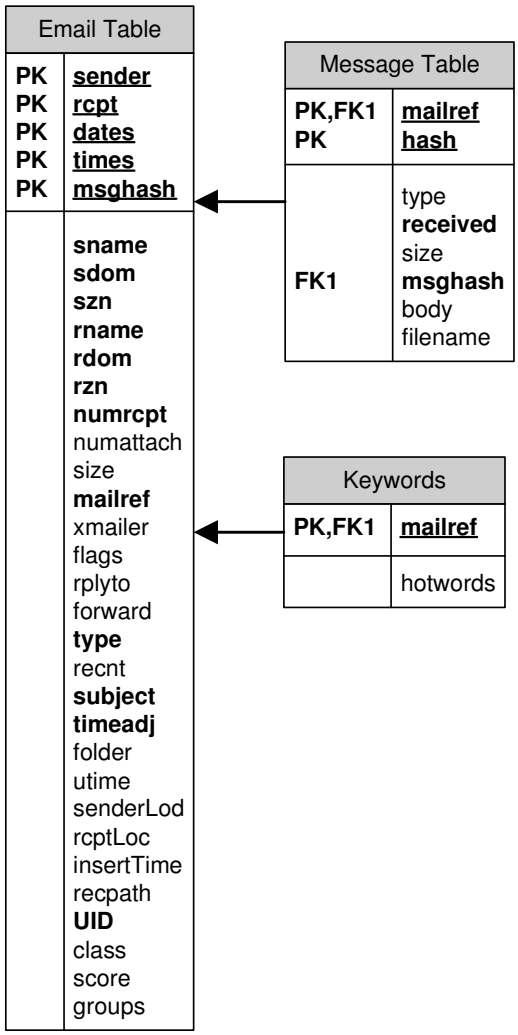


Fig. 1. Main database schema.

The email table is the primary collection of information about an email. Information about the sender, recipient, date, time when the email was moved and its contents including attachments are stored here. If a single email has many recipients, the table includes a row for each recipient (as if the email were viewed by a distinct sender/recipient pair). The reason for this is that, conceptually, there is no difference between sending an email to many people at once and sending it to each person individually.

Each email has two unique identifiers in the schema. Each row in the table has a unique field that indicates the e-mail's unique identification number (UID). In addition, each processed email has a unique ID called the 'mailref'. Some email records may share a single 'mailref' if they were created from a single email that had multiple recipients.

The kwords table collects keyword statistics about the content of an email. A keyword file can be specified and the table saves the frequency of keywords found with each unique 'mailref'. In practice this table is updated when keyword functionality is enabled either when the e-mail's are parsed, or during EMT operations. This feature is used for various content-based analyses available in EMT (for example, similarity of messages and their grouping, or statistical analyses using a "bag of words" feature for author identification, etc.).

3.2 EMT Modeling Features

EMT contains a large collection of (statistical) modeling features that may be combined for various detection tasks. Figures 2 and 3 show two screen shots of EMT's GUI display windows, and a set of tabs to access these features.

EMT is implemented in Java providing a GUI implementing an interface to an underlying database application. The data can reside in any SQL RDBMS. EMT is also provided with a set of parsers written in Perl that can read email files from a variety of formats (mbox, nsmail, Outlook and Lotus are all supported) and insert data into the underlying database. Each row of this data base is a detailed record of an email from which a variety of statistical analyses may be applied. Most of EMT's statistical models are computed by SQL commands against this database. Thus, EMT has been designed for scalability to large email archives, and generality to other communication mediums. A version of EMT that analyzes Instant Messaging traffic has also been implemented.

For this paper, we focus primarily on testing three behavior models computed by EMT to detect the onset of viral propagations. We first describe EMT's analysis of group communication behavior.

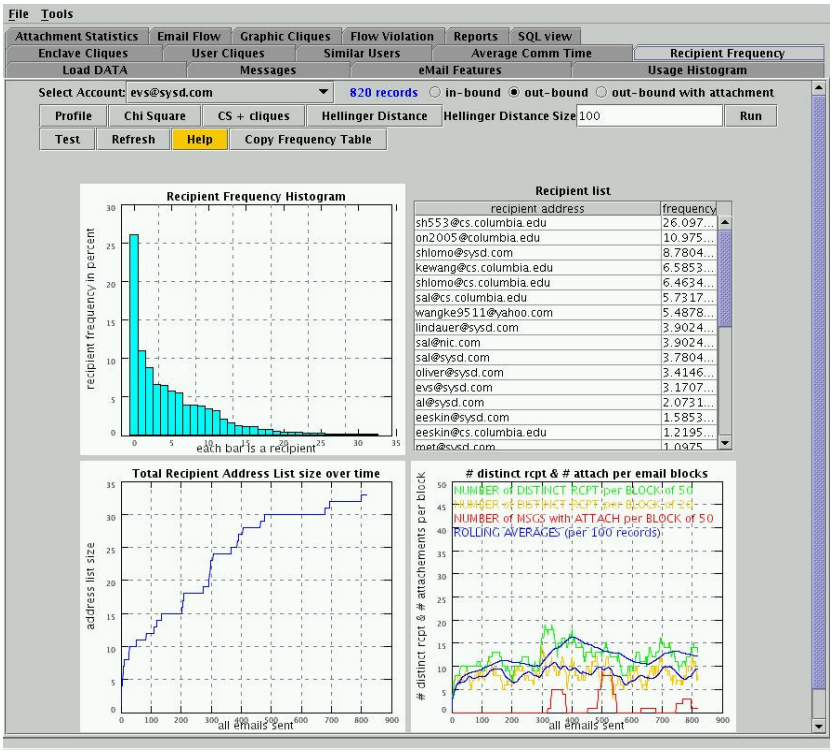


Fig. 2. Recipient/Sender Frequency.

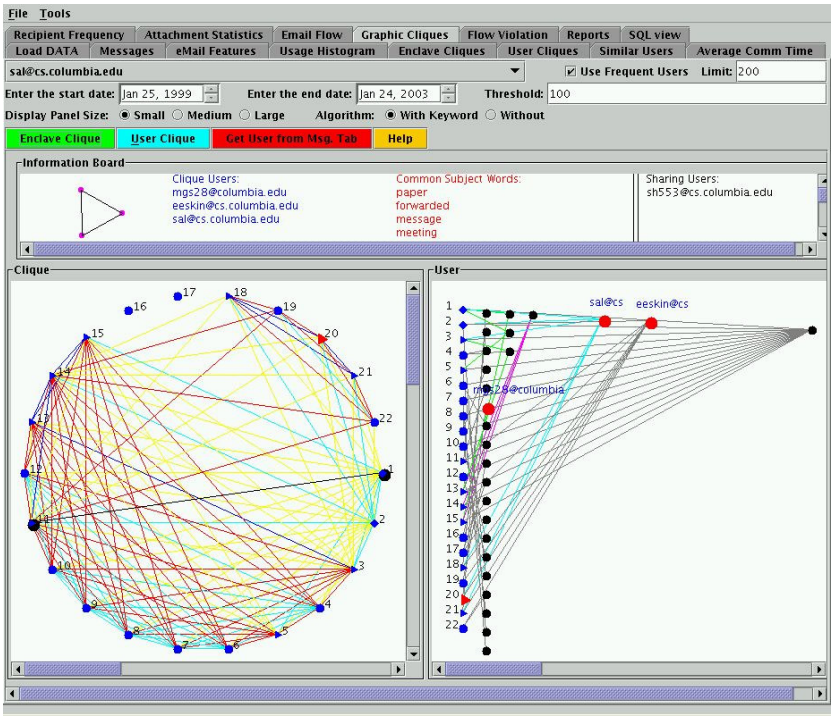


Fig. 3. Groups and cliques of users.

4. GROUP COMMUNICATION MODELS: CLIQUES

In order to study email flows between groups of users, EMT computes a set of *cliques* in an email archive. We seek to identify clusters or groups of related email accounts that participate with each other in common email communications, and then use this information to identify unusual email behavior that violates typical group behavior. For example, intuitively it is unlikely that a user will send a distinct message to a spouse, a boss, "drinking buddies" and church elders all appearing together as recipients of the same message (whether delivered in one email, or a series of emails). Of course this is possible, but it is rather unlikely. A virus attacking a user's address book at random would surely not know these social relationships and the typical communication pattern of the victim. Hence it would violate the users' group behavior profile if it propagated itself in violation of the user's *social cliques*.

Clique violations may also indicate email security policy violations internal to a secured enclave. 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, and probably unwise, if members of the marketing department, and HR services would likewise receive these attachments. We can infer the composition of related groups by analyzing normal email flows to compute the naturally occurring cliques, and use the learned cliques to alert when emails violate that clique behavior.

Conceptually, two broad types of cliques can be extracted from user email archives: *user cliques* and *enclave cliques*. In simple terms, user cliques can be computed by analyzing the email history of only a single user account, while enclave cliques are social groups that emerge as a result of analyzing traffic flows among a group of user accounts within an enclave. In this paper, we utilize only User Clique models, leaving the analysis of enclave cliques to a future paper.

4.1. User Cliques

We model the collection of recipients in a single email as a set, and summarize these sets and their dynamics. This information is used to detect abnormal emails that violate the user's clique behavior.

Formally, email communication can be captured by a directed graph $G(V,E)$ with the set of nodes, V, corresponds to individual email accounts. A directed edge, e_{12}, exists if v_1 sends an email to v_2. Viewed in this way, cliques are a certain pattern in this graph that we characterize and use as norms of communication behavior. (EMT also

provides an “enclave clique” feature that implements the Bron-Kerbosch clique finding algorithm to compute all connected components in this graph.)

Aside from the graphical view, the user clique model is best described in terms of item sets. An item set is a set of items associated with a transaction, such as a single purchase at a supermarket. The goal of analyzing item sets is to extract useful association rules of how items appear together. This problem has been studied in the data mining and database community and is of great commercial interest for its wide range of applications and potential predictive value that can be derived [1].

In the context of mining email, an email can be viewed as a transaction that involves multiple accounts, including a sender (in the FROM field) and recipient(s) in the (TO, CC and, BCC fields). If we discover the rules governing the co-appearance of these addresses, we could then use these rules to detect emails that violate these patterns. Suspicious emails may then be examined further by other models to confirm or deny that they are malicious.

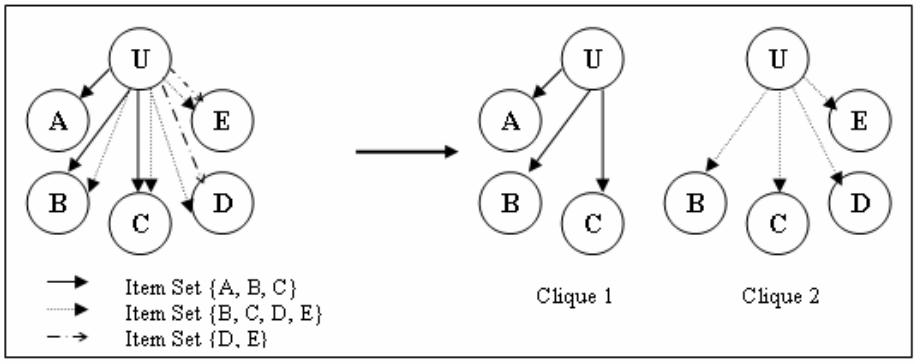


Fig. 4. Three item sets from account U: {A, B, C}, {B, C, D, E} and {D, E}. The first two sets share two nodes and the last set is subsumed by the second set. The resulting user cliques are {A, B, C} and {B, C, D, E}.

The recipient list of a single email can be viewed as a clique associated with the FROM account. However, using this set (or item set) directly is problematic for two reasons. First, a single user account would contain a large number of such sets and enumerating them for real-time reporting or detection tasks would be undesirable. Second, some of these sets are duplicates or subsets of one another and it would be difficult to use them directly for any purpose. For these reasons, we define a *user clique* as a set of recipients that cannot be subsumed by another set. Thus, we compute the most frequent email item sets that are not subsumed by another larger item set. Naturally, a single user will have a relatively small number of user cliques. As an example, suppose a user has in his/her sent-folder four emails with the following recipient lists: {A, B, C}, {A, B, C},

{A, B}, and {A, B, D}. The user cliques belonging to this user would be {A, B, C} and {A, B, D}. Note that duplicate user cliques are removed, as it does not contribute useful information.

Once these sets are derived off-line by analyzing a user's "profile" period, we inspect each email sent from the user's account in a subsequent "test" period of time to determine if there is a *clique violation* – i.e. the recipient list is inconsistent with the user's cliques. An email sent from a user is regarded as inconsistent with the user's cliques if its recipient list is not a subset of any user cliques belonging to that user.

The usefulness of this model depends not only on how quickly new groups of recipients form over time but also on how it is combined with other models. Installing a monitoring tool using this model on a new account or an account that is constantly communicating with new groups may cause too many false alarms and thus render the model useless. However, this very behavior is indicative of user email usage patterns and thus can be turned into a feature that characterizes user behavior.

Although the dynamics of clique formation [7] (and expiration) is implemented in EMT, for the present paper we shall ignore the dynamics of clique formation to explore the utility of the base user clique model. Computing the set of "static cliques" is sufficiently informative for the purpose at hand; this model provides useful evidence of a viral propagation launched from a user's account.

Notice that if a user ever sends a single broadcast email to everyone in their address book, there would be only one user clique remaining in the model for that user. This would render the model almost useless for virus detection task because no clique violation is possible as long as a user does not communicate with someone new. In practice, however, this scenario is highly unlikely to happen. We illustrate this point by examining the communication patterns of 15 users in our database. We show that most of the time, a user will send a single email to less than 10% of the people in their address book. For an account with a small address book, a single email could cover 20%, 30% or an even higher percentage of the address book. As we can see from Table 1, the probability of an email covering a given range of percentages of an address book decreases quickly as the percentage range increases. In fact, none of the 15 users ever sent a broadcast email to everyone in his/her address book.

Table 1. Statistic of the percentage of an address book covered by a single email, broken down for each user and the average case.

User	# of distinct addresses	≤ 10%	10-20%	20-30%	30-40%	40-50%	50-60%	60-70%	70-80%	80-90%	≥90 %
1	324	1	0	0	0	0	0	0	0	0	0
2	1308	1	0	0	0	0	0	0	0	0	0
3	38	0.46	0.49	0.04	0	0	0	0	0	0	0
4	144	0.96	0.01	0.01	0.00	0	0	0	0	0	0
5	26	0	0.74	0.04	0.09	0.06	0.02	0.01	0.02	0	0
6	105	0.95	0.04	0.01	0	0	0	0	0	0	0
7	64	0.98	0.01	0	0	0	0	0	0	0	0
8	92	0.95	0.05	0	0	0	0	0	0	0	0
9	43	0.70	0.15	0.11	0.04	0.01	0	0	0	0	0
10	24	0.54	0.12	0.25	0.05	0.02	0.01	0	0	0	0
11	75	0.91	0.09	0.01	0	0	0	0	0	0	0
12	1231	1	0	0	0	0	0	0	0	0	0
13	231	1	0	0	0	0	0	0	0	0	0
14	368	1.00	0.00	0	0	0	0	0	0	0	0
15	568	1	0	0	0	0	0	0	0	0	0
Avg	291	0.83	0.11	0.03	0.01	0.00	0.00	0.01	0.01	0	0

4.2. Test of Simulated Viruses

Here we evaluate the utility of user clique violations (independent of other modeling techniques) for viral propagation detection. We simulate viruses by inserting "dummy" emails into an email archive following a propagation strategy that has been observed from numerous real viruses seen in the wild. The first 80% of emails sent from each account is deemed the profile period used for deriving user cliques associated with that account. The remaining 20% of the emails are used during the testing phase where the dummy emails simulating the propagation are inserted.

For this simulation, it is not critical exactly when and how often viral emails are sent out. That is, we ignore the propagation rate entirely; determining whether or not a recipient set violates existing user cliques is independent of the timing of the email in question. However, during the simulation/test phase, user cliques are updated on a daily basis and the timing of email is affected slightly. Such effects are still more or less negligible, as having viral emails that are sent later in time is tantamount to having a longer training phase and a shorter test phase.

In terms of modeling attack strategies, we test the effectiveness of the user clique violation model against various sizes of a viral email recipient list. For illustrative purposes, we assume that a virus would fetch email addresses from the address book of an infected user to propagate itself. In reality, email addresses could be obtained via others means, such as scanning the inbox, sent folder and email archives. Without loss of generality, the simulation has the virus propagating itself to recipients chosen at random.

However, the usefulness of user-clique violation detection in practice depends on how a virus obtains the target email addresses. For example, a virus obtaining addresses from an inbox and replying to respective senders and everyone else in the message may not be detected easily, depending upon how compatible they are with existing user cliques. (This implies that the virus would imitate or mimic the user's behavior; avoiding this mimicry attack involves other security mechanisms and is the subject of ongoing work to be reported in a future paper.)

Herein lies the reason for False Positives produced by this model. The other models we explore below mitigate these mistakes by modeling the user's email frequency distribution.

As we can see from the ROC curve in Figure 5, the false positive rate is invariant with respect to the size of the recipient list. This is expected, as this rate is defined as the number of false positives over the number of normal emails, and both of these quantities do not vary with respect to how viral emails are sent under our simulation setting. It is interesting to note that the true positive detection increases dramatically as the size of the recipient lists in a viral email grows from 1 to 2 to 3 and then approaches 100% gradually as the list size further increases (Figure 5). This result is intuitive; we should not expect that there would be many user clique violations if a virus sends an email to only one recipient at a time. The fact that this number is not 0, as one might have thought, deserves some mention. This could happen because certain email addresses appear in an address book before any email is sent to them.

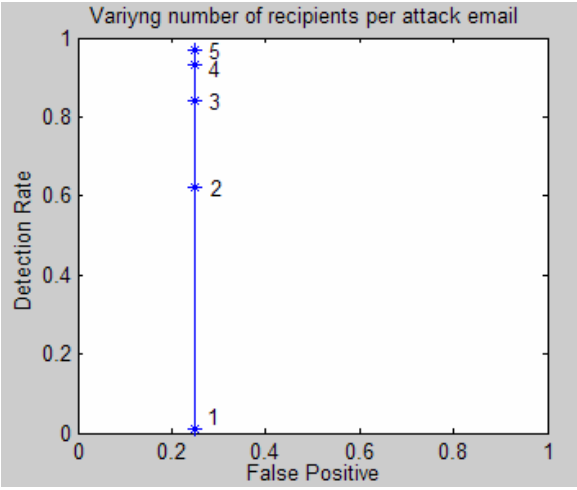


Fig. 5. Test of simulated viruses. Parameter: Varying number of recipients per attack email

While a virus may try to thwart our detection effort by sending itself to one address at a time, it will inevitably have to send many separate emails to achieve the same

propagation speed. In doing so, it is likely a different level of threshold would be triggered by another model that is tuned to the user's outbound email frequency. Thus, we combine the user clique detection model with other methods of detection, such as Hellinger Distance described in the next section, to mitigate this error.

Alternatively, as demonstrated below - the *Backward/Forward Scanning algorithm* - we may delay email transmission to gather evidence of clique violations among a sequential set of similar or equivalent emails indicative of a propagation. The TP and FP detection rates dramatically improve under this strategy as well.

5. NON-STATIONARY USER PROFILES

Most email accounts follow certain trends, which can be modeled by an underlying distribution. As a practical example, many people will typically email a few addresses very frequently, while emailing many others infrequently. Day to day interactions with a limited number of peers usually results in some predefined groups of emails being sent. Other contacts communicated to on a less than daily basis have a more infrequent email exchange behavior. These patterns can be learned through an analysis of a user's email archive over a set of sequential emails. For some users, 500 emails may occur over months, for others over days. The duration of these email transmissions is not material for the profile we now consider.

Almost every user of an email system develops a unique pattern of email emission to a specific list of recipients, each having their own frequency of occurrence (with respect to the number of emails). Modeling every user's idiosyncrasies enables the 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.

It is important to note that a user's email pattern is not static. The frequency distribution computed by EMT accommodates the user's change in frequency that may occur during the profile period, whether the user goes on vacation, is out sick, or is in a flurry of activity to make a deadline for submission. These changes are measured and modeled as we describe next.

5.1. Profile of a user

We analyze the user account's activity in terms of recipient frequency. Figure 6 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 and the corresponding height of the bar measures the frequency of emails sent to this recipient as a percentage. (The

display is an actual distribution from a volunteer email account. All others have been found to follow the same type of distribution.)

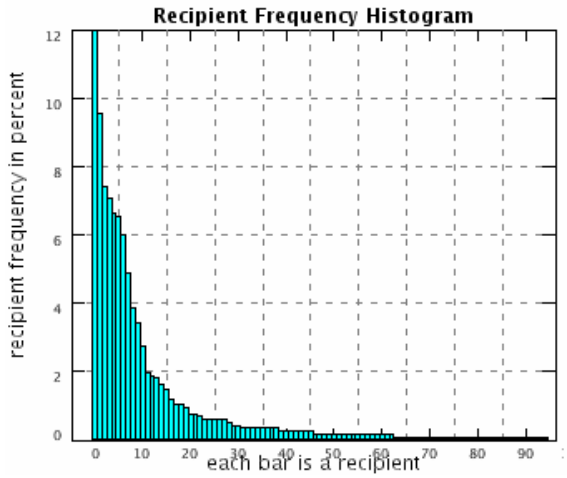


Fig. 6. Recipient Frequency

This bar chart is sorted in decreasing order, and usually appears as a nice convex curve with a strong skew; 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 distribution, or a DGX distribution (Discrete Gaussian Exponential distribution), which is a generalized version of the Zipf distribution. This family of distributions characterize some specific human behavioral patterns, such as word frequencies in written texts, or URL frequencies in Internet browsing [4]. 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 [4].

We also analyze the number of distinct recipients and attachments. However, we use "records" instead of emails. For example, if a user sends one email, to B, and CC's that email to C, D and E, these are four records in the database recording these communication events. Because a virus may send to several victims via one email (by CC'ing to everyone), or several emails (one by one), by using "records" we consider both cases within the model.

Figure 7 contains several curves that visualize the variability of the user's emission of emails. The statistics calculated are the *number of distinct recipients* and the *number of*

messages with attachments. The first type of curve uses a rolling window of 50 (or 20) records to calculate the number of distinct recipients. These values are ordered by time. For example, in Figure 6, this user has 750 records, and all of them are sorted by time. At the location 200 in the chart, the value of the curve, with the rolling window size of 50, is 10 (see the highest plot). This means that in the past 50 records there are 10 different recipients of the user's outbound email. What this analysis means is that the higher this plot approaches 50, the wider the range of recipients the selected user sends emails to over time. (The user is thus conducting many conversations with many people.) On the other hand, if the metric is low, it means that the user predominantly sends messages to a small group of people.

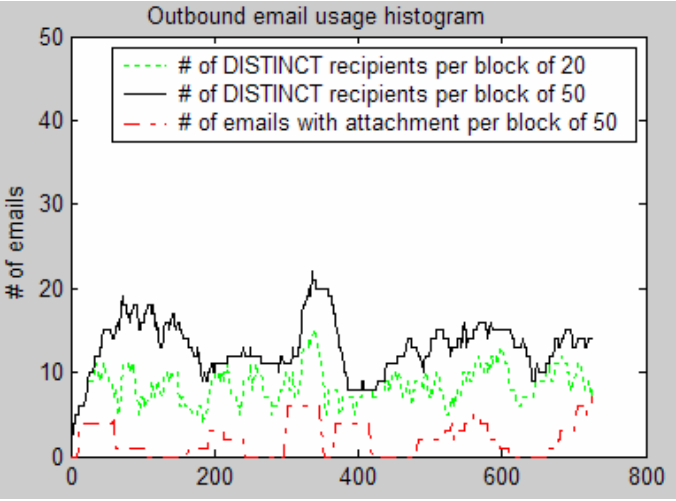


Fig. 7. Analysis of recipient and attachment

We also plot a curve (the middle dashed line) using 20 as the window size instead of 50. This metric has 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 any detected anomalous frequency change.

Another type of curve is the number of messages with attachment(s), per block of 50 records (the lowest dashed line). It shows the average ratio of emails with attachments versus emails without attachments, and any sudden spike of emails sent with attachments will be detected on the plot as a significant spike. The profile displays a fingerprint of a specific 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 curves will increase rapidly to 50, 20 and 50, respectively.

5.2. Chi Square Test of Recipient Frequency

We test the hypothesis that the recipient frequencies are identical over two different time frames by a Chi Square test. 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.

We compare two time periods of activity for the same user. The idea is to treat the first period as the true distribution corresponding to the user under normal behavior, while the second time period is used to evaluate whether or not the user's frequencies have changed, providing evidence that perhaps a malicious activity is taking place. Generally, we operate under the usual 1/5 - 4/5 ratio between testing and training sets. For example, we use 1000 records as a training-testing set, 200 recent records are selected as the testing range, while the previous 800 are the training range. Note that the "testing range" represents a user's new incoming or outgoing emails, and the "training range" represents the previous normal behavior used to generate the profile.

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 drawn from that same distribution [14]. The Chi Square formula is

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

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 during the training period. There are $k-1$ degrees of freedom.

The p-value represents the probability that the frequencies in both time frames come from the same distribution. In order to get an idea of the variability of the frequencies under real conditions, we used a sample of 37,556 records from 8 users. We ran two batches of calculations. First, we used a training period size of 400 records and a testing period size of 100 records; for each user, we started at the first record, calculated the p-value, then shifted 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 records, and a testing period size of 200 records. We thus collected a total of 7,947 p-values, and the histogram is shown in Figure 8.

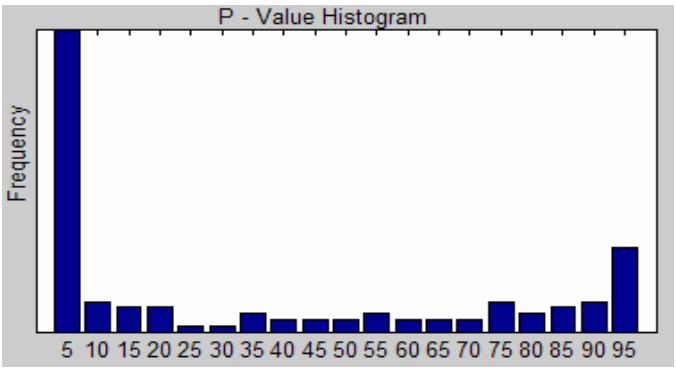


Fig. 8. P-value plot

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. Intuitively, 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). Email users tend to modify their recipient frequencies quite often (at least the 8 volunteers). On the other hand, 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 test itself could not be used to reliably detect an abnormal email behavior. Instead we utilize the Hellinger Distance metric, a related metric that evaluates changes in frequency over two frequency distributions.

5.3. Hellinger Distance

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

We use the *Hellinger Distance* for this purpose, as this metric is efficient in comparing two probability distributions of frequencies. It is defined as

$$HD(f_p[], f_t[]) = \sum_{i=0}^{n-1} (\sqrt{f_p[i]} - \sqrt{f_t[i]})^2$$

where $f_p[]$ is the array of normalized frequencies for the training set (profiling period), $f_t[]$ for the testing set, n is the total number of distinct recipients/senders observed during both periods. We define the *Hellinger testing window size* as the range of emails that are tested, while the *training range size* is a multiple of that, usually 4. The arrays of frequencies is defined as,

$$f_p[i] = N(i)_p / ws_p, \text{ and } f_t[i] = N(i)_t / ws_t,$$

where ws_p is the Hellinger training window size, ws_t is the Hellinger testing window size, $N(i)_p$ and $N(i)_t$ are the number of times that the current recipient (in the case for outbound traffic), or sender (for inbound traffic) of the emails appears in the range ws_p and ws_t, for the profiling period p and testing period t, respectively, of emails being evaluated. This is computed for both inbound and outbound email traffic. Figure 9 displays an example for a user from our group of volunteers who provided their email archive.

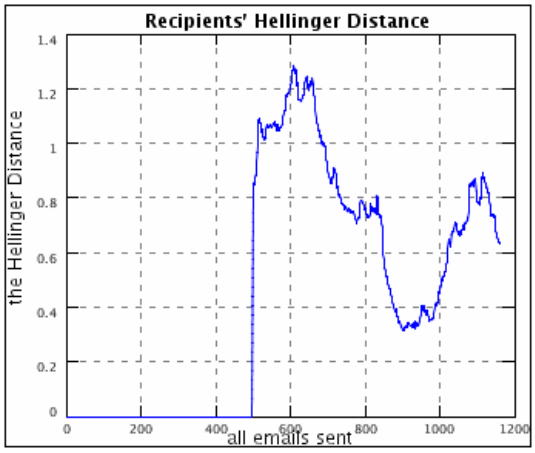


Fig. 9. The Hellinger distance of a typical User

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 records and the window size is 100, 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 (training) and 100 (testing) 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.

What this plot tells us is that when a burst in email activity occurs, the recipient frequencies have been changing significantly. This statistic provides evidence of either a highly variable user, or a possible viral propagation.

5.4. Evaluation techniques using simulated viruses and threshold settings

As real email data with real embedded viral emails are very difficult to obtain [29] (and dangerous and possibly illegal to generate), we injected "dummy" viruses into a real email log file as described above. A set of parameters introduces randomness in the process, in order to mimic real conditions and explore boundary conditions: the time at which the virus starts, the number of corrupted emails sent by the virus and its propagation rate.

For testing purposes, all the recipients of such "dummy" corrupted records are picked randomly from the address list of a selected user. In reality, where addresses are obtained and how they are combined can be a crucial issue for a virus to successfully propagate itself without being detected. The simulated recipient list of the virus can be set to be all distinct addresses, as most viruses seem to do. But not all viruses would send an email only once to each target recipient account. In our simulation, each "dummy" record contains one attachment, but no information about the attachment is provided or used. (Recall, our focus here is to demonstrate the value of behavior models, as an adjunct to content-based analyses.) For our purposes, we do not need to know the content of the message, its size, and the size and content of the attachments. So, these techniques may be general enough that they encompass polymorphic viruses as well (where content analysis or scanners may fail).

The experiments use a combination of three plots, Hellinger Distance, "number of distinct recipients", and the "number of attachments". Figure 10 displays plots detailing the profile of one user in our archive. Our intuition is that when a virus is executed, its propagation will cause each plot to grow, i.e. it will not "simulate" the user's real frequency distribution. We use a threshold logic to detect "abnormal" growth of these plots.

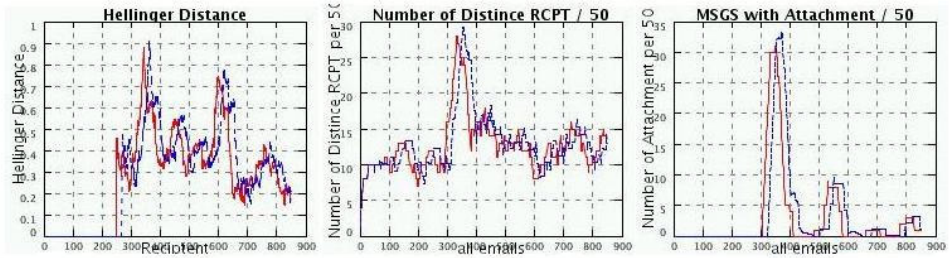


Fig. 10. Three evaluating models.
Solid lines: Calculated value, Dash lines: Threshold

We use two types of thresholds to determine when a "burst" occurs, a threshold proportional to the standard deviation of the plots, and a threshold based upon the changing "trend" revealed by the Hellinger Distance, both conditioned on a window size of prior plotted values. An email is deemed viral if any model deems it to be viral according to the threshold settings.

We believe it may be necessary to calibrate the threshold settings on a per user basis. For this study, we did not implement specific "user calibration"; rather we aim to establish a baseline of performance over all users to reveal whether user specific thresholds might be needed.

The first threshold is the value of the plot at some point adjusting it by a factor proportional to the standard deviation of the average value of the plot calculated from the previous *n* values of the plot. Thus, the threshold is dynamic, essentially proportional to one standard deviation from the mean for the recent user's behavior. The threshold value is defined as

$$T[i] = (V[j] + \alpha * st(j, j-n))$$

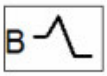
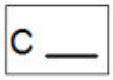
where $T[i]$ is the threshold value at location *i*, $V[j]$ is the value of the plots at location *j*, α is a constant that is set to 0.1, the function *st(a, b)* returns the standard deviation in the range $[a,b]$, *n* is the window size, and *j* = *i* – *shift*. We need a *shift* value to calculate the threshold using the prior range of data. Without the *shift* value, the threshold would be always higher than the original value since $T[i]$ is always greater than $V[i]$.

The second threshold was developed by observing the trends of the three plotted statistics. When both the "number of distinct recipients" and the "number of attachments" plots grow (the values increase), and when the *slope* of the "Hellinger Distance" model grows, this range of emails is marked as suspicious. This means that the

"slope" acts as a threshold for the Hellinger Distance model, but is used only when both plots (number of recipients and number of emails with attachments) exceed their threshold. Hellinger Distance thus severs as a "confirmation" of the other two models.

In Figure 10, the test "dummy" emails simulating viral emails are injected at location 300. In the two rightmost curves we see a "burst". However, in the Hellinger Distance plot on the left, it's not confirmed as a "burst" since in this period of time the user's changing behavior is not abnormal. Recall the Hellinger function. The Hellinger distance expresses the change in the user's behavior. There are four trends that the Hellinger metric may reveal as shown in Table 2. The columns are the user behavior indicating the number of recipients that the user usually sends to.

Table 2. User behavior vs. Hellinger Distance

	Lots of different recipients	Few different recipients
High Hellinger Distance	A 	B 
Low Helllinger Distance	C 	D 

The curves graphically represent different user behaviors during periods of time when viral emails may appear. In region A, the user is changing his/her behavior rapidly, and a viral propagation with many recipients would be more difficult to detect. These would be detected towards the end of the period when the slope of the Hellinger plot changes to zero. In region B, when the user's behavior is stable, viral propagations are more noticeable. In region D, a stable user behavior provides the means of detecting a viral propagation more easily. However, in region C, nothing can be found easily. A stable user who sends to lots of different recipients is the best victim of a virus. This situation means the user's normal behavior is akin to a virus propagation! He or she always sends emails to all the people he or she knows.

5.4.1 Results and Discussion

The dataset used for this independent test is an archive of 15 users, totaling 20,301 emails. The parameters that were randomly generated at each simulation were the time of the injected viral emails and the list of recipients (taken from the address list of each selected user). The parameters that were controlled were the propagation rate, the number of corrupted records sent, and the window size (of the Hellinger distance metric). In total, about 500,000 simulations were performed.

As expected, a slower propagation rate (longer inter-departure time) makes detection harder. Each email record corresponding to a virus email inserted into the archive becomes less "noticeable" among the entire email flow. As can be seen in Figure 11, the performance gets worse when the inter-departure time increases; that is to say slow and stealthy propagations are hard to detect.

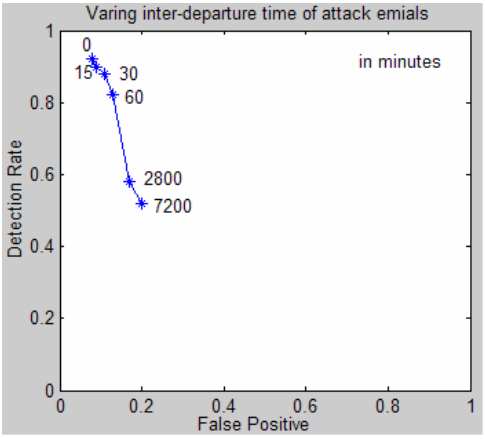


Fig. 11. Varying inter-departure time

The Hellinger window size is the most important parameter. We plot the TP rate as a function of window size in Figure 12 to evaluate the sensitivity of the model to this parameter. In this test, the inter-departure time is 30 minutes. The performance is best when the window size is the same as the number of dummy records inserted into the archive. The reason is that, for example, when the window size is 50 and there are 20 injected viral records (the number of injected viral records is less than window size), these records do not occupy a very significant portion of the 50 records. The model may not determine that they are suspicious. On the other hand, if there are 100 injected records and the first 50 are detected, these 50 dummy records will be treated as normal records in the next round of Hellinger training. As a consequence, the system will likely model the first 50 as normal and not be able to detect any abnormality.

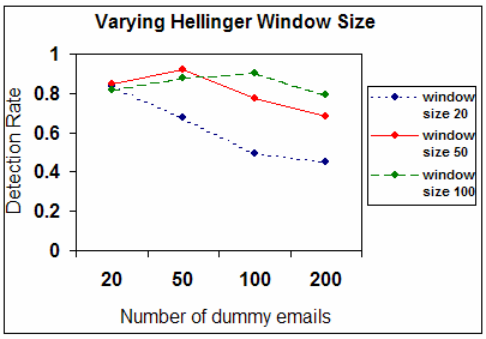


Fig. 12. Increasing Hellinger window size produces lower detection rates.

In summary, we achieved very reasonable results with the Hellinger distance model. However, there are still three problems. First, we assumed that we had enough normal records before and after the inserted viral emails to develop sufficient statistics modeling the user and detecting the propagation. We also assume in this simulation that we can analyze all the records (both dummy and normal) at the same time, which is not practical. We cannot block a user's email for a long time, for instance, a few hours. However, we may store a record of the emails and detect the propagation after the fact, but perhaps still in sufficient time to forewarn the recipients that they likely have received a viral email in their inbox from the recently detected infected victim.

Second, it's difficult to optimize the Hellinger window size, as it depends on the viral strategy used. In practice, we can overcome this by blocking all outgoing emails once we detect a virus. The question is then how can we detect the first instance of the virus propagation?

Third, the false positive rate is about 15%, which cannot be reduced in this model easily. Thus, to achieve a better detector, this method has to be used in combination with other models. The first two issues will be addressed in the next section.

6. COMBINING USER CLIQUE AND HELLINGER DISTANCE

The Hellinger distance model is the result of analyzing the aggregate behavior of a sequence of emails. As such, it would not react immediately when a viral email appears. Similarly, it would keep setting off alarms for a short while after a batch of viral emails has already been sent out. On the other hand, the user cliques model could detect a suspicious viral email upon its first appearance. It is worth mentioning that every time an email with a new address appears in the user's inbox, the user clique model will treat it as a violation. In short, Hellinger analyzes the trend of users' behavior by analyzing a buffer of email records of their recent behavior, while the user clique method is oriented towards detection of individual viral emails at that moment in time when they are sent or received. Ideally, combining these models may achieve better overall detection performance.

6.1 Backward/Forward Scanning algorithm

The intuitive reasoning here is quite simple. When sufficient evidence for a viral propagation has been detected, i.e. an email has an alert generated by both models (clique violation and a substantial change in the user's email emission) it is highly likely that prior and subsequent emails will be part of the virus propagation. We seek to detect these

other emails by searching a set of buffered emails (or their record of emission) inspecting the model outputs for each. We search prior emails for evidence of being part of the onset of a propagation. This evidence is simply whether one of the EMT models has deemed it a violation. We also search forward in time and test emails until we find an email that violated no model. Intuitively, therefore, the propagation has terminated, or the user has sent legitimate emails during the propagation. We apply this technique for both inbound traffic (the optimal case to prevent infection) and the outbound case when an infection has succeeded but we wish to limit the viral spread as quickly as possible.

The most straightforward method to combine the user clique and Hellinger Distance models is to "intersect" their alert outputs. Depending upon the threshold settings, a close examination shows that they have different distributions of false positives. For example, the user clique model may generate false positives on email number 1, 3 and 5, while Hellinger may generate false positives on email number 2, 4 and 6. If we take the intersection, we can eliminate most false positives. However, a lower false positive rate may be achieved at the expense of a lower TP detection rate.

We propose an alternative strategy we call the Backward/Forward Scanning algorithm. Emails are assumed to be buffered before they are actually sent out or, as we mentioned, a record of all sent emails are kept for analysis, including instances of the virus that have escaped without early detection. These records however inform as to where those viral emails were sent so new victims may be warned. This is a key feature introduced in the MET system.

Such rate limiting or buffering of email could be hidden and unbeknownst to the user. Email may be viewed as a store and forward technology (at least one hop through the server). However, an egress "*store for a while, then forward*" strategy for email delivery has a practical advantage. As far as the user is concerned, the email is sent from client to server and is delivered by the underlying communication system at some arbitrary future time. Thus, the strategy of buffering and holding emails for some period of time allows sufficient statistics to be computed by the models and also benefits mitigation strategies to quarantine viral emails before their delivery, limiting exposure to the enterprise or enclave.

Alternatively, a record of the recently delivered emails may also benefit early detection and mitigation strategies. When the system sees an alert triggered by both the Hellinger Distance model and the user clique model, it will examine all adjacent emails more closely, those preceding it and those newly sent by the client. Namely, it will trace (scan) all buffered emails forward and backward (or their record of delivery), starting

from the common trigger. The trace attempts to find all sequential emails that are deemed suspicious by the user clique model and will end once a harmless email, as viewed by user cliques, is encountered. The system then marks all those emails found along the trace as suspicious. The algorithm schema below lists the main steps in this algorithm. The input test emails denoted T, are ordered by time. The output is a bit vector R where bit i is set indicating T[i] is deemed viral.

The backward/forward scanning algorithm

```
1. Let T be the ordered list of test emails, which are sorted by time.
2. The length of T = n.
3. Let C be the alerts generated by the Clique model from T and H be the alerts generated
   by the Hellinger Model from T.
4. For each email T[i], i=1, ..., n,  H[i] = true, if the Hellinger model scored the email
   as anomalous. Likewise, C[i], i=1, ..., n records whether an email had been issued an alert
   by the Clique violation model.
5. Let R be the result alerts.
6. Let i and j be the reference of the first and last emails on testing day, respectively.
7. FOR k = i to j
        IF C[k] = true and H[k] = true
                R[k] = true
                last_position = k

                k = last_position -1            //backward
                WHILE k >= i and C[k] = true and H[k] = false
                        R[k] = true
                        k = k-1

                k = last_position + 1           //forward
                WHILE k <= j and C[k] = true and H[k] = false
                        R[k] = true
                        k = k+1
        ELSE
                R[k] = false
                k = k+1
RETURN R
```

Figure 13 is a graphical view of this Backward/Forward Scanning algorithm. Each email in the sequence is denoted by “x” or “o”, depending on whether or not there is an alert associated with it. In this example, we have 18 emails, labeled from #1 to #18. These emails are buffered (stored) and analyzed by both models. The alerts generated by the user clique model are in the first row. The suspicious emails with alerts are #4 to #11, #16 and #17. The alerts generated by the Hellinger Distance model are in the second row, and the suspicious emails are #7, #8, #9, #13 and #14.

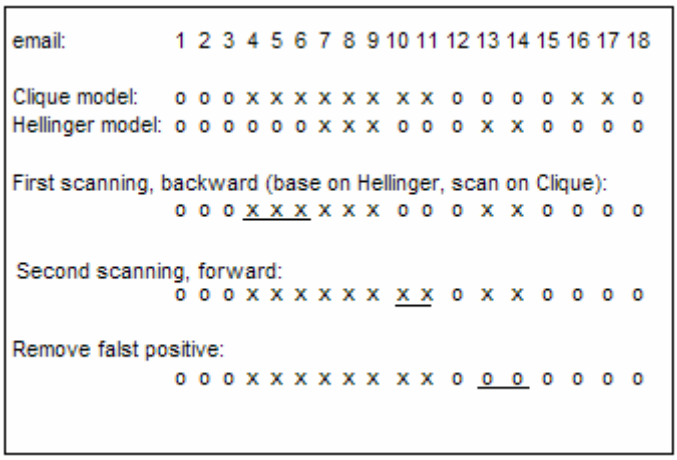


Fig. 13. A graphical view of the buffer scanning method.

The algorithm proceeds as follows. In the first step, we find the first alert triggered by both models. In this case email #7 is detected. We then inspect the model outputs for each of the adjacent emails prior to #7 and find #4, #5 and #6 have model outputs that triggered alerts by the user clique model, but not by the Hellinger Distance model. We thus generate alerts for each of these as the first set of outputs by this buffer scanning method.

In the second step, we scan forward, those emails occurring after #7. Here we find emails #8 and #9 have alerts triggered by both models, but #10 does not. We check the model outputs for those immediately following emails occurring after #10 for which one model has generated an alert. Stepping forward, we capture #10 and #11, but terminate at email #12, which has generated no alerts for either model. Finally, we clear all the alerts that are not triggered by both models after # 11 deeming these as False Positives. We next report results using this strategy.

6.2 Tests of simulated virus propagations

The dataset for this independent test, both inbound and outbound emails spanning a time frame between 2001-01-01 and 2002-12-31, includes 53163 emails from 15 users, and 8% of these emails contain attachments. The distributions for these users are depicted in Table 3. EMT computes one user's model in less than 5 seconds, and each model output requires at most 1M byte of space.

Table 3. testing dataset

	Outbound			Inbound		
Users	# of Emails	# of Records	# of records with attachment	# of Emails	# of Records	# of records with attachment
1	1526	3849	270	3739	3929	325
2	6723	9509	309	8334	8461	925
3	678	678	21	523	579	50
4	400	1059	7	543	544	10
5	128	547	0	209	276	12
6	188	513	18	506	518	26
7	456	726	33	693	721	29
8	207	708	6	352	354	12
9	261	890	122	345	398	51
10	211	588	4	619	671	12
11	188	595	14	388	420	29
12	5577	11447	1347	7203	7647	937
13	507	1484	191	680	711	161
14	2519	3917	107	2365	2431	251
15	3396	5198	281	3699	3877	454

We treat the first 80% of the email data as each user's normal behavior (training data) and inject a batch of "dummy records" into the last 20% (testing data). We test 100 times for each user, generate the dummy emails at a random location (time), and take the average of the results over all runs for the three different users. Results for each user are displayed in Tables 4 - 11. Naturally, only emails with attachments are tested by the EMT models since only these can contain viruses. Thus, for clarity, the TP rate is the percentage of dummy emails deemed correctly to be viral, and the FP rate is the percentage of "normal" user emails with attachments that are mislabeled by EMT as viral.

During the test phase, we train and test the email data on a daily basis. For example, on the first day of the eleventh month, we put all the data (on that day) into the buffer. Then we use the training data to test whether the emails are suspicious depicting abnormal behaviors. We then move forward on a day by day basis for testing. After an individual test day, we assume the user will review and confirm the normal and viral data. Then, we will update the database, which is training data (i.e. add the normal emails for next daily test and drop viral emails).

The parameters that are controlled are the *propagation rate* and the *number of recipients in a single dummy email*. The first parameter is one of the most important issues in the Hellinger simulation (section 5.4). The second parameter is more pertinent to user cliques. Having more recipients in a single email makes it easier for the user clique model to detect a violation. Both inbound and outbound emails are tested. However, we divide the results, because a user's behavior of inbound and outbound emails may be different.

Another important issue is the Hellinger window size (Hellinger Distance, see section 5.3, 5.4). Since it is impossible *a priori* to choose a single and perfect Hellinger window size in the general case for all users, we change it by evaluating the size of data (records) each day for each user. The window size is meaningless if it is too small or too large. If it's too small, each email has too much of an influence on the statistics and each email may look like a virus. If it's too large, the training data would not be enough to establish sufficient statistics and a small number of virus emails could easily go undetected. In our simulation, we set the window size to the average number of daily emails sent by each user, bounded below by 20 and above by 100. Optimally calibrating this parameter for each user is the subject of ongoing research.

We first test and measure the outbound email from a user account to detect the onset of a viral propagation from an early victim. Varying the number of recipients in a single virus email yields a very interesting result. The upper-left plot of Figure 14 displays the average of the results. The TP rate increases with the size of the recipient list in a simulated viral email rapidly approaching 100%. This means a virus email is easy to detect if it propagates itself to many email addresses (for example, 9) and sends them in a single email or at the same time. We found that with just three recipients in a single email, the average TP rate is about 90%. The reason FP hovers around 8% and is almost invariant with respect to virus strategy is rooted in the definition of FP, the number of false positives divided by the total number of non-viral emails with attachments. Only the numerator depends on the properties of the emails being tested by the models. In addition, the alarm is triggered due to clique violations and Hellinger violation. The same false alerts are always triggered, regardless of the viral propagation strategy. We find that each user model exhibits a different TP rate and FP rate.

The first test reveals encouraging results. However, this is because we set a high propagation rate in our simulated "dummy" emails. The inter-departure time used is uniformly distributed between 0 and 10 minutes in this test. The next test varies this propagation rate.

In this independent test, the number of recipients in a single email is set to 4. Similar to the propagation rate test in section 5.4, the detection rate gets worse when the inter-departure time increases (the upper-right plot of Figure 14 and Tables 6 and 7) (i.e. the virus is stealthily propagating at a very slow rate). If this happens in the real world, once we detect the first virus (with long inter-departure time), we would likely have enough time to mitigate its effects, since it propagates slowly. Thus, the issue here again is how best to detect the first virus in a new propagation.

Fortunately, on average, the models in both tests do detect the first or second dummy email. In the last row of Tables 4 and 6, we see that Backward/Forward Scanning algorithm can accurately detect the first viral emails, but alas these have already escaped to their new targets.

We next consider the results achieved by EMT for inbound email; the optimal case to prevent viral propagations from entering an enclave or attacking a victim in the first place. The parameters for the inbound test of EMT are the same as the outbound test. The results are displayed in the lower plots of Figure 14, and Tables 8 through 11.

Here we aim to detect an inbound email with an attachment from a sender that is unusual. Notice, in the lower-left plot of Figure 14 inbound emails with a single recipient have a very low TP rate. For the user data on hand, it is NOT unusual that inbound emails have only one recipient. When the number of recipients increases, and clique violations appear, the detection rate naturally grows, but with a fairly stable FP rate. Notice too in the lower-right plot of Figure 14 that the rate of receipt of viral emails affects detection performance. Fast arrival times are easy to detect inbound viral propagations. Slow rates decrease performance markedly. We may mitigate these FP rates in the same fashion as outbound traffic. By inspecting sequences of inbound emails destined to a user over a period of time, we increase the likelihood of detecting the inbound viral traffic. Moreover, because each user may a distinct behavior from other users and a user's behavior may change rapidly over time, we can also measure the dynamic activity of users by using statistics capturing cyclic interaction patterns [7] to achieve better performance.

Table 4. Outbound Detection Rate (TP). Parameter: varying number of recipients in a single email. Miss: the average number of virus mails missed before a propagation is detected.

	1	2	3	4	5	6	7	8	9
User 1	0.0	0.935	0.984	1.0	1.0	0.972	0.996	0.996	0.992
User 2	0.0	0.92	1.0	0.996	1.0	1.0	0.996	1.0	1.0
User 3	0.0	0.616	0.924	0.996	0.98	1.0	1.0	0.992	0.996
User 4	0.0	0.424	0.672	0.856	0.972	0.936	0.944	0.996	0.98
User 5	0.0	0.328	0.496	0.748	0.916	0.968	0.956	0.948	0.992
User 6	0.0	0.58	0.812	0.952	0.968	0.9	0.952	0.972	0.952
User 7	0.0	0.924	0.988	0.996	0.992	1.0	0.996	1.0	1.0
User 8	0.0	0.82	0.984	0.996	0.988	1.0	0.976	0.98	0.992
User 9	0.0	0.528	0.756	0.892	0.968	0.964	0.988	1.0	1.0
User 10	0.0	0.56	0.904	0.94	0.992	1.0	1.0	1.0	0.996
User11	0.0	0.692	0.976	0.94	0.992	0.988	0.996	0.996	0.98
User 12	0.0	0.996	0.976	0.972	0.984	1.0	0.992	0.992	1.0
User 13	0.0	0.904	0.988	0.996	1.0	0.992	1.0	1.0	1.0
User 14	0.0	0.964	0.98	0.992	0.988	0.992	0.984	0.988	1.0
User 15	0.0	0.956	0.988	0.976	0.996	1.0	1.0	0.996	1.0
Avg.	0.0	0.743	0.888	0.95	0.982	0.981	0.985	0.99	0.992
Miss	NA	1.48	1.02	0.38	0.14	0.147	0.147	0.067	0.08

Table 5. Outbound False Positive (FP). Parameter: varying number of recipients in a single email.

	1	2	3	4	5	6	7	8	9
User 1	0.122	0.122	0.1	0.115	0.113	0.096	0.11	0.11	0.101
User 2	0.128	0.093	0.102	0.108	0.097	0.097	0.136	0.136	0.116
User 3	0.15	0.189	0.195	0.176	0.143	0.137	0.146	0.187	0.161
User 4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 7	0.089	0.101	0.095	0.092	0.095	0.098	0.094	0.099	0.105
User 8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 9	0.158	0.142	0.152	0.122	0.168	0.146	0.122	0.194	0.186
User 10	0.113	0.156	0.198	0.167	0.174	0.159	0.172	0.187	0.178
User11	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 12	0.077	0.093	0.096	0.091	0.103	0.095	0.098	0.103	0.092
User 13	0.216	0.197	0.182	0.189	0.178	0.196	0.162	0.145	0.158
User 14	0.071	0.097	0.06	0.083	0.081	0.078	0.065	0.069	0.087
User 15	0.016	0.025	0.031	0.031	0.029	0.033	0.025	0.032	0.026
Avg.	0.0760	0.0810	0.0807	0.0783	0.0787	0.0757	0.0753	0.0841	0.0807

Table 6. Outbound Detection Rate (TP). Parameter: varying inter-departure time (in minutes). Miss: the average number of virus emails missed before a propagation is detected.

	0	10	15	30	60	720	2880	7200
User 1	1.0	0.968	0.996	0.988	0.992	0.988	0.812	0.86
User 2	1.0	0.992	0.992	0.992	0.976	0.912	0.804	0.548
User 3	1.0	0.996	0.98	0.992	0.952	0.984	0.884	0.928
User 4	1.0	0.932	0.932	0.94	0.912	0.872	0.568	0.492
User 5	1.0	0.816	0.712	0.724	0.812	0.764	0.448	0.336
User 6	1.0	0.908	0.916	0.98	0.9	0.88	0.68	0.58
User 7	1.0	0.996	0.992	0.996	0.996	0.976	0.972	0.996
User 8	1.0	1.0	1.0	1.0	0.988	0.996	0.8	0.896
User 9	1.0	0.928	0.908	0.88	0.924	0.92	0.696	0.7
User 10	1.0	0.944	0.96	0.932	0.972	0.932	0.884	0.864
User11	1.0	0.976	0.976	0.976	0.948	0.996	0.888	0.672
User 12	1.0	0.988	0.964	0.992	0.944	0.856	0.716	0.68
User 13	1.0	1.0	1.0	1.0	0.996	0.972	0.832	0.752
User 14	1.0	0.992	1.0	0.996	0.992	0.912	0.784	0.644
User 15	1.0	0.996	0.988	0.964	0.996	0.832	0.736	0.88
Avg.	1.0	0.962	0.954	0.957	0.953	0.919	0.767	0.722
Miss	NA	0.247	0.567	0.44	0.593	0.473	1.013	1.3

Table 7. Outbound False Positive (FP). Parameter: varying inter-departure time (in minutes).

	0	10	15	30	60	720	2880	7200
User 1	0.103	0.103	0.107	0.103	0.123	0.116	0.112	0.118
User 2	0.114	0.108	0.119	0.101	0.099	0.099	0.077	0.055
User 3	0.191	0.171	0.191	0.182	0.182	0.17	0.157	0.139
User 4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 7	0.09	0.093	0.093	0.093	0.099	0.1	0.096	0.089
User 8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 9	0.154	0.124	0.152	0.12	0.148	0.192	0.172	0.122
User 10	0.215	0.109	0.163	0.089	0.161	0.145	0.198	0.176

User11	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 12	0.109	0.094	0.096	0.093	0.094	0.082	0.082	0.083
User 13	0.194	0.156	0.163	0.159	0.179	0.191	0.16	0.156
User 14	0.086	0.077	0.071	0.089	0.084	0.053	0.064	0.056
User 15	0.025	0.027	0.027	0.026	0.025	0.022	0.024	0.019
Avg.	0.0854	0.0708	0.0788	0.0703	0.0796	0.0780	0.0761	0.0675

Table 8. Inbound Detection Rate (TP). Parameter: varying number of recipients in a single email. Miss: the average number of virus emails missed before a propagation is detected.

	1	2	3	4	5	6	7	8	9
User 1	0.0	1.0	1.0	0.996	1.0	1.0	1.0	1.0	1.0
User 2	0.0	0.984	0.992	1.0	1.0	1.0	1.0	1.0	1.0
User 3	0.0	0.946	0.971	0.976	0.992	0.976	0.988	0.984	0.992
User 4	0.0	0.996	1.0	0.996	1.0	0.996	1.0	1.0	1.0
User 5	0.0	0.9	0.748	0.768	0.484	0.996	0.988	0.988	0.988
User 6	0.0	0.992	0.996	1.0	1.0	1.0	1.0	1.0	0.996
User 7	0.0	0.8	0.996	0.932	0.932	0.988	0.996	0.964	0.984
User 8	0.0	0.992	0.924	0.992	1.0	1.0	1.0	0.952	0.98
User 9	0.0	0.992	0.996	1.0	0.984	0.992	1.0	1.0	0.996
User 10	0.0	0.88	1.0	0.992	1.0	0.996	0.992	1.0	0.996
User11	0.0	0.976	0.992	1.0	0.944	0.96	0.996	0.996	0.988
User 12	0.0	0.972	0.996	1.0	1.0	1.0	1.0	1.0	1.0
User 13	0.0	0.972	1.0	0.996	0.992	0.992	0.996	0.996	0.972
User 14	0.0	0.984	0.976	0.996	1.0	0.992	1.0	1.0	1.0
User 15	0.0	0.988	0.984	1.0	1.0	0.996	1.0	0.992	1.0
Avg.	0.0	0.959	0.971	0.976	0.955	0.992	0.997	0.991	0.993
Miss	NA	0.287	0.513	0.327	0.473	0.1	0.02	0.067	0.073

Table 9. Inbound False Positive (FP). Parameter: varying number of recipients in a single email.

	1	2	3	4	5	6	7	8	9
User 1	0.014	0.021	0.014	0.015	0.017	0.012	0.011	0.018	0.021
User 2	0.102	0.129	0.124	0.098	0.129	0.114	0.119	0.116	0.098
User 3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 4	0.008	0.007	0.005	0.011	0.005	0.006	0.003	0.006	0.003
User 5	0.013	0.02	0.009	0.014	0.014	0.013	0.02	0.022	0.014
User 6	0.09	0.098	0.096	0.097	0.098	0.094	0.092	0.093	0.088
User 7	0.032	0.027	0.024	0.025	0.024	0.026	0.019	0.029	0.022
User 8	0.026	0.022	0.016	0.019	0.019	0.022	0.019	0.014	0.023
User 9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 10	0.006	0.007	0.007	0.007	0.007	0.005	0.007	0.004	0.008
User11	0.021	0.02	0.02	0.02	0.035	0.021	0.023	0.02	0.021
User 12	0.068	0.061	0.077	0.08	0.064	0.065	0.069	0.067	0.081
User 13	0.009	0.006	0.004	0.003	0.004	0.001	0.004	0.011	0.006
User 14	0.057	0.052	0.037	0.043	0.044	0.037	0.045	0.061	0.044
User 15	0.021	0.024	0.024	0.028	0.023	0.023	0.023	0.019	0.026
Avg.	0.031	0.033	0.03	0.03	0.032	0.029	0.03	0.032	0.03

Table 10. Inbound Detection Rate (TP). Parameter: varying inter-departure time (in minutes). Miss: the average number of virus emails missed before a propagation is detected.

	0	10	15	30	60	720	2880	7200
User 1	1.0	0.996	1.0	1.0	1.0	0.992	0.904	0.744
User 2	1.0	0.976	1.0	0.996	0.964	0.94	0.804	0.724
User 3	1.0	0.988	0.984	0.984	0.948	0.976	0.948	0.932
User 4	1.0	1.0	1.0	0.996	1.0	0.996	0.64	0.68
User 5	1.0	0.988	0.976	0.992	0.956	0.952	0.892	0.86
User 6	1.0	0.98	1.0	0.988	0.984	0.984	0.94	0.92
User 7	1.0	0.908	0.964	0.948	0.98	0.932	0.796	0.88
User 8	1.0	1.0	1.0	0.96	0.971	0.916	0.736	0.6
User 9	1.0	0.996	0.996	0.996	0.984	0.932	0.768	0.392
User 10	1.0	0.996	0.988	0.98	0.992	0.952	0.68	0.708
User11	1.0	0.996	0.996	0.992	1.0	0.972	0.932	0.932
User 12	1.0	0.996	1.0	0.996	0.992	0.908	0.864	0.724
User 13	1.0	0.988	0.988	0.992	0.952	0.956	0.836	0.8
User 14	1.0	0.98	0.984	0.987	1.0	0.96	0.748	0.7
User 15	1.0	0.996	0.992	0.984	0.98	0.904	0.792	0.88
Avg.	1.0	0.985	0.991	0.986	0.98	0.951	0.819	0.766
Miss	0.0	0.287	0.113	0.247	0.306	0.313	0.96	0.827

Table 11. Inbound False Positive (FP). Parameter: varying inter-departure time (in minutes).

	0	10	15	30	60	720	2880	7200
User 1	0.017	0.019	0.019	0.015	0.017	0.018	0.02	0.019
User 2	0.117	0.099	0.124	0.121	0.116	0.125	0.102	0.082
User 3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 4	0.002	0.008	0.005	0.011	0.001	0.01	0.007	0.005
User 5	0.022	0.018	0.02	0.022	0.014	0.025	0.45	0.031
User 6	0.1	0.092	0.1	0.093	0.098	0.088	0.09	0.089
User 7	0.018	0.025	0.018	0.022	0.021	0.025	0.023	0.015
User 8	0.015	0.02	0.022	0.019	0.023	0.016	0.019	0.02
User 9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 10	0.005	0.006	0.009	0.005	0.004	0.006	0.004	0.004
User11	0.022	0.021	0.017	0.024	0.019	0.031	0.02	0.017
User 12	0.065	0.055	0.067	0.072	0.064	0.068	0.062	0.047
User 13	0.002	0.009	0.006	0.005	0.007	0.008	0.004	0.012
User 14	0.043	0.04	0.044	0.062	0.061	0.064	0.063	0.033
User 15	0.025	0.025	0.027	0.023	0.023	0.022	0.021	0.02
Avg.	0.03	0.029	0.032	0.033	0.031	0.034	0.032	0.026

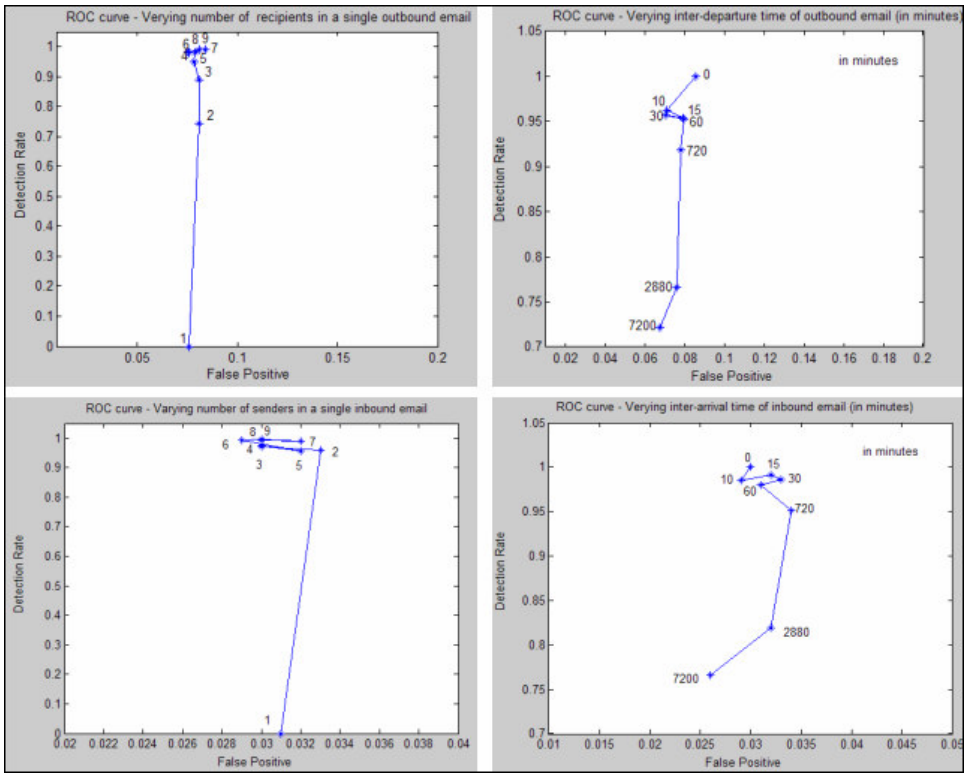


Fig. 14. Average results of each test

Upper-Left: Outbound test, varying number of recipients in a single email.

Upper-Right: Outbound test, varying inter-departure time (in minutes).

Lower-Left: Inbound test, varying number of recipients in a single email.

Lower-Right: Inbound test, varying inter-arrival time (in minutes).

7. IMPROVING THE DETECTION OF CHANGES IN FREQUENCY

The Hellinger distance model performs fairly well with an impressive TP rate. However, the FP rate may render the approach too frustrating for users who are accustomed to seeing no false alarms generated by their virus scanners. Combining the Hellinger model with the clique violation model improved the results, yet the false positives remain too high. Here we explore the addition of a third model, the cumulative distribution of emitted emails by a user in a sequence of emails. Here, we restrict the statistic to those emails with attachments that appear in the user's archive.

7.1 Cumulative attachment distribution

Suppose we have a period over T days, from day 1, 2,…, T, and let N_i be the number of emails with attachments on day i, and U_i be the cumulative number of emails with attachment until day i. Thus,

$$U_i = \sum_{j=1}^{i} N_j$$

The idea here is that a user will emit emails with attachments at a relatively low and constant (human-oriented) daily rate. If we take a long period of time (for example, three months), the slope may be a positive constant. The introduction of a viral propagation would manifest as a significant discontinuity in this plot. Because in the real email data in our database, most users don't send emails with attachments each day, we compute the slope of several days (testing days), and test the slope with a previous longer period (the longer term profiling training days).

Assume we have t testing days from day i to day $i+t$, and r training days from day j to day $j+r$. We detect this discontinuity of the user emission rate behavior over t days by comparing the slopes of the cumulative distribution over r days.

$$\alpha * (U_{i+t} - U_i)/t > (U_{j+r} - U_j)/r$$

where α is the tolerance parameter for the change of the slope. The threshold α is set by the following intuition. Assume a user's normal behavior of sending emails with attachments will not be more than on standard deviation above the mean of the prior d days, i.e. let $V_i = U_i +$ *standard deviation in previous d days* and V_i is the number of emails with attachment of the user's normal trend on day i. For example, if on average, a user sends one email with an attachment every day, we expect that the user will send one with perhaps a few more email(s) with attachments on the next day, denoted V_i. Then, to compute the boundary of normal behavior, we compute:

$$\alpha = ((V_i - U_{i-d})/d)/((U_i - U_{i-d})/d)$$

After computing this ratio for each users' email in our database, α ranged between 1.1 and 1.5. Using this formula, d can be set to any value of course. In the experiments reported here, d is set to 5 and α is set to 1.2, which we shall see are acceptable values.

If the calculated value of the data violates the inequality in our daily testing (day $i+t$), we say emails on this day are suspicious. Then we use this fact to confirm the alerts that may have been generated by the Hellinger and Clique models. If either of them has issued alerts on day $i+t$, we keep these alerts. If not, we cancel the alerts on day $i+t$.

In Figure 15, the blue (upper) lines are the cumulative number of emails plotted day by day, and the red (lower) lines are likewise the cumulative number of emails with attachments (a strict subset of all emails emitted by the user). The left plot is a user's normal email cumulative distribution. In the right plot, after we add some dummy

simulated viral emails; the red line displays an apparent discontinuity or a burst identified by the green circled area.

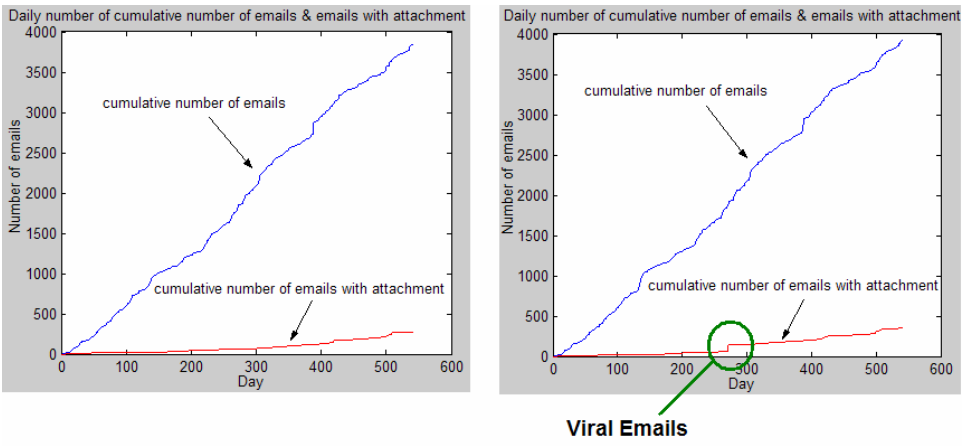


Fig. 15. Cumulative distribution analysis of emails with and without attachments.

7.2 Combing all three models

Here we test the application of the three combined models, Hellinger distance, violations of the cumulative distribution, and clique violations using the same Backward/Forward scanning algorithm described earlier in Section 6.1. The dataset for this test is the same data used in the previous tests described in Section 6.2, which includes 53,163 emails from 15 users; approximately 8% of these emails contain attachments. The detailed performance results are provided in Tables 12-19, and the average ROC plots are displayed in Figure 16. The false positive rate has now substantially dropped down to 0.9%. (Note, the two right side plots of Figure 16 are scaled by a factor of 10^3. Some false positives remain simply because sometimes users form new cliques, and these situations cause false positives, which we cannot entirely avoid without modeling the dynamics of clique formation. (That remains as future research with preliminary results reported in [7].)

Table 12: Outbound Detection Rate (TP). Parameter: varying number of recipients in a single email. Miss: the average number of virus emails missed before a propagation is detected.

	1	2	3	4	5	6	7	8	9
User 1	0.0	0.893	1.0	1.0	0.993	1.0	0.987	1.0	1.0
User 2	0.0	0.97	0.99	0.97	1.0	1.0	0.96	0.99	1.0
User 3	0.0	0.595	0.9	0.93	0.93	0.93	0.925	0.965	0.96
User 4	0.0	0.475	0.75	0.905	0.94	0.97	0.97	0.985	0.98
User 5	0.0	0.175	0.535	0.61	0.87	0.945	0.92	0.93	0.925
User 6	0.0	0.52	0.51	0.675	0.84	0.78	0.83	0.755	0.83
User 7	0.0	0.93	0.97	0.99	0.985	0.985	0.99	0.99	0.985

User 8	0.0	0.76	0.98	0.985	0.975	0.935	0.965	0.975	0.975
User 9	0.0	0.48	0.705	0.92	0.905	0.96	0.96	0.975	0.985
User 10	0.0	0.505	0.78	0.9	0.985	0.94	0.965	0.98	0.99
User11	0.0	0.6	0.675	0.65	0.735	0.74	0.73	0.71	0.895
User 12	0.0	0.96	0.95	0.99	0.95	1.0	1.0	0.98	0.95
User 13	0.0	0.93	0.995	0.975	0.985	0.99	0.995	0.975	0.985
User 14	0.0	0.94	0.967	0.96	0.993	1.0	0.973	1.0	1.0
User 15	0.0	0.91	0.93	0.99	1.0	0.96	0.99	0.99	1.0
Avg.	0.0	0.71	0.842	0.9	0.939	0.942	0.944	0.947	0.964
Miss	NA	1.03	0.608	0.453	0.292	0.383	0.196	0.327	0.177

Table 13. Outbound False Positive (FP). Parameter: varying number of recipients in a single email.

	1	2	3	4	5	6	7	8	9
User 1	0.0949	0.0949	0.0949	0.0995	0.0814	0.0814	0.0768	0.0814	0.0768
User 2	0.0024	0.0024	0.0024	0.0024	0.0024	0.0024	0.0024	0.0024	0.0024
User 3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 7	0.0056	0.0056	0.0056	0.0056	0.0056	0.0056	0.0056	0.0056	0.0056
User 8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 9	0.0056	0.0056	0.0056	0.0056	0.0056	0.0056	0.0056	0.0056	0.0056
User 10	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User11	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 12	0.0171	0.0171	0.0171	0.0171	0.0171	0.0171	0.0171	0.0171	0.0171
User 13	0.0056	0.0056	0.0056	0.0056	0.0056	0.0056	0.0056	0.0056	0.0056
User 14	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 15	0.0034	0.0034	0.0034	0.0034	0.0034	0.0034	0.0034	0.0034	0.0034
Avg.	0.0090	0.0090	0.0090	0.0093	0.0081	0.0081	0.0078	0.0081	0.0078

Table 14. Outbound Detection Rate (TP). Parameter: varying inter-departure time (in minutes). Miss: the average number of virus emails missed before a propagation is detected.

	0	10	15	30	60	720	2880	7200
User 1	1.0	0.987	0.987	1.0	1.0	0.98	0.847	0.727
User 2	1.0	0.98	0.99	0.99	0.95	0.94	0.81	0.77
User 3	1.0	0.91	0.935	0.97	0.9	0.965	0.68	0.58
User 4	1.0	0.925	0.935	0.895	0.845	0.72	0.41	0.285
User 5	1.0	0.624	0.635	0.68	0.745	0.65	0.475	0.285
User 6	1.0	0.74	0.78	0.665	0.825	0.894	0.715	0.405
User 7	1.0	0.99	0.99	0.965	0.96	0.96	0.945	0.99
User 8	1.0	0.94	0.935	0.92	0.985	0.95	0.47	0.54
User 9	1.0	0.875	0.9	0.945	0.88	0.765	0.55	0.75
User 10	1.0	0.89	0.975	0.94	0.91	0.96	0.82	0.87
User11	1.0	0.745	0.895	0.94	0.87	0.91	0.87	0.685
User 12	1.0	1.0	0.99	0.815	1.0	0.89	0.52	0.42
User 13	1.0	0.97	0.995	0.95	0.955	0.945	0.76	0.56
User 14	1.0	0.99	1.0	0.975	0.993	0.853	0.72	0.613
User 15	1.0	1.0	0.97	0.98	0.97	0.86	0.86	0.85
Avg.	1.0	0.905	0.927	0.911	0.919	0.883	0.697	0.636
Miss	0.0	0.41	0.356	0.373	0.351	0.364	0.71	0.733

Table 15. Outbound False Positive (FP). Parameter: varying inter-departure time (in minutes).

	0	10	15	30	60	720	2880	7200
User 1	0.0995	0.0995	0.0995	0.0995	0.0995	0.0995	0.0995	0.0995
User 2	0.0102	0.0102	0.0102	0.0102	0.0102	0.0102	0.0102	0.0102
User 3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 7	0.0056	0.0056	0.0056	0.0056	0.0056	0.0056	0.0056	0.0056
User 8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 9	0.0056	0.0056	0.0056	0.0056	0.0056	0.0056	0.0056	0.0056
User 10	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User11	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 12	0.0239	0.0171	0.0239	0.0171	0.0171	0.0171	0.0171	0.0171
User 13	0.0056	0.0056	0.0056	0.0056	0.0056	0.0056	0.0056	0.0056
User 14	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 15	0.0034	0.0034	0.0034	0.0034	0.0034	0.0034	0.0034	0.0034
Avg.	0.0103	0.0098	0.0103	0.0098	0.0098	0.0098	0.0098	0.0098

Table 16. Inbound Detection Rate (TP). Parameter: varying number of recipients in a single email. Miss: the average number of virus emails missed before a propagation is detected.

	1	2	3	4	5	6	7	8	9
User 1	0.0	0.987	0.987	0.967	1.0	1.0	1.0	1.0	1.0
User 2	0.0	1.0	0.99	0.96	0.98	1.0	1.0	1.0	1.0
User 3	0.0	0.955	0.97	0.94	0.91	0.945	0.975	0.975	0.98
User 4	0.0	0.995	0.995	0.98	1.0	0.98	1.0	0.995	1.0
User 5	0.0	0.995	0.905	0.95	0.965	0.815	0.795	1.0	0.99
User 6	0.0	1.0	1.0	1.0	0.965	0.98	0.98	0.995	1.0
User 7	0.0	0.79	0.885	0.98	0.935	0.945	0.955	0.965	0.995
User 8	0.0	0.995	0.94	0.945	0.845	0.955	0.965	0.9	0.845
User 9	0.0	0.815	0.73	0.78	0.8	0.935	0.795	0.82	0.885
User 10	0.0	0.895	0.9	0.945	0.94	0.96	0.93	0.94	0.95
User11	0.0	0.855	0.845	0.73	0.895	0.8	0.865	0.85	0.905
User 12	0.0	0.99	0.98	0.98	1.0	1.0	0.98	1.0	1.0
User 13	0.0	1.0	0.995	0.99	0.985	0.99	0.95	1.0	0.985
User 14	0.0	0.973	0.993	0.993	0.987	1.0	0.993	1.0	1.0
User 15	0.0	0.967	0.953	0.993	0.993	1.0	1.0	0.987	1.0
Avg.	0.0	0.947	0.938	0.942	0.947	0.953	0.946	0.962	0.969
Miss	NA	0.246	0.394	0.398	0.41	0.34	0.368	0.236	0.227

Table 17. Inbound False Positive (FP). Parameter: varying number of recipients in a single email.

	1	2	3	4	5	6	7	8	9
User 1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 2	0.0478	0.0478	0.0444	0.0444	0.0410	0.0410	0.0410	0.0410	0.0410
User 3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 5	0.0056	0.0056	0.0056	0.0056	0.0056	0.0056	0.0056	0.0056	0.0056
User 6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 10	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

User11	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 12	0.0034	0.0034	0.0034	0.0034	0.0034	0.0034	0.0034	0.0034	0.0034
User 13	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 14	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 15	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Avg.	0.0038	0.0038	0.0036	0.0036	0.0033	0.0033	0.0033	0.0033	0.0033

Table 18. Inbound Detection Rate (TP). Parameter: varying inter-arrival time (in minutes). Miss: the average number of virus emails missed before a propagation is detected.

	0	10	15	30	60	720	2880	7200
User 1	1.0	0.973	0.987	0.973	0.993	0.94	0.733	0.727
User 2	1.0	1.0	1.0	0.98	0.98	0.79	0.78	0.64
User 3	1.0	0.905	0.95	0.88	0.82	0.95	0.885	0.93
User 4	1.0	0.985	0.995	1.0	0.995	0.995	0.86	0.905
User 5	1.0	0.945	0.955	0.98	0.92	0.74	0.625	0.77
User 6	1.0	0.99	1.0	0.98	0.995	0.97	0.755	0.76
User 7	1.0	0.935	0.87	0.92	0.885	0.79	0.94	0.97
User 8	1.0	0.955	0.86	0.94	0.9	0.965	0.635	0.68
User 9	1.0	0.84	0.93	0.82	0.595	0.71	0.475	0.16
User 10	1.0	0.97	0.885	0.885	0.92	0.79	0.705	0.75
User11	1.0	0.795	0.935	0.77	0.645	0.915	0.885	0.805
User 12	1.0	0.98	0.95	0.98	0.98	0.87	0.79	0.78
User 13	1.0	0.995	0.99	0.97	0.98	0.865	0.7	0.665
User 14	1.0	0.967	1.0	0.987	1.0	0.973	0.647	0.673
User 15	1.0	0.973	0.973	0.953	0.96	0.813	0.72	0.707
Avg.	1.0	0.947	0.939	0.935	0.905	0.872	0.742	0.728
Miss	0.0	0.368	0.413	0.331	0.524	0.507	0.742	0.72

Table 19. Inbound False Positive (FP). Parameter: varying inter-arrival time (in minutes).

	0	10	15	30	60	720	2880	7200
User 1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 2	0.0444	0.0444	0.0444	0.0444	0.0444	0.0444	0.0444	0.0444
User 3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 5	0.0056	0.0056	0.0056	0.0056	0.0056	0.0056	0.0056	0.0056
User 6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 10	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User11	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 12	0.0068	0.0068	0.0068	0.0068	0.0068	0.0068	0.0068	0.0068
User 13	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 14	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 15	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Avg.	0.0038	0.0038	0.0038	0.0038	0.0038	0.0038	0.0038	0.0038

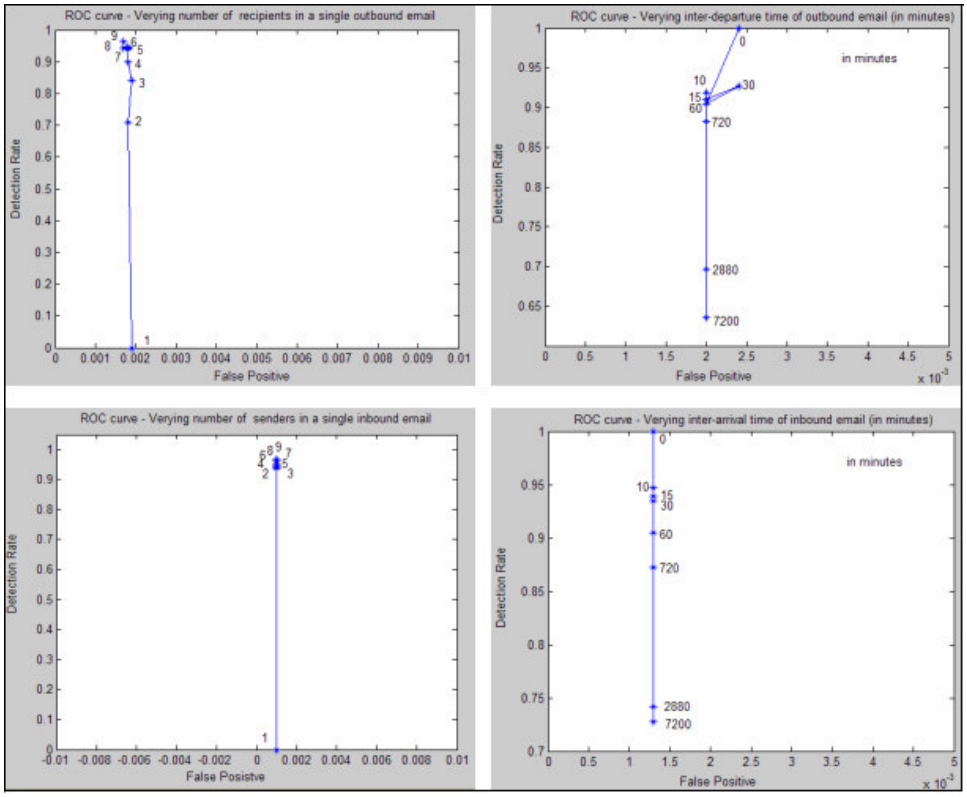


Fig. 16. Average results of each test

Upper-Left: Outbound test, varying number of recipients in a single email.

Upper-Right: Outbound test, varying inter-departure time (in minutes).

Lower-Left: Inbound test, varying number of recipients in a single email. Lower-Right: Inbound test, varying inter-arrival time (in minutes).

7.3 EMT Examples

Here we demonstrate some real examples of the virus simulation test of EMT. Table 20 displays a "dummy" injected viral email detected by the EMT models; the generated alerts are labeled "Y", while "N" denotes normal. In the example, we can see that this user communicates with many different organizations. A virus searches the address book, picks up recipients randomly and sends itself four times. Each of the viral emails includes 4 recipients, and the propagation rate (inter-departure time) is a randomly chosen schedule, from 0 to 30 minutes. The Hellinger model considers each sender-recipient pair as an individual event. An email sent to four recipients will be four records. The Hellinger model detects the anomalous email record after the second email (fifth record). These emails are all on the same day (2002-10-09), and the cumulative attachment distribution indicates an unusual jump in the rate of emission on this day. So, all of the

emails with attachments for this user on that day had alerts issued. Note, too that all of the emails violate the user's normal cliques. We can also see that the backward/forward scanning algorithm can recover the first viral email missed by the Hellinger model.

Table 21, however, displays a false positive. In this case the user uncharacteristically sent an email with an attachment to different domains never before seen in the user's history. Thus, a clique violation occurred, confirmed by a high rate of email emissions on that day causing a false alarm. Finally, Figure 17 displays a screenshot of EMT's GUI representation of this information as models are applied to email archives.

Table 20. Real injected viral emails. He: Hellinge Model, Cu: Cumulative Analysis, Cl: Clique Model

	He	Cu	Cl	Combination	MailRef	Time	Recipient
Email 1	N	Y	Y	Y	NA	2002-10-09 08:12:40	B1@baka.org
	N	Y	Y	Y	NA	2002-10-09 08:12:40	CU1@cs.columbia.edu
	N	Y	Y	Y	NA	2002-10-09 08:12:40	P1@pingnet.com
	N	Y	Y	Y	NA	2002-10-09 08:12:40	P2@pingnet.com
Email 2	Y	Y	Y	Y	NA	2002-10-09 08:32:12	A1@allianttech.com
	Y	Y	Y	Y	NA	2002-10-09 08:32:12	CU2@columbia.edu
	Y	Y	Y	Y	NA	2002-10-09 08:32:12	Ta1@tamashunas.com
	Y	Y	Y	Y	NA	2002-10-09 08:32:12	CU3@cs.columbia.edu
Email 3	Y	Y	Y	Y	NA	2002-10-09 08:33:59	CU4@cs.columbia.edu
	Y	Y	Y	Y	NA	2002-10-09 08:33:59	L1@lucent.com
	Y	Y	Y	Y	NA	2002-10-09 08:33:59	N2@nic.com
	Y	Y	Y	Y	NA	2002-10-09 08:33:59	P3@pingnet.com
Email 3	Y	Y	Y	Y	NA	2002-10-09 08:54:41	Ot1@outpost.tanis.org
	Y	Y	Y	Y	NA	2002-10-09 08:54:41	CU5@cs.columbia.edu
	Y	Y	Y	Y	NA	2002-10-09 08:54:41	Acm1@acm.org
	Y	Y	Y	Y	NA	2002-10-09 08:54:41	A2@allianttech.com

Table 21. Real false positive. He: Hellinger Model, Cu: Cumulative Analysis, Cl: Clique Model

He	Cu	Cl	Combination	MailRef	Time	Recipient
N	Y	Y	Y	1024036921	2002-06-14 02:42:01.0	Student1@cs.ucsb.edu
N	Y	Y	Y	1024036921	2002-06-14 02:42:01.0	Student2@cs.ucsb.edu
Y	Y	Y	Y	1024036921	2002-06-14 02:42:01.0	Employee1@ibm.com
Y	Y	Y	Y	1024036921	2002-06-14 02:42:01.0	CU1@columbia.edu
Y	Y	Y	Y	1024036921	2002-06-14 02:42:01.0	CU2@columbia.edu
Y	Y	Y	Y	1024036921	2002-06-14 02:42:01.0	CU3@columbia.edu
N	Y	Y	Y	1024036921	2002-06-14 02:42:01.0	CU4@cs.columbia.edu
N	Y	Y	Y	1024036921	2002-06-14 02:42:01.0	User1@icir.org

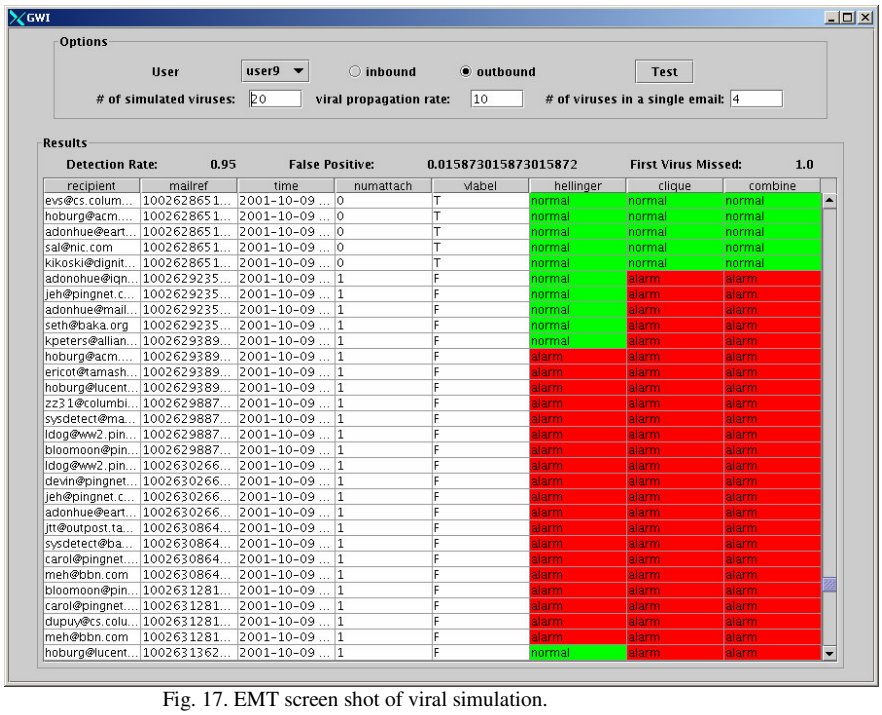


Fig. 17. EMT screen shot of viral simulation.

Virus writers are constantly devising new ways of beating detection algorithms. The behavior-based models presented in this paper provide an alternative to content-based detection methods and can serve as a foundation to detect new viruses. There are numerous avenues of research these techniques suggest. Several are described in our concluding remarks.

8. CONCLUDING REMARKS

We have introduced in this paper several email behavior-based methods using principled statistical analysis techniques and described how these notions can be used in detecting viral email propagations. These methods complement traditional signature-based approaches to virus detection, and are aimed at detecting new viruses for which signatures have not yet been developed and deployed. This is the maximal period of vulnerability when a new virus does its damage.

In general, we find that fast and broad-based viral propagations sent to many victims are very easy to detect using behavior based techniques without content-based analyses. Stealthy and slow moving propagations remain a challenge. (We hope the next generation of viruses do not seize upon this opportunity, but they probably will.)

In particular, we have defined user cliques, and user email frequency behavior profiles. Three specific modeling techniques were combined, user cliques, Hellinger distance and the daily cumulative distribution of emails, to achieve high detection rates with remarkably good FP rates. Tests on outbound traffic indicate that using EMT's combined models, a high detection rate can be achieved: 95% or more in general cases with an FP rate ranging from about 0.38% in the best case to as high as 9% in the worst cases of very slow and stealthy propagations. Tests on inbound traffic show similar results.

The FP rate would translate to a different daily false alarm rate depending upon the user's email emission rate. In the general case, where one user's data exhibited one email per day with attachments, the outbound FP rate of 2% suggests that that user would have received one false alarm every 45 days for the outbound email. The FP rate of 1% of the user's inbound email suggests a false alarm once every 90 days.

There are several areas of new research that can provide substantial improvement in several respects. We chose to use threshold settings for the models based upon a static prior period of time (the window size in Hellinger, *d* days in the cumulative distribution) and did not attempt to incorporate user-specific calibration. As noted in prior work, the choice of such parameters has a very big impact on detection performance (see Maxion [33]). Calibrating a detector and setting parameters specific to a user would logically improve individual detection performance for each user. Our current research includes strategies and techniques to best calibrate the detector for each user.

Furthermore, detection is performed on a “per user profile” basis; we do not yet have performance results at the “enclave” level. This is particularly interesting in that shared statistics among a group of users would naturally inform a model more precisely about the onset of a viral propagation within an organization served by a single mailserver. Any infected user would naturally propagate to members of their own organization (those the user frequently communicates with) and the combined statistics among multiple users would tend to favor early detection for all users. These shared statistics would make a viral propagation appear faster moving than would otherwise be seen by an individual victim. Furthermore, it is sensible that there would be a higher likelihood of detecting clique violations at the enclave level where more email traffic may be inspected.

The models used here for clique violation are not only specific to a user, they are also static. We chose to analyze a user’s historical cliques without modeling the dynamics of clique formation and expiration. Clearly, conversations with different groups of folks would tend to be revealed by considering the dynamics of the interaction between the parties to a conversation. This is the subject matter of recent work by Kleinberg [17]

where he considers the onset of new "content", rather than new viral propagations, by stochastically modeling the flow of subject lines in email streams. Our current work is focused on modeling dynamic clique formations [7] which would logically improve the performance of clique violation models that consider shorter term statistics.

We are presently updating EMT by incorporating the MEF content-based modeling techniques [30]. MEF uses a Naïve Bayes classifier over attachment contents to produce a probability that an attachment is malicious. In this paper, all attachments were regarded as one type of object. We made no use of the file extension associated with the attachment to suggest different types of content and thus different statistics that this may derive. Even so, there may be no means of determining malicious content solely by file extension type. MEF analyzes the attachment content and outputs a score representing the likelihood the attachment is malicious. Our earlier worked cast the problem as a multi-class supervised training problem where viral attachments and non-malicious attachments were provided as positive and negative training data, respectively. Our recent work has cast this as a "one-class" supervised training problem that has the practical advantage of training against known viruses alone. Here we intend to learn statistical models that score the likelihood that an attachment is malicious and combine these with the models explored in this paper.

It is possible that EMT may be thwarted by viruses that mimic the user's behavior revealed in the user's email client sent folder. There are several defenses against this strategy of mimicry attack [35] that are the subject matter of our ongoing research in behavior-based detection. For example, we aim to integrate EMT with the FWRAP system [12] that detects abnormal file system accesses, such as the attempted access of email data by a virus or malware.

Lastly, the focus of this paper has been on viral propagation detection as an example of the power of behavior-based computer security. This concept is applicable to a far wider range of problems, including spam detection, security policy violations, and a host of other detection tasks. We believe EMT thus serves as a model anomaly detection system for any audit stream and detection problem of interest. This work suggests a general framework that is the subject matter of our ongoing work. This framework posits that anomaly detection is best viewed and cast as a problem to optimally correlate multiple detectors, where each detector models normal behavior using different features of the audit stream. These detectors generate alerts when there are violations of volume and velocity statistics, anomalous values exhibited in an audit stream, and abnormal or inconsistent formation of vertices when viewing data in the audit stream in graph

theoretic formulations. All of these concepts and modeling techniques are embodied in EMT.

REFERENCES

1. R. AGRAWAL, T IMIELINSKI, AND A. SWAMI. 1993. Mining Association Rules between Sets of Items in Large Databases. In *Proceedings of the ACM SIGMOD Int'l Conf. on the Management of Data*, pp. 207-216.
2. FRANK APAP, ANDREW HONIG, SHLOMO HERSHKOP, ELEAZAR ESKIN, AND SALVATORE J. STOLFO. 2002. Detecting Malicious Software by Monitoring Anomalous Windows Registry Accesses. In *Proceedings of the Fifth International Symposium on Recent Advances in Intrusion Detection (RAID-2002)*. Zurich, Switzerland: October 2002, 16-18.
3. M. BHATTACHARYYA, S. HERSHKOP, E. ESKIN, AND S. J. STOLFO. 2002. MET: An Experimental System for Malicious Email Tracking. In *Proceedings of the 2002 New Security Paradigms Workshop (NSPW-2002)*, Virginia Beach, VA, September 2002.
4. ZHIQIANG BI, CHRISTOS FALOUSTOS, AND FLIP KORN. 2001. The DGX Distribution for Mining Massive, Skewed Data. In *Proceedings of the 7th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining*, pp. 17-26.
5. C. BRON, AND J. KERBOSCH. 1973. Finding all cliques of an undirected graph. *Comm. ACM 16(9)*, pp. 575-577.
6. M. DAMASHEK. 1995. Gauging Similarity With n-grams: Language Independent Categorization of Text. In *Science*, 267(5199), pp. 843-848.
7. PETER T. DAVIS. 2003. Finding Friends and Enemies through the Analysis of Clique Dynamics. *Technical report*.
8. D. E. DENNING. 1987. An Intrusion-Detection Model. *IEEE Transactions on Software Engineering*, SE-13, pp. 222-232.
9. ELEAZAR ESKIN. 2000. Anomaly Detection Over Noisy Data Using Learned Probability Distributions. In *Proceedings of the 17th Int'l Conf. on Machine Learning (ICML-2000)*.
10. E. ESKIN, A. ARNOLD, M. PRERAU, L. PORTNOY AND S. J. STOLFO. 2002. A Geometric Framework for Unsupervised Anomaly Detection: Detecting Intrusions in Unlabeled Data. *Kluwer*.
11. ANUP K. GHOSH, AARON SCHWARTZBARD, MICHAEL SCHATZ 1999. Learning Program Behavior Profiles for Intrusion Detection. *Workshop on Intrusion Detection and Network Monitoring* 1999: 51-62.
12. SHLOMO HERSHKOP, RYAN FERSTER, LINH H. BUI, KE WANG AND SALVATORE J. STOLFO. 2003. Host-based Anomaly Detection by Wrapping File System Accesses. *CU Tech Report*, http://www.cs.columbia.edu/ids/publications/, April 2003.
13. S. A. HOFMEYR, STEPHANIE FORREST, AND A. SOMAYAJI. 1998. Intrusion Detection Using Sequences of System Calls. *Journal of Computer Security*, 6, 1998, pp. 151-180.
14. R.V. HOGG AND A.T. CRAIG. 1994. *Introduction to Mathematical Statistics*, Prentice Hall, pp. 293-301.
15. H. S. JAVITZ AND A. VALDES. 1993. The NIDES Statistical Component: Description and Justification. *Technical report, SRI International*.
16. GEORGE H. JOHN AND PAT LANGLEY. 1995. Estimating continuous distributions in Bayesian classifiers. In *Proceedings of the 11th Conf. on Uncertainty in Artificial Intelligence*, pp. 338-345.
17. JON KLEINBERG. 2002. Bursty and Hierarchical Structure in Streams. In *Proceedings 8th ACM SIGKDD Int'l Conf on Knowledge Discovery and Data Mining*, pp. 91-101.
18. TERRAN LANE AND CARLA E. BRODLEY. 1999. Temporal Sequence Learning and Data Reduction for Anomaly Detection. *ACM Transactions on Information and System Security*, 2:295-331.
19. WENKE LEE AND SALVATORE STOLFO. 1999. A Framework for Constructing Features and Models for Intrusion Detection Systems. In *Proceedings of 1999 IEEE Symposium on Computer Security and Privacy and the Proceedings of the 8th ACM SICKDD international Conference on Knowledge Discovery and Data Mining.*
20. WENKE LEE, SAL STOLFO, AND PHIL CHAN. 1997. Learning Patterns from Unix Process Execution Traces for Intrusion Detection. *AAAI Workshop: AI Approaches to Fraud Detection and Risk Management,* July 1997.
21. WENKE LEE, SAL STOLFO, AND KUI MOK. 1998. Mining Audit Data to Build Intrusion Detection Models. In *Proceedings of the 4th Int'l Conf. on Knowledge Discovery and Data Mining (KDD '98)*, New York, NY, August 1998.
22. WENKE LEE, S. J. STOLFO, AND K. MOK. 1999. Mining in a Data-flow Environments: Experiences in Intrusion Detection. In *Proceedings of the 1999 Conf. on Knowledge Discovery and Data Mining (KDD-99)*.

23. WENKE LEE AND DONG XIANG. 2001. Information-theoretic measures for Anomaly Detection. In *Proceedings of the 2001 IEEE Symposium on Security and Privacy*. May, 2001.
24. MATTHEW V. MAHONEY AND PHILIP K. CHAN. 2001. Detecting Novel Attacks by Identifying anomalous Network Packet Headers. *Florida Institute of Technology Technical Report* CS-2001-2.
25. TOM M MITCHELL. 1997. *Machine Learning*, McGraw-Hill, pp. 180-183.
26. MYSQL, 2002. www.mysql.org.
27. M.E. NEWMAN, S. FORREST AND J. BALTHRUP. 2002. Email networks and the spread of computer viruses. *The American Physical Society*.
28. W. NIBLACK ET AL. 1993. The QBIC project: querying images by content using color, texture, and shape. In *Proceedings of the SPIE*, February 1993.
29. M. SCHONLAU, W. DUMOUCHEL, W. JU, A.F. KARR, M THEUS AND Y. VARDI. 2001. Computer Intrusion Detecting Masquerades. *Statistical Science*, *Vol. 16*.
30. MATTHEW G. SCHULTZ, ELEAZAR ESKIN, AND SALVATORE J. STOLFO. 2001. Malicious Email Filter - A UNIX Mail Filter that Detects Malicious Windows Executables. In *Proceedings of USENIX Annual Technical Conference - FREENIX Track*. Boston, MA: June 2001.
31. J.R. SMITH. 1997 Integrated Spatial and Feature Image Systems: Retrieval, Compression and Analysis. *Ph.D. Thesis*, Columbia University.
32. SALVATORE J. STOLFO, SHLOMO HERSHKOP, KE WANG, OLIVIER NIMESKERN, AND CHIA-WEI HU. 2003. Behavior Profiling of Email. *1st NSF/NIJ Symposium on Intelligence & Security Informatics(ISI 2003).* Tucson, Arizona,USA.
33. KYMIE M. C. TAN AND ROY A. MAXION. 2002. Why 6? Defining the Operational Limits of stide, an Anomaly-Based Intrusion Detector. *IEEE Symposium on Security and Privacy*, *IEEE Computer Society Press*, Los Alamitos, California, pp. 188-201.
34. C. TAYLOR AND J. ALVES-FOSS. 2001. NATE: Network Analysis of Anomalous Traffic Events, a low-cost approach. In *Proceedings New Security Paradigms Workshop*. Sept 2001, pp 89-96.
35. DAVID WAGNER AND PAOLO SOTO. 2002. Mimicry Attacks on Host-Based Intrusion Detection Systems. In *Proceedings of the 9th ACM Conf. on Computer and Communications Security (CCS)*, Washington, DC, November 2002. pp. 255-264.
36. CHRISTINA WARRENDER, STEPHANIE FORREST, AND BARAK PEARLMUTTER. 1999. Detecting Intrusions Using System Calls: Alternative Data Models. *IEEE Computer Society.*
37. DUNCAN J. WATTS. 2003. *Six Degrees: The Science of a Connected Age*, W.W. Norton & Company, Feb 2003.
38. M.M. WILLIAMSON. 2002. Throttling viruses: Restricting propagation to defeat malicious mobile code. In *Proceedings ACSAC Security Conference*, Las Vegas, NV, 2002.
39. NONG YE. 2000. A Markov Chain Model of Temporal Behavior for Anomaly Detection. In *Proceedings of the 2000 IEEE Workshop on Information Assurance and Security*, United States Military Academy, West Point, NY, 6-7 June, 2000.
40. SAL STOLFO, PHIL CHAN, ANDREAS PRODROMIDIS. 1999. Distributed Data Mining in Credit Card Fraud Detection, *IEEE Intelligent Systems*, Vol. 14, No. 6, 1999.