MODELS TO COMBAT EMAIL SPAM BOTNETS AND UNWANTED PHONE CALLS

Husain Husna, B.E.

Thesis Prepared for the Degree of

MASTER OF SCIENCE

UNIVERSITY OF NORTH TEXAS

May 2008

APPROVED:

Ram Dantu, Major Professor
Armin Mikler, Committee Member
Parthasarathy Guturu, Committee Member
Krishna Kavi, Chair of the Department of Computer Science and Engineering
Oscar N. Garcia, Dean of the College of Engineering
Sandra L. Terrell, Dean of the Robert B. Toulouse School of Graduate Studies

Husna, Husain. Models to Combat Email Spam Botnets and Unwanted Phone Calls. Master of Science (Computer Science and Engineering), May 2008, 91 pp., 12 tables, 45 illustrations, 90 references.

With the amount of email spam received these days it is hard to imagine that spammers act individually. Nowadays, most of the spam emails have been sent from a collection of compromised machines controlled by some spammers. These compromised computers are often called bots, using which the spammers can send massive volume of spam within a short period of time.

The motivation of this work is to understand and analyze the behavior of spammers through a large collection of spam mails. My research examined a the data set collected over a 2.5-year period and developed an algorithm which would give the botnet features and then classify them into various groups. Principal component analysis was used to study the association patterns of group of spammers and the individual behavior of a spammer in a given domain. This is based on the features which capture maximum variance of information we have clustered.

Presence information is a growing tool towards more efficient communication and providing new services and features within a business setting and much more. The main contribution in my thesis is to propose the willingness estimator that can estimate the callee's willingness without his/her involvement, the model estimates willingness level based on call history. Finally, the accuracy of the proposed willingness estimator is validated with the actual call logs.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

This thesis aims at providing techniques that analyzes a user's or a group of users' behavior using linear algebra, statistical analysis, clustering algorithms and information theory. The model proposed consist of fundamentals which can be used in various applications for combating spam botnets, for quantifying the willingness of a phone user thereby reducing network congestion and also identifying the structure of a group/user in a social network. Chapter 1 presents the motivation for this work, describes what I hoped to accomplish, and the approach and limitations that are specific to this work. The remainder of this thesis is divided in to the following chapters:

(i) Introduction
(ii) Related Works and our Contribution
(iii) Principal Factor Analysis
(iv) Grouping Algorithms
(v) Information Theory: Entropy
(vi) Behavior Analysis of Spam Botnets
(vii) Behavior Analysis of Callee for Unwanted Calls
(viii) Behavioral Analysis using Entropy

1.1. Motivation for Behavior Analysis of Spam Botnets

In recent years, a wide variety of spam filtering techniques have gained popularity. Unfortunately, these techniques, though being widely used, do not allow us to discover common trends or variations shared by a majority of the spammers spamming within a given domain. Spam filtering using statistical analysis, email authentication standards, and trust and reputation techniques do not identify common association patterns among a specific set

of spammers. In this thesis, I identify underlying spamming patterns by applying principal component analysis(PCA) and clustering techniques.

The researched classifier applies principal component analysis to analyze both the association patterns of groups of spammers and the individual behavior of a spammer in a given domain. With the aid of PCA it tries to find possible association patterns such as,

- Spammers spamming at the same time of the day
- Spammers repeatedly sending the same content
- Spammers changing email id's and spamming the same recipient again
- Spammers sharing contact lists

From analysis I found that, for multiple recipients' in a domain, the kinds of spam received are diverse enough to render common association patterns between spammers within that environment insignificant. However, when analyzed recipients individually, the common association patterns emerge as more significant.

Further analysis show individual spammer association patterns periodically and found a consistent trend. Because most spammers show common association patterns, or behavior over the time, I identify such patterns as the the spammers' eigen-behaviors , which can then be used to characterize individual spammers and to describe their association behaviors. To achieve this, I applied PCA and clustering techniques, such as hierarchical and K-means, on three email corpuses to validate findings with recipients' preferences.

With the amount of email spam received these days it is hard to imagine that spammers act individually. Most of the spam emails have been sent from a collection of compromised machines (often called bots) that the spammers control. About 30,000 new machines are compromised daily, and become bots. One of the most common usages of botnets is to launch massive spam. These bots allow spammers to send a massive volume of spam within a short time period. According to the recent survey [1] [2] [3] [4] [5] an estimated 80% of email spam is sent through such zombie PCs. Spam remains an annoying problem because a majority of the spam filtering techniques focus on the email's content, which the spammers control. To circumvent traditional spam filter techniques, most spammers and phishers obfuscate their

email content to circumvent these spam filters. So, such techniques have little use as their categories depend on the message's meaning. My definition does not have this limitation as it is based on an individual user's behavior.

I classified each spammer's behavior based on the features of its header contents. Therefore, the analysis does not focus on email's content. Thus it is irrelevant if spammers obfuscate the content of an email. I categorized each email spammer based on features such as IP address, content length, time of arrival, frequency of spamming, content type e.g. 'MIME-Version' - used for encoding binary content as attachments. Because these are spammers, other parameters (for example, reciprocity, read emails, and storage time) need not apply; if we assume users do not read telemarketing emails.

On each spammer's feature set (a huge data set matrix), I applied PCA to identify those features which captured most of the variance present in the data set. Further I clustered spammers into groups based on their behavior patterns. There is a possibility of a spammer spamming multiple receivers in the same domain. Thus, I could find some common behavior patterns for a group of bots by finding the proximity between senders using euclidean distance. Spammers were clustered using various algorithms like K-means and hierarchical. There are several challenges in clustering spammers based on their behavior. One such is that if a host is running dynamic host configuration protocol (DHCP), the host IP address could change from several hours to several days. The change of IP addresses could change the spammers' clustering structure. After forming groups I manually verified each botnet group by looking at its feature set, proximity values, and location of spammer and we were able to achieve a precision of 90%. Thus, using the technique, which may effectively block spammers as groups instead of blocking them individually.

In this thesis I also propose a mechanism to classify incoming traffic into buckets of legitimate and suspicious spam. I computed the entropy values for parameters such as active time, time of arrival, frequency, and content length which were extracted using header analysis. Based on these values, I sub-classified suspicious bucket into spammers and botnet spam. Next, I detected and grouped the spam mail into botnets based on the entropy

values. Also, a high correlation of entropy values for a group of bots was identified; and, this information assisted to classify the botnets more finely. For future work, based on the entropy based classification, I plan to delay the traffic generated from these bots and believe that this delay can prevent spam, by setting up a Turing Test on all the suspicious and spam mails. This can further block the spam from these sources.

To validate the effectiveness of the proposed mechanism, I perform three experiments to analyze traffic. The experimental results reveal that the techniques are applicable for detecting botnets and further mitigate future actions with a precision of more than 95%.

To perform experiments, I obtained spam data from a corpus consisting of 5000-6000 emails. These emails had been received at a domain mail server over a one-year period in such a way that the IP addresses were recorded when spammers attempted to establish a TCP connection to transmit their spam mail. Typically, the spammer's IP address recorded during the 3-way handshake is the spammer's true IP address. However, in rare cases a spammer might successfully use a spoofed address during the handshake. Fortunately, as spamming becomes more distributed, there is less successful spoofing of such IP addresses. Attackers already use several levels of indirection to hide their identities and thousands of compromised machines, e.g., the bots, to directly send massive spam mails.

In the experiments reported here, I ignore such cases because blocking potential spam mails from unused IP addresses will never cause false alarms. The spam mail data contains the full mail header information and the full mail contents including the attachment files. The mail header information contains the real IP address of the spam source and the route information, which can be used to identify the information of the spam source. I extracted information fields from these headers such as:

(i) The Sender IP address

(ii) Content length

(iii) Content type

(iv) Time of arrival

1.2. Motivation for Quantifying Presence

Presence information is a recent tool that is allowing more efficient communication, particularly within business settings as it allows provision of new services and features to application users. Presence-aware technology enhances communication between parties with reduced costs and time as it allows information such as who and when a party is available in a corporate network to be available to users. According to [6], presence propagates information about a users willingness, ability, and desire to communicate using a variety of mediums. A caller can subscribe to learn a callee's behavior when accepting and rejecting calls. This ability results in a more precise communication because it eliminates the inefficiency of phone tags. We are already seeing this kind of improved communication in instant messenger (IM) users. IM users can communicate directly with those people who are available at the present. Users do not need to guess whether the person they want to communicate with wishes to speak with them. IM is just a beginning of presence-aware technology evolution. The Gartner research [7] predicted that by 2009, 80 percent of business applications will have presence-aware functionality to support business processing and management of customer relationship and corporate performance.

It is evident that the presence service will become an important feature of communication systems as it provides advantages to both parties in a communication event. The service can reduce disrupting calls for callee's when they are engaged in important work and wish no interruptions. Additionally, callers will know how willing a caller is to receive a call at that specific time and can decide whether to initiate a call or delay to another, more auspicious time. Integrating presence service with SIP protocol can provide multiple medium of communication between parties. Presence-aware technology has received a lot of attention from the researchers [8] [9]. Shan and Shriram's work [27] reduces enterprise server load by mobile clients sharing presence information within the network. In their work, only one of the clients acts as a gateway to interact with the server to supply the network's presence information.

In this thesis, I used the actual call logs for our analysis. These logs were collected at MIT [11] by the Reality Mining Project group for a period of 8 months. The project collected mobile phone use by 100 users(students, professors, and staffs). The data collected consisted of user IDs,(unique number representing a mobile phone user), time of calls, call direction (incoming and outgoing), incoming call description (missed, accepted), talk time, and tower IDs (location of phone users). I used this extensive data set for our willingness estimator (WE) analysis and validation. More information about the reality mining project can be found in [11].

The main contribution of this section is to propose the Willingness Estimator (WE) that can estimate the calle's willingness without his/her involvement, the model estimates willingness level based on call history. I present the methodology and architecture of the Willingness Estimator, refer section 7.1 and 7.2, that computes willingness level of the specified callee. Further, I also describe the willingness computation. Finally, I discuss the accuracy achieved with our proposed WE.

CHAPTER 2

RELATED RESEARCH WORKS AND OUR CONTRIBUTIONS

This chapter gives a brief overview of the current research work on Models to combat email spam especially botnet spam and also in the field of "presence". Each section of this chapter concludes with a brief discussion of how the work reported here compares to or extends the research of others.

2.1. Research in Behavioral/Header Analysis of Spam Botnets

Extensive research is being done combating spam based on content and header analyzes, [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] but few are working on building a spam filter based on header analysis of spammer and categorizing those spammers based on their behavior of spamming a particular domain. Several works available in the current research cater to the design and building models to cluster groups of people, but few focus on email spam.

To our knowledge, only one work reported on group-based anti-spam strategies. Li and Hsieh, [23] conducted an empirical study on the clustering behavior of spammers and detected different groups of spammers. Majority of their emails content is related to money. They found 2% of the spammers accounted to 20% of the spam, whereas 68% of the spammers sent only one spam. They clustered the spammers based on the content, such as, URL and money-amount in the emails. They observed that more URLs specified in the email, then, there is a higher probability of getting further spam from the source IP address of the spammer. They converted ASCII characters in the URL into binary data and calculated complementary cumulative distribution function (CCDF) [23] of spam scores of the spam-groups. They claim to block 70-90% of email based on the URL-based group approach. However, these results are limited to content-based filtering and in particular for URL and money-specified emails. Moreover, they observed money-amount-based clustering structures may not be

effective for group-based anti-spam strategies. I believe that spammers share email lists and use this list for various kinds of advertisements. Therefore, content-based grouping may not work for blocking all types of contents. In contrast, I grouped the spammers based on their behavior and transmission patterns. These patterns show high correlation between group members irrespective of geographic location, network ID, content, and kind of receivers. To our knowledge, there is no work reported on detecting botnets based on their behavior and in particular transmission patterns.

First, I developed a feature set for each spammer, as a data set matrix. I then applied PCA to the feature set to identify features which captures most of the variance in the data set. Further, I clustered these spammers into groups based on their behavior patterns. I considered the possibility that a spammer might spam multiple receivers within the same domain. To examine this I identified common behaviors for a group of spammers by using the proximity between the senders (using a distance metric) and by applying clustering algorithms.

To verify our approach's accuracy, I hand-labeled the data and compared those results to the automatic botnet identification of each corpus. Accuracies around 90% have been achieved. Thus, using the proposed technique, it may effectively block those spammers as groups instead of blocking them individually [24] [25].

2.2. Research in Traffic Shaping of Spam Bots

There is substantial work done in traffic shaping of email channels. One such work which inspired our analysis was by Anirudh Ramachandran, Nick Feamster and David Dagon [26], who focus on identifying members of botnets which could help stem these attacks, but passively detecting botnet membership (i.e., without disrupting the operation of the botnet) proves to be difficult. Ramachandran, *et al.* discusses the effectiveness of monitoring lookups to a DNS-based blackhole list (DNSBL) to expose botnet membership. They perform counter-intelligence based on the insight that botmasters themselves perform DNSBL lookups to determine whether their spamming bots are blacklisted. Using heuristics to identify which DNSBL lookups are perpetrated by a botmaster performing such reconnaissance,

they were able to compile a list of likely bots. Their paper studies the prevalence of DNSBL reconnaissance observed at a mirror of a well-known blacklist for a 45- day period, identifies the means by which botmasters are performing reconnaissance, and suggests the possibility of using counter-intelligence to discover likely bots. They find that bots are performing reconnaissance on behalf of other bots. Based on this finding, they suggest counter intelligence techniques that may be useful for early bot detection.

Our work is inspired by that of Osterneyer [27], who suggest that spammers are aggressive, smart, and continuously adopt new techniques to send increasing volumes of spam. The result is not only an enormous quantity of spam ending up in spam quarantines and users' mailboxes, but also email servers that are brought down by excessive quantities of unwanted content and increasing investments in servers and software to keep up with the growing deluge of spam. What is needed, therefore, is a technique that blocks spam based on spammers readily identifiable behavior without the CPU-intensive activity of content filtering. But I used this idea to come up with a 'pre-filtering' analysis of senders' reputation based on his header pattern analysis. Using this approach, we argue, systems can operate more efficiently because it blocks majority of spam before it reaches the email servers. To extend the "counter-intelligence" metaphor, I propose using a Turing Test technique that is applied to all suspicious traffic categorized from behavioral patterns. With this information, spammers' spam can be more readily identified and blocked.

2.3. Research in Quantifying Presence

Nathan Eagle from MIT, [28] in his work uses call logs to derive conclusions based on behavior analysis. He identifies the structure inherent in daily human behavior using models that accurately analyze, predict, and cluster multi-modal data from individuals and from groups. He represents this structure by the principal components of a complete behavioral dataset, a set of characteristic vectors which they have termed eigenbehaviors. In Eagle's model, an individual's behavior over a specific day can be approximated by a weighted sum of his or her primary eigenbehaviors. When these weights are calculated halfway through a day, the resulting information can be used to predict the day's remaining behaviors. Eagle

reports a 78% accuracy for his test subjects. Additionally, he shows that users of a similar demographic can be clustered into a "behavior space" spanned by a set of their aggregate eigenbehaviors. These behavior spaces make it possible to determine the behavioral similarity between both individuals and groups, enabling 96% classification accuracy of group affiliations. This approach capitalizes on the large amount of rich data previously captured during the reality mining study from mobile phones continuously logging location, proximate people, and communication of 100 subjects at MIT over the course of nine months.

Another related work on the MIT Data set which stands close to our work is "Mining the Mine Exploratory Social Network Analysis of the Reality Mining Dataset" [29]. In this article, Congleton and Nainwal present an exploratory network analysis and derive interesting insights into both the dataset and into patterns which have potential for a more rigorous and analytical exploration. Congleton and Nainwal found an interesting pattern in call reciprocity in the call network. In instances of the call network where there was a triad, i.e. when people called two others while they were in the same location as the other person, the mutual exchange of calls or call reciprocity was higher when two people mostly called each other while they were in the same location. In the latter case, the call exchange was highly asymmetric with one person calling the other most of the time.

The contribution of my thesis is to propose the willingness estimator (WE) which can estimate the callee's willingness without his/her involvement, the model estimates willingness level based on call history. I believe that callee's willingness of accepting a call is influenced by time of the call, location of the callee, and day of the call. The willingness computations based on these three factors are carried out as a simple ratio of the call frequency at particular (time, location, day) to the total number of calls. I believe that contribution of these factors to the overall (final) willingness level are different. It is validated by the principal component analysis that these three factors have different relevant fweights to the final willingness level. Hence, the callee's final willingness level is the sum of the product of the willingness based on time of the call, callee's location, the day of the call, and its corresponding contribution coefficients. The model's accuracy is evaluated with the actual call logs of 10 phone users

from the MIT's Reality Mining data sets. The WE performs well with high accuracy when the computed willingness level is low and the unanswered rate is high, and vice versa. This new area of research uses estimating a callee's presence as a form of willingness(desire to engage in an communication event with a particular caller). Such information can be helpful in many ways. For example, awareness of willingness would allow telemarketers to know exactly when to call so the callee will be most likely to listen. Within the corporate world, knowledge of willingness would provide a technique that allowed individuals to know when a person is most likely to take a call, what type of caller, and whether the person is available (i.e., not busy in a meeting.)

CHAPTER 3

PRINCIPAL FACTOR ANALYSIS

3.1. Definition and Derivation of Principal Components

In this chapter, I discuss the fundamental concepts of principal components and their various computational mathematics involved. Typically, PCA is used to reduce the dimensionality of a data set consisting of a large number of inter-related variables while retaining as much as possible of the variation present in the data set [30] [31] [32]. This is achieved by transforming to a new set of variables, the principal components (PCs), which are uncorrelated, and ordered so that the first few retain most of the variation present in the original variables.

Suppose that X is a vector of p random variables, and I want to infer about the variances of the p random variables and the structure of the covariances or correlations between the p variables. One observes the p variances and $\left\{\frac{1}{2} \times p(p-1)\right\}$ correlations or covariances, which increase in complexity with the increase in size of X. An alternative is to observe for a few derived variables ($<< p$) that preserve most of the information given by these variances and correlations or covariances.

First step is to look for a linear function $\alpha_1' x$ of the elements of x having maximum variance, where α_1 is a vector of p constants α_{11}, α_{12}, α_{1p}, and $'$ denotes transpose, so that

$$\alpha_{1x} = \alpha_{11}x_1 + \alpha_{12}x_2 + ... + \alpha_{1p}x_p = \sum_{j=1}^{p} \alpha_{1j}x_j. \tag{1}$$

Next, I select a linear function $\alpha_2' x$, uncorrelated with $\alpha_1' x$ having maximum variance, and so on, so that at the k_{th} stage a linear function of $\alpha_k' x$ is found, that has maximum variance subject to being uncorrelated with $\alpha_1' x$, $\alpha_2' x$,....., $\alpha_{k-1}' x$. The k_{th} derived variable, $\alpha_k' x$ is the k_{th} PC. Upto p PCs could be found, but it is hoped, in general, that most of the

variance in x will be accounted for by m PCs, where $m<<p$. The advantage of having p=2 or 3 is, of course, that the data can be plotted exactly in 2 or 3 dimensions.

Having defined PCs, now let us know how to find them. Consider, a random variable x has a known covariance matrix $\sum$. This is the matrix whose $(i,j)^{th}$ element is the covariance between the i^{th} and the j^{th} elements of x when $i \neq j$, and the variance of the j^{th} of x when $i = j$. The more realistic case, where $\sum$ is unknown, follows by replacing $\sum$ by a sample covariance matrix S. For k=1,2,..p, the k^{th} PC is given by $z_k = \alpha_k' x$ where α_k is chosen to have unit length ($\alpha_k'\alpha_k$=1), then var(z_k)= λ_k, where var(z_k) denotes the variance of z_k.

To derive the form of the PCs, consider first $\alpha' x$; the vector α_1 maximizes var[$\alpha_1' x$]=$\alpha_1' \sum \alpha_1$. Clearly, the maximum will not be achieved for a finite α_1 so a normalization constraint must be imposed. The constraint used in the derivation is $\alpha_1'\alpha_1$=1, that is, the sum of squares of elements of α_1 equals 1.

To maximize $\alpha_1' \sum \alpha_1$ subject to $\alpha_1'\alpha_1$=1, the standard approach is to use the technique of Lagrange multipliers. Maximize

$$\alpha_1' \sum \alpha_1 - \lambda(\alpha_1'\alpha_1 - 1), \tag{2}$$

where λ is a Lagrange multiplier. Differentiation with respect to α_1 gives

$$\sum \alpha_1 - \lambda\alpha_1 = 0, \tag{3}$$

or

$$(\sum -\lambda I_p)\alpha_1 = 0, \tag{4}$$

where I_p is the (pXp) identity matrix. Thus, λ is an eigenvalue of $\sum$ and α_1 is the corresponding eigenvector. To decide which of the p eigenvectors gives $\alpha_1' x$ with maximum variance, the quantity to be maximized is

$$\alpha_1' \sum \alpha_1 = \alpha_1' \lambda \alpha_1 = \lambda \alpha_1' \alpha = \lambda, \tag{5}$$

so λ must be as large as possible. Thus, α_1 is eigenvector corresponding to the largest eigenvalue of $\sum$, and the var$(\alpha_1' x)$=$\alpha_1' \sum \alpha_1$=λ_1, the largest eigenvalue.

In general, the k^{th} PC of x is $\alpha_k' x$ and $var(\alpha_k' x) = \lambda_k$, where λ_k is the k^{th} largest eigenvalue of $\sum$, and α_k is the corresponding eigenvector.

The second PC, $\alpha_2' x$, maximizes $\alpha_2' \sum \alpha_2$ subject to being uncorrelated with $\alpha_1' x$, or equivalently subject to $cov[\alpha_1' x, \alpha_2' x] = 0$, where $cov(x, y)$ denotes the covariance between the random variables x and y. But

$$(6) \qquad cov[\alpha_1' x, \alpha_2' x] = \alpha_1' \sum \alpha_2 = \alpha_2' \sum \alpha_1 = \alpha_2' \lambda_1 \alpha_1' = \lambda_1 \alpha_2' \alpha_1 = \lambda_1 \alpha_2' \alpha_1 = \lambda_1 \alpha_1' \alpha_2.$$

Thus, any of the following equations

$$(7) \qquad \alpha_1' \sum \alpha_2 = 0, \alpha_2' \sum \alpha_1 = 0, \alpha_1' \alpha_2 = 0, \alpha_2' \alpha_1 = 0$$

could be used to specify zero correlation between $\alpha_1' x$ and $\alpha_2' x$. Choosing the last of these, and noting that a normalization constraint is again necessary, the quantity to be maximized is

$$(8) \qquad \alpha_2' \sum \alpha_2 - \lambda(\alpha_2' \alpha_2 - 1) - \phi \alpha_2' \alpha_1,$$

where λ, ϕ are Lagrange multipliers. Differentiation with respect to α_2 gives

$$(9) \qquad \sum \alpha_2 - \lambda \alpha_2 - \phi \alpha_1 = 0$$

and multiplication of this equation on the left by α_1' gives

$$(10) \qquad \alpha_1' \sum \alpha_2 - \lambda \alpha_1' \alpha_2 - \phi \alpha_1' \alpha_1 = 0,$$

which, since the first two terms are zero and $\alpha_1' \alpha_1 = 1$, reduces to ϕ=0. Therefore, $\sum \alpha_2 - \lambda \alpha_2 = 0$, or equivalently $(\sum - \lambda I_p)\alpha_2 = 0$, so λ is once more an eigenvalue of $\sum$, and α_2 the corresponding eigenvector. Again, $\lambda = \alpha_2' \sum \alpha_2$, so λ is to be as large as possible. Assuming

that $\sum$ does not have repeated eigenvalues, λ cannot equal to λ_1. If it did, it follows that $\alpha_2 = \alpha_1$, violating the constraint $\alpha_1'\alpha_2 = 0$. Hence, λ is the second largest eigenvalue of $\sum$, and α_2 is the corresponding eigenvector.

As stated above, it can be shown that for the third, fourth,...,p^{th} PC's, the vectors of coefficients $\alpha_3\alpha_4, ...\alpha_p$ are the eigenvectors of $\sum$ corresponding to $\lambda_3\lambda_4, ...\lambda_p$, the third and fourth largest,...., and the smallest eigenvalue respectively. Furthermore,

$$var[\alpha_k'x] = \lambda_k.....k = 1, 2, ...p. \tag{11}$$

It should be noted that sometimes the vectors α_k are referred to as 'principle components'.

3.2. Analytical Chart

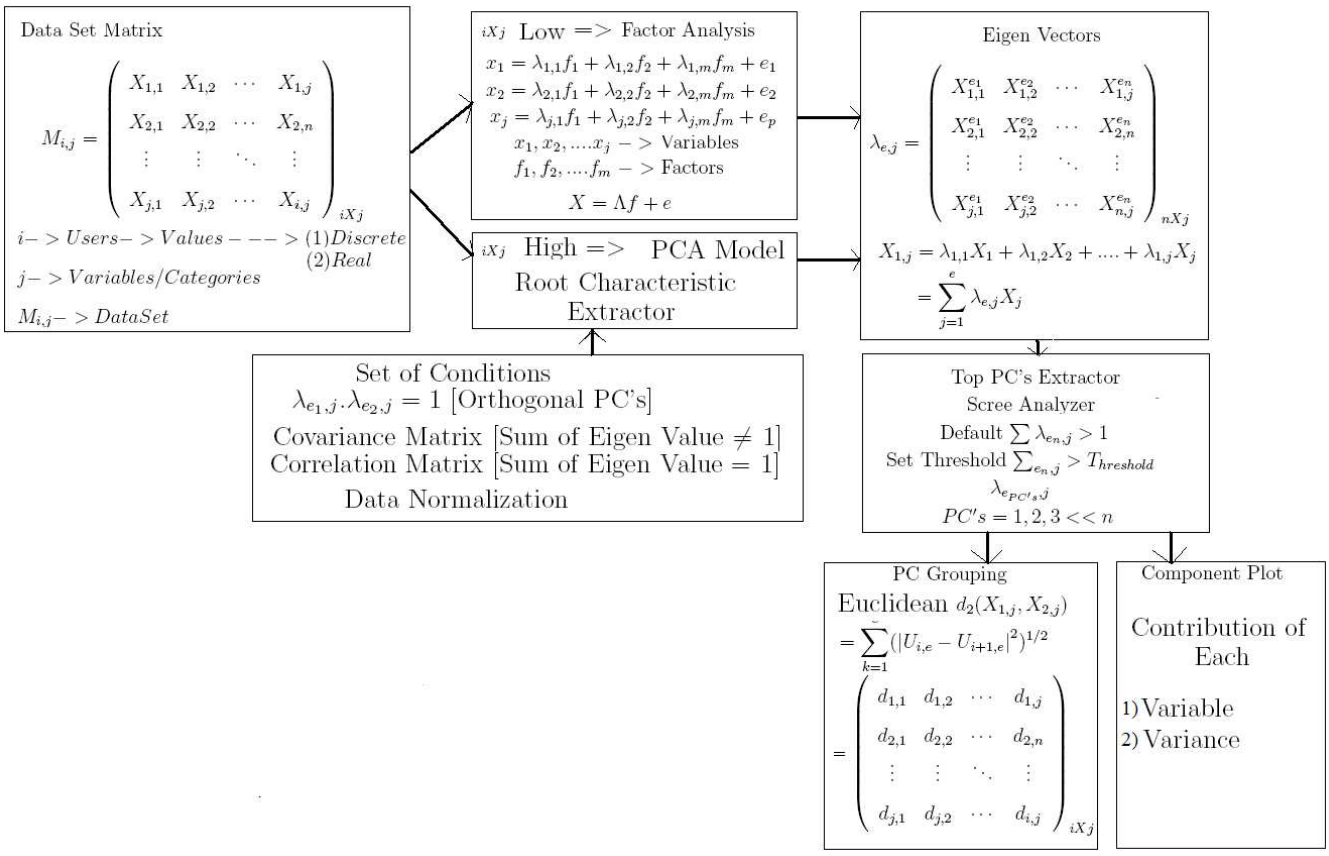

FIGURE 3.1. Analytical work describing contribution of thesis by block diagram

3.2.1. Data Set Matrix

$$M_{i,j} = \begin{pmatrix} X_{1,1} & X_{1,2} & \cdots & X_{1,j} \\ X_{2,1} & X_{2,2} & \cdots & X_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ X_{j,1} & X_{j,2} & \cdots & X_{i,j} \end{pmatrix}_{iXj}$$

$M_{i,j} -> DataSet$

$i -> Users -> Values - - - > (1)Discrete(2)Real$

$j -> Variables/Categories$

I want to

- Represent the relationship among the groups/values
- Represent any structure
- Groups
- Behavioral Patterns
- Dependencies between Each other

Set of Conditions

- $\lambda_{e_1,j}.\lambda_{e_2,j} = 1$[Orthogonal PC's]
- Covariance Matrix [Sum of Eigenvalue $\neq 1$] for spammers data set, as grouping is the focus.
- Correlation Matrix [Sum of Eigenvalue = 1] for Call Logs, as focus is on Extraction of few PC's and variable with maximum importance
- Data Normalization

Depending upon the datasets we have different applications of PC Analysis. There are different set of conditions based on the data set and which I want to inherit from it. For spammers data matrix there is less ambiguity compared to call log analysis.

3.2.2. Analyzer

If iXj is low then it undergoes factor analysis as I need to derive the factors with most contribution. Whereas, if iXj is high then Principal Component Analysis is applied on the Dataset. Analyzer is a critical stage as choice of different analysis will lead to ambiguous results. Both forms of analysis come under the section of factor analysis though it has some variations. After initial calculations it has similar stages later on.

3.2.3. Factor Analysis

The basic idea underlying factor analysis is that p observed random variables, x, can be expressed, except for an error term, as linear functions of $m(< p)$ hypothetical (random) variables or common factors. That is if $x_1, x_2, ..., x_p$ are the variables and $f_1, f_2, ...f_m$ are the factors, then

$$x_1 = \lambda_{1,1}f_1 + \lambda_{1,2}f_2 + \lambda_{1,m}f_m + e_1 \tag{12}$$

$$x_2 = \lambda_{2,1}f_1 + \lambda_{2,2}f_2 + \lambda_{2,m}f_m + e_2 \tag{13}$$

$$x_j = \lambda_{j,1}f_1 + \lambda_{j,2}f_2 + \lambda_{j,m}f_m + e_p \tag{14}$$

where $\lambda_{jk},\ j = 1, 2, ...p; k = 1, 2, ...m$ are constraints called the Factor Loadings, and $e_j, j = 1, 2, ...p$ are error terms, sometimes called specific factors (because e_j is 'specific' to x_j, whereas the f_k are 'common' to several x_j). Equations above can be rewritten in matrix form, with obvious notation as,

$$X = \Lambda f + e \tag{15}$$

$x_1, x_2, ...x_j -> $ Variables $f_1, f_2, ...f_m -> $ Factors

One contrast between PCA and factor analysis is immediately apparent. Factor analysis attempts to achieve a reduction from p to m dimensions by invoking a model relating

$x_1, x_2, ..., x_p$ to m hypothetical or latent variables. For most practical purposes PCA differs from Factor analysis in having no explicit model [30] [33].

3.2.4. PCA Model Root Characteristic Extractor

The key to the problem is that much of the variability in the data set is not independent. From all variables under consideration I could extract two variables that captured most of the independent variability in the entire data set, a simple binary scatter diagram would reveal most of the information in the data. Accordingly, data reduction is the primary objective to extract a few uncorrelated variables that may capture most of the variability in the data set, while preserving the orthogonality of these new optimal reference axes/variables (i.e., principal components). The 1^{st} principal component captures the maximum variation in the data set. The 2^{nd} principal component has the next most variation, and so on.

- The coefficients of these new optimal reference axes are called loadings, and the projections of the original data onto these axes are called scores.
- In a standard principal component analysis, the new reference represents the eigenvectors of the covariance matrix of the data by default. However, you can use the correlation matrix instead.
- If you choose to standardize the data prior to the analysis, the new reference will represent the eigenvectors of the correlation matrix of the data.

For more information regarding PCA Refer Section 3.

3.2.5. Eigenvectors

Eigenvalues are a special set of scalars associated with a linear system of equations (i.e., a matrix equation) that are sometimes also known as characteristic roots, characteristic values, proper values, or latent roots.

The determination of the eigenvalues and eigenvectors of a system is important in physics and engineering, where it is equivalent to matrix diagonalization. They arise in common applications such as stability analysis, the physics of rotating bodies, and small oscillations of vibrating systems, to name only a few. Each eigenvalue is paired with a corresponding

so-called eigenvector (or, in general, a corresponding right eigenvector and a corresponding left eigenvector; there is no analogous distinction between left and right for eigenvalues).

The decomposition of a square matrix X into eigenvalues and eigenvectors is known in this work as eigen decomposition. The fact that this decomposition is always possible, as long as the matrix consisting of the eigenvectors of X is square, it is known as the eigen decomposition theorem.

After computing the eigenvalues the eigen matrix will look as follows,

$$\lambda_{e,j} = \begin{pmatrix} X_{1,1}^{e_1} & X_{1,2}^{e_2} & \cdots & X_{1,j}^{e_n} \\ X_{2,1}^{e_1} & X_{2,2}^{e_2} & \cdots & X_{2,n}^{e_n} \\ \vdots & \vdots & \ddots & \vdots \\ X_{j,1}^{e_1} & X_{j,2}^{e_2} & \cdots & X_{n,j}^{e_n} \end{pmatrix}_{nXj}$$

After computing the eigenvalues, the fundamental PC's are as follows,

$$X_{1,j} = \lambda_{1,1}X_1 + \lambda_{1,2}X_2 + ... + \lambda_{1,j}X_j \tag{16}$$

$$= \sum_{j=1}^{e} \lambda_{e,j}X_j \tag{17}$$

3.2.6. Top PC Extractor

A scree Plot is a simple line segment plot that shows the fraction of total variance in the data as explained or represented by each PC. The PCs are ordered, and by definition are therefore assigned a number label, by decreasing order of contribution to total variance. The PC with the largest fraction contribution is labeled with the label name from the preferences file. Such a plot when read left-to-right across the abscissa can often show a clear separation in fraction of total variance where the 'most important' components cease and the 'least important' components begin. The point of separation is often called the 'elbow'.

One rule is to consider only those with eigenvalues over 1.

Default $\sum \lambda_{e_n,j} > 1$

Another rule of thumb is to plot all the eigenvalues in their decreasing order.

Set Threshold $\sum_{e_n,j} > T_{hreshold}\ \lambda_{e_{PC's},j}\ PC's = 1,2,3 << n$

The plot looks like the side of a mountain, and "scree" refers to the debris fallen from a mountain and lying at its base. So the scree test proposes to stop analysis at the point the mountain ends and the debris (error) begins.

This is the final step of PCA, and is also the easiest. Once I have chosen the components (eigenvectors) that I wish to keep in the data and formed a feature vector, I simply take the transpose of the vector and multiply it on the left of the original data set, transposed.

$FinalData = RowFeatureVectorXRowDataAdjust$

where $RowFeatureVector$ is the matrix with the eigenvectors in the columns transposed so that the eigenvectors are now in the rows, with the most significant eigenvector at the top, and $RowDataAdjust$ is the mean-adjusted data transposed, that is the data items are in each column, with each row holding a separate dimension.

It will give us the original data solely in terms of the vectors I chose. The original data set had two axes, x and y , so the data was in terms of them. It is possible to express data in terms of any two axes that you like. If these axes are perpendicular, then the expression is the most efficient. This was why it was important that eigenvectors are always perpendicular to each other. I have changed the data from being in terms of the axes x and y , and now they are in terms of 2 eigenvectors. In the case of when the new data set has reduced dimensionality. I have left some of the eigenvectors out, the new data is only in terms of the vectors that I decide to keep.

3.2.7. PC Grouping

The majority of cluster analysis techniques require a measure of similarity or dissimilarity between each pair of observations, and PCs have been used quite extensively in the computation of one type of dissimilarity. If the p variables that are measured for each observation are quantitative and in similar units, then an obvious measure of dissimilarity between two observations is the Euclidean distance between the observation in the p dimensional space defined by the variables.

Suppose that a PCA is done based on the covariance or correlation matrix, and that $m(< p)$ PCs account for the most of the variation in x. A possible alternative dissimilarity measure is the Euclidean distance between a pair of observations in the m-dimensional subspace defined by the first m PCs. Similarly, the distance calculated from all p PCs for the correlation matrix is the same as that calculated from the p standardized variables. Using m instead of p PCs simply provides an approximation to the original Euclidean distance, and it is calculated as follows,

$$d_2(X_{1,j}, X_{2,j}) = \sum_{k=1}^{e} (|U_{i,e} - U_{i+1,e}|^2)^{1/2} \tag{18}$$

The distance matrix is represented as ,

$$= \begin{pmatrix} d_{1,1} & d_{1,2} & \cdots & d_{1,j} \\ d_{2,1} & d_{2,2} & \cdots & d_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ d_{j,1} & d_{j,2} & \cdots & d_{i,j} \end{pmatrix}_{iXj}$$

3.2.8. Component Plot

In the scatter plot for factor loadings, each variable is represented as a point. In this plot I could rotate the axes in any direction without changing the relative locations of the points to each other; however, the actual coordinates of the points, that is, the factor loadings would of course change. In this example, if you produce the plot it will be evident that if I rotate the axes by about 45 degrees we might attain a clear pattern of loadings identifying the work satisfaction items and the home satisfaction items.

There are various rotational strategies that have been proposed. The goal of all of these strategies is to obtain a clear pattern of loadings, that is, factors that are somehow clearly marked by high loadings for some variables and low loadings for others. This general pattern is also sometimes referred to as simple structure. Typical rotational strategies are varimax, quartimax, and equamax.

CHAPTER 4

GROUPING ALGORITHMS

The chapter presents an overview of pattern clustering methods from a statistical pattern recognition perspective, with a goal of providing useful advice and references to fundamental concepts accessible to the broad community of clustering practitioners. I present a taxonomy of clustering techniques, and identify cross-cutting themes and recent advances [34]. It is important to understand the difference between clustering (un-supervised classification) and discriminant analysis (supervised classification). In supervised classification, we are provided with a collection of labeled (pre-classified) patterns; the problem is to label a newly encountered, yet unlabeled, pattern.

Clustering is the un-supervised classification of patterns (observations, data items, or feature vectors) into groups (clusters). The clustering problem has been addressed in many contexts and by researchers in many disciplines; this reflects its broad appeal and usefulness as one of the steps in exploratory data analysis. However, clustering is a difficult problem combinatorially, and differences in assumptions and contexts in different communities has made the transfer of useful generic concepts and methodologies slow to occur.

Typically, the given labeled (training) patterns are used to learn the descriptions of classes which in turn are used to label a new pattern. In the case of clustering, the problem is to group a given collection of unlabeled patterns into meaningful clusters. In a sense, labels are associated with clusters also, but these category labels are data driven; that is, they are obtained solely from the data.

Clustering is useful in several exploratory pattern-analysis, grouping, decision-making, and machine-learning situations, including data mining, document retrieval, image segmentation, and pattern classification. However, in many such problems, there is little prior information (e.g., statistical models) available about the data, and the decision-maker must

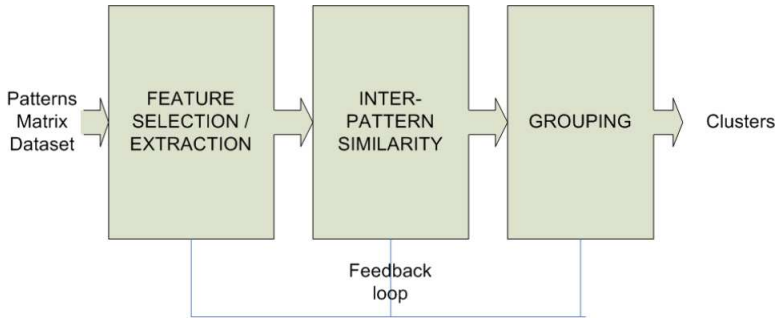

FIGURE 4.1. Components of a clustering task

make as few assumptions about the data as possible. It is under these restrictions that clustering methodology is particularly appropriate for the exploration of interrelationships among the data points to make an assessment (perhaps preliminary) of their structure.

The term "clustering" is used in several research communities to describe methods for grouping of unlabeled data. These communities have different terminologies and assumptions for the components of the clustering process and the contexts in which clustering is used.

4.1. Components of a Clustering Task

Typical pattern clustering activity involves the following steps [35]:

(i) Pattern representation (optionally including feature extraction and/or selection),

(ii) Definition of a pattern proximity measure appropriate to the data domain

(iii) Clustering or grouping

(iv) Data abstraction (if needed)

(v) Assessment of output (if needed)

Pattern representation refers to the number of classes, the number of available patterns, and the number, type, and scale of the features available to the clustering algorithm. Some of this information may not be controllable by the practitioner. Feature selection is the process of identifying the most effective subset of the original features to use in clustering. Feature extraction is the use of one or more transformations of the input features to produce new salient features. Either or both of these techniques can be used to obtain an appropriate set of features to use in clustering. Pattern proximity is usually measured by a distance

function defined on pairs of patterns. A variety of distance measures are in use in the various communities [36] [35] [37]. A simple distance measure like Euclidean distance can often be used to reflect dissimilarity between two patterns; whereas, other similarity measures can be used to characterize the conceptual similarity between patterns. Distance measures are discussed in Section 4.1.1.

The grouping step can be performed in a number of ways. The output clustering (or clusterings) can be hard (a partition of the data into groups) or fuzzy (where each pattern has a variable degree of membership in each of the output clusters). Hierarchical clustering algorithms produce a nested series of partitions based on a criterion for merging or splitting clusters based on similarity. Partitional clustering algorithms identify the partition that optimizes (usually locally) a clustering criterion. Additional techniques for the grouping operation include probabilistic [38] and graph-theoretic [39] clustering methods. The variety of techniques for cluster formation is described in Section 4.1.2.

Data abstraction is the process of extracting a simple and compact representation of a data set. Here, simplicity is either from the perspective of automatic analysis (so that a machine can perform further processing efficiently) or it is human-oriented (so that the representation obtained is easy to comprehend and intuitively appealing). In the clustering context, a typical data abstraction is a compact description of each cluster, usually in terms of cluster prototypes or representative patterns such as the centroid [37].

How is the output of a clustering algorithm evaluated? What characterizes a 'good' clustering result and a 'poor' one? All clustering algorithms will, when presented with data, produce clusters regardless of whether the data contain clusters or not. If the data does contain clusters, some clustering algorithms may obtain 'better' clusters than others. The assessment of a clustering procedure's output, then, has several facets. One is actually an assessment of the data domain rather than the clustering algorithm itself. Data which do not contain clusters should not be processed by a clustering algorithm.

4.1.1. Metrics Used: Similarity Measures or Dissimilarity Measures

Since similarity is fundamental to the definition of a cluster, a measure of the similarity between two patterns drawn from the same feature space is essential to most clustering procedures. Because of the variety of feature types and scales, the distance measure (or measures) must be chosen carefully. It is most common to calculate the dissimilarity between two patterns using a distance measure defined on the feature space. I will focus on the well-known distance measures used for patterns whose features are all continuous. The most popular metric for continuous features is the euclidean distance

$$d_2(X_i, X_j) = (\sum_{k=1}^{d}(x_{i,k} - x_{j,k})^2)^{1/2} = ||X_i - X_j||_2, \tag{19}$$

which is a special case ($p = 2$) of the minkowski metric

$$d_p(X_i, X_j) = (\sum_{k=1}^{d}|x_{i,k} - x_{j,k}|^p)^{1/p} = ||X_i - X_j||_p, \tag{20}$$

The euclidean distance has an intuitive appeal as it is commonly used to evaluate the proximity of objects in two or three-dimensional space. It works well when a data set has "compact" or "isolated" clusters [40]. The drawback to direct use of the minkowski metrics is the tendency of the largest-scaled feature to dominate the others. Solutions to this problem include normalization of the continuous features (to a common range or variance) or other weighting schemes. Linear correlation among features can also distort distance measures; this distortion can be alleviated by applying a whitening transformation to the data or by using the squared mahalanobis distance

$$d_M(X_i, X_j) = (X_i - X_j)\sum^{-1}(X_i - X_j)^T, \tag{21}$$

where the patterns x_i and x_j are assumed to be row vectors, and $\sum$ is the sample covariance matrix of the patterns or the known covariance matrix of the pattern generation process; $d_M(.,.)$ assigns different weights to different features based on their variances and pairwise linear correlations. Some clustering algorithms work on a matrix of proximity values instead

of on the original pattern set. It is useful in such situations to pre-compute all the $n(n-1)/2$ pairwise distance values for the n patterns and store them in a (symmetric) matrix.

4.1.2. Hierarchical Algorithms

The operation of a hierarchical clustering algorithm is illustrated using the two-dimensional data set in figure 4.2. This figure depicts seven patterns labeled A, B, C, D, E, F, and G in three clusters. A hierarchical algorithm yields a dendogram representing the nested grouping of patterns and similarity levels at which groupings change. A dendogram corresponding to the seven points in figure 4.2(obtained from the single-link algorithm [35]) is shown in figure 4.3. The dendogram can be broken at different levels to yield different clusterings of the data.

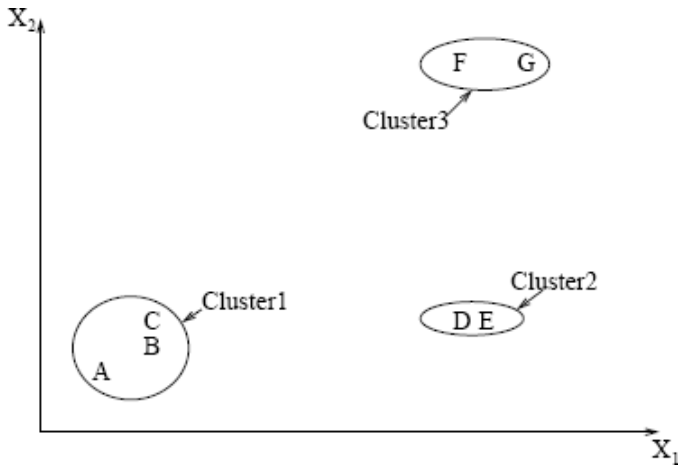

FIGURE 4.2. Points falling in three clusters

Most hierarchical clustering algorithms are variants of the single-link [41], complete-link [42], and minimum-variance [43] [44] algorithms. Of these, the single-link and complete link algorithms are most popular. These two algorithms differ in the way they characterize the similarity between a pair of clusters. In the single-link method, the distance between two clusters is the minimum of the distances between all pairs of patterns drawn from the two clusters (one pattern from the first cluster, the other from the second). In the complete-link algorithm, the distance between two clusters is the maximum of all pairwise distances between patterns in the two clusters. In either case, two clusters are merged to form a

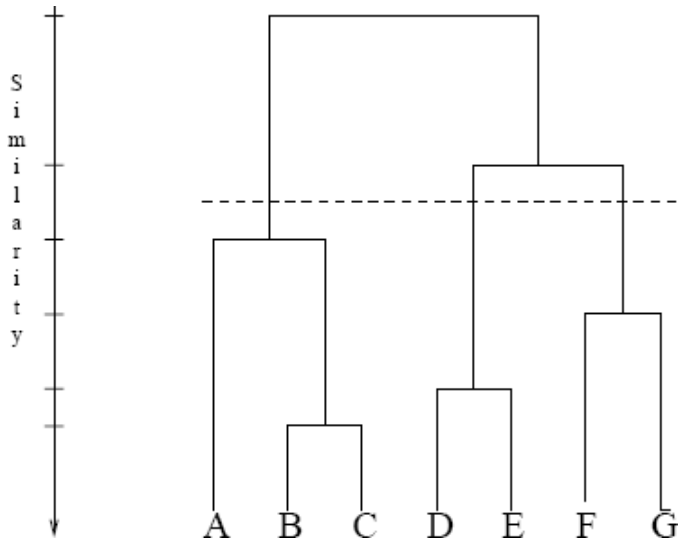

FIGURE 4.3. The dendogram obtained using the single-link algorithm

larger cluster based on minimum distance criteria. The complete-link algorithm produces tightly bound or compact clusters [45]. The single-link algorithm, by contrast, suffers from a chaining effect [46]. It has a tendency to produce clusters that are straggly or elongated.

4.1.3. Agglomerative Single-Link Slustering algorithm

(i) Place each pattern in its own cluster. Construct a list of inter-pattern distances for all distinct unordered pairs of patterns, and sort this list in ascending order.

(ii) Step through the sorted list of distances, forming for each distinct dissimilarity value d_k a graph on the patterns where pairs of patterns closer than d_k are connected by a graph edge. If all the patterns are members of a connected graph, stop. Otherwise, repeat this step.

(iii) The output of the algorithm is a nested hierarchy of graphs which can be cut at a desired dissimilarity level forming a partition (clustering) identified by simply connected components in the corresponding graph.

4.1.4. Agglomerative Complete-Link Clustering algorithm

(i) Place each pattern in its own cluster. Construct a list of inter-pattern distances for all distinct unordered pairs of patterns, and sort this list in ascending order.

(ii) Step through the sorted list of distances, forming for each distinct dissimilarity value d_k a graph on the patterns where pairs of patterns closer than d_k are connected by a graph edge. If all the patterns are members of a completely connected graph, stop.

(iii) The output of the algorithm is a nested hierarchy of graphs which can be cut at a desired dissimilarity level forming a partition (clustering) identified by completely connected components in the corresponding graph.

Hierarchical algorithms are more versatile than partitional algorithms. For example, the single-link clustering algorithm works well on data sets containing non-isotropic clusters including well-separated, chain-like, and concentric clusters, whereas a typical partitional algorithm such as the k-means algorithm works well only on data sets having isotropic clusters [46]. On the other hand, the time and space complexities [47] of the partitional algorithms are typically lower than those of the hierarchical algorithms. It is possible to develop hybrid algorithms [48] that exploit the good features of both categories.

4.1.5. Hierarchical Agglomerative Clustering algorithm

(i) Compute the proximity matrix containing the distance between each pair of patterns. Treat each pattern as a cluster.

(ii) Find the most similar pair of clusters using the proximity matrix. Merge these two clusters into one cluster. Update the proximity matrix to reflect this merge operation.

(iii) If all patterns are in one cluster, stop. Otherwise, go to step 2.

Based on the way the proximity matrix is updated in step (ii), a variety of agglomerative algorithms can be designed. Hierarchical divisive algorithms start with a single cluster of all the given objects and keep splitting the clusters based on some criterion to obtain a partition of singleton clusters.

4.1.6. K-Means Clustering Algorithm

(i) Choose k cluster centers to coincide with k randomly-chosen patterns or k randomly defined points inside the hyper-volume containing the pattern set.

(ii) Assign each pattern to the closest cluster center.

(iii) Recompute the cluster centers using the current cluster memberships.

(iv) If a convergence criterion is not met, go to step 2. Typical convergence criteria are: no (or minimal) reassignment of patterns to new cluster centers, or minimal decrease in squared error.

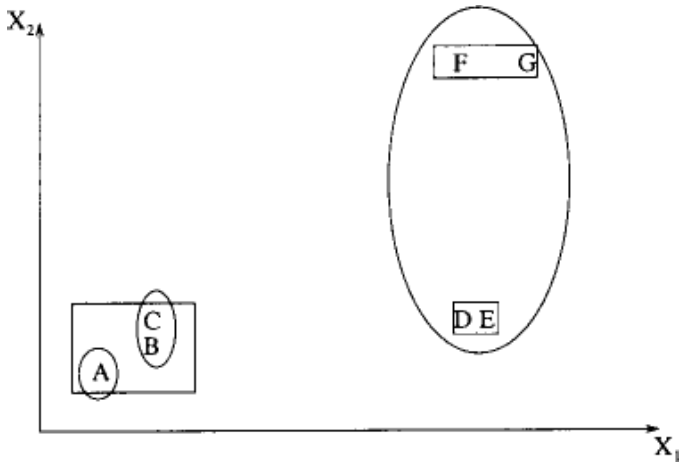

FIGURE 4.4. The K-means algorithm is sensitive to the initial partition

Several variants [36] of the k-means algorithm have been reported in the literature. Some of them attempt to select a good initial partition so that the algorithm is more likely to find the global minimum value. Another variation is to permit splitting and merging of the resulting clusters. Typically, a cluster is split when its variance is above a pre-specified threshold, and two clusters are merged when the distance between their centroids is below another pre-specified threshold. Using this variant, it is possible to obtain the optimal partition starting from any arbitrary initial partition, provided proper threshold values are specified. The well-known ISODATA [49] algorithm employs this technique of merging and splitting clusters. If ISODATA is given the ellipse partitioning shown in figure 4.4 as an initial partitioning, it will produce the optimal three-cluster partitioning. ISODATA will

first merge the clusters A and B, C into one cluster because the distance between their centroids is small and then split the cluster D, E, F, G, which has a large variance, into two clusters D, E and F, G.

Another variation of the k-means algorithm involves selecting a different criterion function altogether. The dynamic clustering algorithm (which permits representations other than the centroid for each cluster) describes a dynamic clustering approach obtained by formulating the clustering problem in the framework of maximum-likelihood estimation. The regularized mahalanobis distance was used in [40] to obtain hyperellipsoidal clusters.

CHAPTER 5

INFORMATION THEORY: ENTROPY

5.1. Introduction to Entropy Analysis

In information theory, the Shannon entropy or information entropy is a measure of the uncertainty associated with a random variable. It quantifies the information contained in a message, usually in bits or bits/symbol. It is the minimum message length necessary to communicate information.

Equivalently, the Shannon entropy is a measure of the average information content the recipient is missing when he does not know the value of the random variable.The concept was introduced by Claude E. Shannon in his 1948 paper "A Mathematical Theory of Communication" [50].

I had nothing to do with making a communication channel lossless or compression was not our focus, I used concepts of entropy to extract the randomness associated to a variable or a person to capture more behavioral information. For our model entropy was a crucial element as I could derive and verify certain results, which gave us a technical backing to the work I did using this analysis. For example one such application is to determine how random is a particular botnet based on its spamming patterns, so that one can say how predictable structure a particular spam bot is, and how difficult it is to trace one.

5.1.1. Definition

The information entropy of a discrete random variable X, that can take on possible values $x_1...x_n$ is

$$H(X) = E(I(X)) = -\sum_{i=1}^{n} p(x_i) log_2 p(x_i) \tag{22}$$

where

$I(X)$ is the information content or self-information of X, which is itself a random variable; and

$p(x_i) = Pr(X = x_i)$ is the probability mass function of X where $p(x_i)$ is the probability mass function of outcome x_i, and b is the base of the logarithm used. Possible values of b are 2, e, and 10. The unit of the information entropy H is bit for $b = 2$, *nat* for $b = e$, *dit* (or digit) for $b = 10$.

5.1.2. Entropy Application for Call Logs

Mobile phone has moved beyond being a mere technological object and has become an integral part of many people's social lives. This has had profound implications on both how people as individuals perceive communication as well as in the patterns of communication of humans as a society. In this thesis I try to capture the behavior of phone users based on their calling patterns and infer trend of behavior dependencies using techniques such as entropy, principal factor analysis and correlation function. I present a new method for precise measurement of randomness of phone user based on their calling patterns such as location of the call, talk time, calling time and interconnected time and infer relationship among them [51].

Recently there has been increasingly growing interests in the field of mobile social networks analysis, but due to the unavailability of data, there have been far fewer studies. The reality mining project at Massachusetts Institute of Technology (MIT) [11] has made publicly available large datasets from their projects. I implement the techniques on the reality mining dataset which was collected over 9 months by monitoring the cell phone usage of 100 participants. The information collected in the call logs includes user IDs (unique number representing a mobile phone user), time of call, call direction (incoming and outgoing), incoming call description (missed, accepted), talk time, and tower IDs (location of phone users). These 100 phone users are students, professors, and staffs. Using purely objective data first time the researchers can get an accurate glimpse into human behaviors. Our interest in this data set is to study the behavior of the phone user using information theory [52], data mining and data reduction techniques.

The main contribution of entropy is to infer the relationship between the randomness levels in behavior of the phone users in a cellular network.

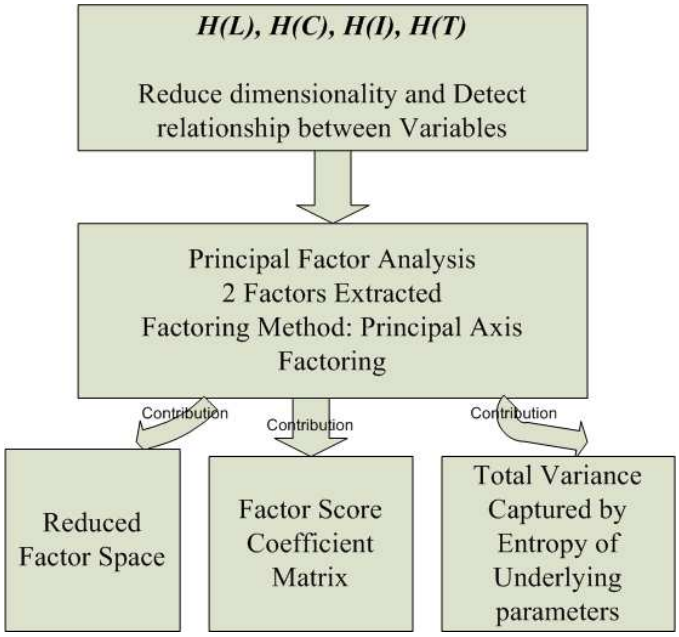

FIGURE 5.1. Block diagram for contribution of entropy using call logs

While individual phone user's calling behavior is random, some users might be more predictable than others. Being more predictable can also mean being less random. To quantify the randomness or amount of predictable structure in an individual calling pattern, the information entropy can be used. Next, I perform factor analysis in order to further study the relationship of the randomness levels (entropy) based on the underlying parameters.

The main application of factor analysis is: (1) to reduce the number of variables and (2) to detect structure in the relationship between variables, that is to classify variables. In the analysis I use it for both the purposes. The flow diagram of the principal factor analysis is shown in figure 5.3. Factor analysis is generally used to encompass both principal components and principal factor analysis. The eigenvalue for a given factor measures the variance in all the variables which is accounted by that factor, as stated in table 5.1. If a factor has a low eigenvalue, then it is contributing little to the explanation of variances in the variables and may be ignored as redundant with more important factors. Eigenvalue is not the percent of variance explained but rather a measure of amount of variance in relation

TABLE 5.1. Total variance explained

Factor	Initial Eigenvalues		
	Total	% of Variance	Cumulative %
1	1.598	39.953	39.953
2	1.025	25.616	65.568
3	0.730	18.243	83.811
4	0.648	16.189	100.00

to total variance (since variables are standardized to have means of 0 and 1, total variance is equal to the number of variables).

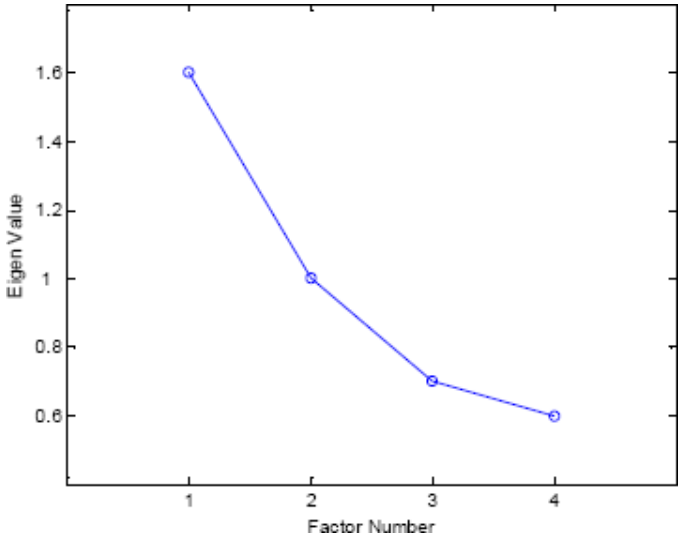

FIGURE 5.2. Scree plot determining number of components extracted for our analysis

Initial eigenvalues and eigenvalues after extraction (extracted sums of squared loadings) are same for principal component analysis (PCA) extraction [30], but for factor analysis eigenvalues after extraction will be lower than their initial counterparts. The plot of the entropy based on four parameters lying on the first and second factor is shown in figure 5.3 where the actual values are given by table 6.1. It can be observed that the entropy based on location and calling time are positively lying on the first factor whereas the entropy based on inter-connected time and talk time are positively lying on the second factor. Since the first and second factor are orthogonal i.e., uncorrelated, one can notice two established relations between; (1)entropy based on location and calling time and (2) entropy based on inter-connected time and talk time.

TABLE 5.2. Factor matrix

Component	Factor	
H(T)	0.548	-0.169
H(I)	0.478	-0.007
H(C)	-0.371	0.519
H(L)	-0.049	0.501

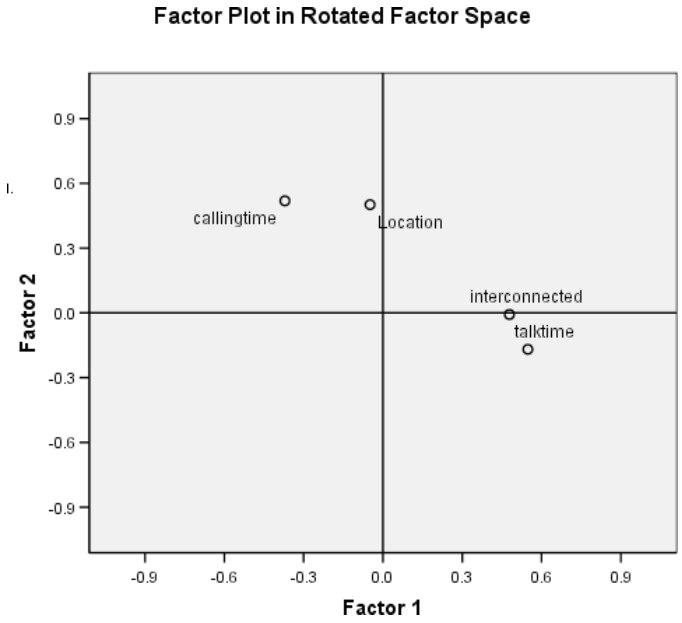

FIGURE 5.3. Factor plot determining relationship between variables $H(T)$, $H(I)$ and $H(C)$, $H(L)$

The result based on the trend lines, correlation coefficients, and factor analysis tells us that there is a high correlation in the randomness in phone user's location and his/her calling time, also high correlation in the randomness in phone user's inter-connected time and his/her talk time. This draws the conclusion of our study that phone users who have high mobility tend to be more variable in time of making calls but less variable in time spent talking on the phone as well as the inter-connected time. By the same token, the phone users who spend high random amount of time talking on the phone tend to not be so much random in mobility compared to those who have more predictable talk time.

5.1.3. Entropy Application for Spammers Behavior Analysis

How easy it is to detect/trace botnet spam machines?

One has to distinguish between normal and malicious activities. In figure 8.2 one can see that after bot receive commands from their controller they respond immediately and accurately so I have a constant response time in terms of Bot structure. Where as a legitimate host receives a message, it responds or performs an action from a wide variety of possibilities, after a variable thinking time. A botnet machine I assume performs a preprogrammed set of activities so therefore I have all the variables of header analysis used as metrics for botnet detection.

To quantify the randomness or amount of predictable structure in an individual botnet group, the information entropy can be used. The information entropy or Shannon's entropy is a measure of uncertainty of a random variable. The information entropy as given in 23 was introduced by Shannon [50].

$$H(X) = -\sum_{x} p(x) log_2 p(x), \tag{23}$$

where X is a discrete random variable, and the probability mass function $p(x) = Pr(X = x)$. The spam botnet pattern can be observed from the active time, time of arrival, frequency and content length of email spam. Let A, T, N and C be random variables representing active time, time of arrival, frequency and content length respectively. The entropy of Active time can be calculated by 24.

$$H(A) = -\sum_{t} p(a) log_2 p(a), \tag{24}$$

where the probability $p(a)$ is a ratio of the number of mails during t^{th} hour slot to the total number of mails of all time slots. The entropy of time of arrival can be calculated by 25.

$$H(T) = -\sum_{t} p(t) log_2 p(t), \tag{25}$$

where the probability $p(t)$ is a ratio of the number of mails during t^{th} hour slot to the total number of mails of all time slots. By the same token, the randomness in the spammers active time content length $H(C)$and frequency $H(N)$, can also be quantified using information entropy which is defined in 23. Further detail explanation is also given in Section 8.1

CHAPTER 6

BEHAVIOR ANALYSIS OF SPAM BOTNETS

First, I developed a feature set for each Spammer, as a data set matrix. I then applied PCA to the feature set to identify features which captures most of the variance in the data set. Further, I clustered these spammers into groups based on their behavior patterns. I considered the possibility that a spammer might spam multiple receivers within the same domain. To examine this I identified common behaviors for a group of spammers by using the proximity between the senders (using a distance metric) and by applying clustering algorithms.

To verify the approach's accuracy, I hand-labeled the data and compared those results to the automatic botnet identification of each corpus. Accuracies around 90% have been achieved. Thus, using the proposed technique, I may effectively block those spammers as groups instead of blocking them individually.

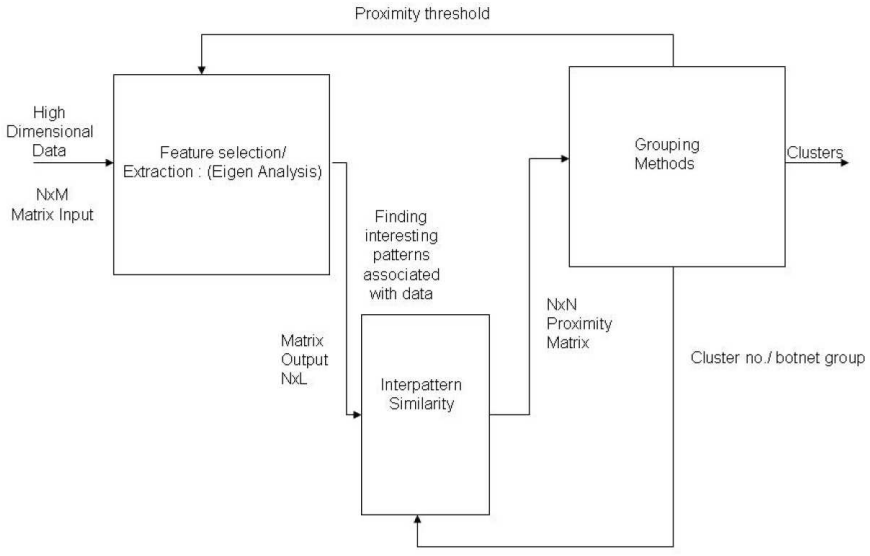

FIGURE 6.1. Architecture

6.1. Proposed Approach

Identifying the behavior patterns of spam botnets is based on three phases which are described next.

Phase I: Feature selection using Principal Component Analysis

The goal of phase I is to reduce the data set and extract only relevant data through the traditional use of eigen analysis. After performing Eigen analysis, I select the feature set which can be used for grouping and which captures maximum variance in the data. The matrix input is NxM; where N is the number of spammers and M is the number of features in the set.

Phase II: Proximity between Senders

This phase is used to find the association pattern between spammers. The association pattern is evaluated using the euclidean distance defined between pairs of senders. Input given to this stage is the NxL matrix from Phase I. The output is a NxN proximity matrix (a dissimilarity matrix); it gives us the proximity relation between each pair of senders. The lesser the value the more closely associated the particular pair of spammers are.

Phase III: Grouping Methods

In phase III, we want to cluster spammers who exhibit similar patterns of spamming behavior. Now, using the proximity matrix generated in phase II, we group spammers with similar proximity values into one cluster, using algorithms such as Hierarchical and K-means clustering. The output at this stage gives us clusters of spammers. Thus, based on the closeness of each sender, we have groupings of compromised computers, i.e., bots for each spammer.

6.2. Background

The approach proposed here relies on two existing methods. The first, principal component analysis, is used to extract the components that have a higher impact for a given data set. The second method, clustering, enables classifying individuals into groups evincing similar patterns. A brief description of both techniques is given next.

6.2.1. Principal Component Analysis

Typically, [30] [57] PCA is used to reduce the dimensionality of a data set consisting of a large number of interrelated variables while retaining as much as possible of the variation present in the data set. This is achieved by transforming to a new set of variables, the principal components (PCs), which are uncorrelated, and ordered so that the first few retain most of the variation present in the original variables.

Suppose that X is a vector of p random variables, and we want to infer about the variances of the p random variables and the structure of the covariances or correlations between the p variables. One observes the p variances and $\left\{\frac{1}{2} \times p(p-1)\right\}$ correlations or covariances, which increase in complexity with the increase in size of X. An alternative is to observe for a few derived variables ($<< p$) [30] that preserve most of the information given by these variances and correlations or covariances.

6.2.1.1. *Choosing a Subset of Principal Components.* In this study we use the scree test, developed by Cattell [58] [33], to decide how many PCs should be retained to account for most of the variation in X.

Principal components are successively chosen to have the largest possible variance [30]. Suppose the variance of the k^{th} PC is l_k, scree test involves looking at a plot of l_k against k and deciding at which value of k the slopes of lines joining the plotted points are 'steep' to the left of k, and 'not steep' to the right. This value of k, (defining an 'elbow' in the graph), is taken to be the number of components m to be retained.

6.2.2. Hierarchical Clustering

Hierarchical algorithms [59] can be agglomerative ("bottom-up") or divisive ("top-down"). Agglomerative algorithms begin with each element as a separate cluster and merge them into successively larger clusters. Divisive algorithms begin with a whole set and divide the set into successively smaller clusters.

We have used an agglomerative algorithm for clustering. This method builds the hierarchy from the individual elements by progressively merging clusters. In this approach,

clustering begins with each element as a separate cluster. We continue grouping based on correlation until we have formed one cluster.

6.2.3. K-Means Clustering

K-Means clustering provides a simple procedure to classify a data set through a certain number of fixed clusters(assume k clusters). The idea is to define k centroids, one for each cluster. These centroids must be placed carefully because location will affect the results. A better choice is to place the centroids as far as possible from each other. Then, we can take each point belonging to a given data set and associate it with its nearest centroid.

6.3. Eigen-Behavior of Spammers

Typically, incoming emails consist of emails from numerous senders. For instance, emails may originate from telemarketers, fraudsters, family, friends and opt-in senders. As the percentage of these unsolicited emails increases in the incoming email traffic, annoyance or nuisance increases, resulting in a loss of productivity. To verify this, we categorized the incoming email traffic collected at an enterprise's mailserver by asking the recipients to hand-label their emails. We have examined a corpus of emails, all from spammers. Based on the spammers' locations, we categorized the traffic profile of the botnet groups. Identifying the spammers' physical locations cannot be achieved using the originating location of the spam as spammers use compromised machines (bots). Often the spammer is physically located elsewhere. Here we define spammers-feature matrix and group-feature matrix (mix of spam & legitimate emails), over which we performed PCA to identify the association patterns among spammers. Spammers-feature matrix and group-feature matrix are m by n matrices, where m is the total number of senders and n is the number of considered features.

$$Spammers - feature = \begin{pmatrix} e^1_{11} & e^1_{1,2} & \cdots & e^1_{1n} \\ e^1_{21} & e^1_{2,2} & \cdots & e^1_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ e^1_{m1} & e^1_{m2} & \cdots & e^1_{mn} \end{pmatrix}$$

$$group - feature = \begin{pmatrix} e^2_{11} & e^2_{1,2} & \cdots & e^2_{1n} \\ e^2_{21} & e^2_{2,2} & \cdots & e^2_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ e^2_{m1} & e^2_{m2} & \cdots & e^2_{mn} \end{pmatrix}$$

In spammers-feature matrix m, corresponds to the total number of spammers whereas, in group-feature matrix m is a mix of spammers and legitimate senders. The values e^1_{ij} and e^2_{ij} for each entry in the column vector are measurements corresponding to parameters, listed below, extracted from the header of the e-mails.

- Time of arrival
- Inter-arrival Time
- Active Time
- Content length
- Frequency
- Content type

Typical header format of an email is as follows:

X-Originating-IP: [222.33.169.23]

From: HomeDepot GiftCenter¡Patiencesztxehfcswgpk

Date: Mon, 23 Jul 2007 08:40:28 +0-800

Content-Length: 839

X-Originating-IP: [207.210.120.235]

From: “Lingerie Order: 20060910B7”

Date: Sun, 22 Jul 2007 12:59:25 +0000

Content-Length: 1004

X-Originating-IP: [207.210.120.230]

Received: from 207.210.120.230 (HELO ecardnotice.com) (207.210.120.230)

Content-Type: text/html; charset="UTF-8"

From: "TJ Maxx Certificate"

MIME-Version: 1.0

To: "XXXXX"

Date: Sun, 22 Jul 2007 04:54:01 +0000

Content-Length: 898

The above features have been selected based on PCA. We can see at this point that I do not need to use all the eigenvectors. We represent the data in terms of vectors where the eigenvalues are higher than a threshold (pre-specified at the time of analysis). In the following sections we will discuss the strategies used in choosing a subset of principal components and get a consistent representation of the underlying patterns.

6.3.1. Components Selection

As a first step in the analysis, we decide how many principal components to retain. This helps identify the predominant features common between the spammers spamming a specific recipient. To achieve this, we retain only components with eigenvalues above 1.0 [57, 30]. That is, we drop any component that accounts for less variance than does a single variable. We used scree test [58] to determine the most significant eigenvectors, those account for most of variation.

Figure 6.2 is a scree plot obtained by performing PCA on one of the corpus. It can be observed that four principal components (active time, time of arrival, frequency, and content length) account for most of the email corpus's variation. Therefore, I retained four components for further analysis. The plot also provides a visual aid for deciding at what point including the additional features no longer increases the amount of variance accounted for by a non-trivial amount.

For the corpus-I feature set, the first four components have eigenvalues greater than 1.0. Component 1:active time, component 2:content length, component 3:frequency cumulatively

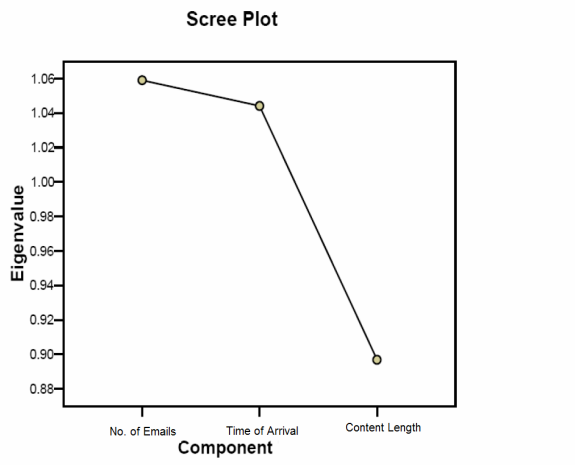

FIGURE 6.2. Scree plot for corpus

TABLE 6.1. Component matrix

Component	eigen Value	% of Variance	cumulative %
1	1.798	29.971	29.971
2	1.094	18.231	48.202
3	1.014	16.906	65.108
4	0.994	16.572	81.634
5	0.897	14.954	96.634
6	0.202	3.366	100.00

account for 65% of the variance whereas the inclusion of time of arrival increases the cumulative variance to 81%. The features active time, content length, frequency and time of arrival account for the maximum variation.

6.3.2. Component Plot in Rotated Space

Another matrix of interest is the component matrix, also known as the feature pattern matrix. Loadings, the entries in this component matrix, are correlations between the components and the variables (in our case the features) between various parameters.

Each principal component represents an orthogonal dimension. I retained three dimensions, so that we can plot them on a 3-D plane. But, later in our study we also retain more than three components so that we can examine at several pairwise plots.

I rotate these axes so that the three dimensions passed more nearly through the major clusters. By rotating them, (preserving their perpendicularity), one axis passes through or

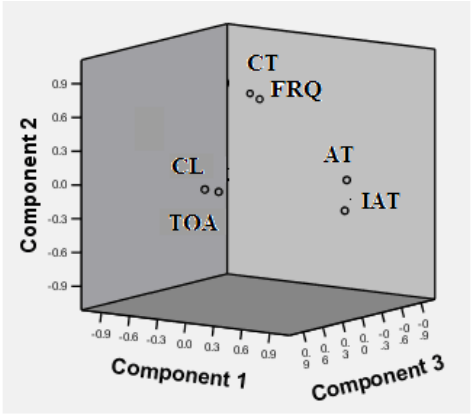

FIGURE 6.3. Component plot in rotated space

near the one cluster, the other through or near the other cluster. Table 6.1 is the loading matrix after rotation and figure 6.3 gives the component plot in rotated space.

One can see from figure 6.3 that different variables load well on different components, Active time shows its highest positive loadings towards the first component and so does Inter-arrival time. Thus component 1 has a strong affinity towards these two parameters. Similarly the next two parameters, content length and time are positively loaded on component 3. Using this qualitative approach, we provide quantitative based result, where based on two parameters grouping was obtained.

6.3.3. Eigen-clustering of spammers

Once we obtained the principal components, we identified clusters of spammers sharing similar spamming patterns. This is done by rotating the components and plotting the points in a 3-D plane. Figure 6.4 displays the corpus-I subjects in a 3-D plane formed by rotating the first three eigen-vectors.

In the figure 6.4 one can notice the cluster of spammers well separated from the clusters of legitimate senders. We further studied the cluster of spammers and classified these spammers based on the similarities in their association patterns. Figure 6.5 displays the various sub-clusters of spammers having similar association patterns. Overall results obtained from this

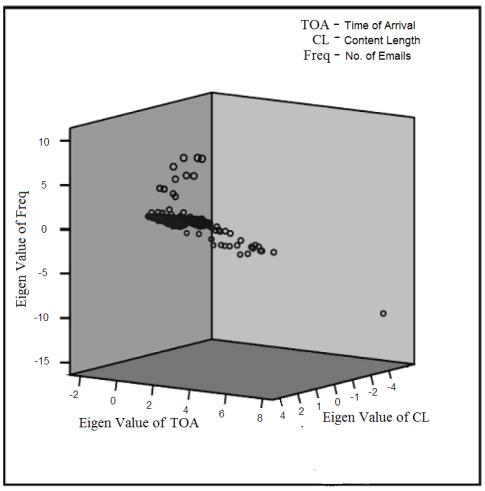

FIGURE 6.4. rns

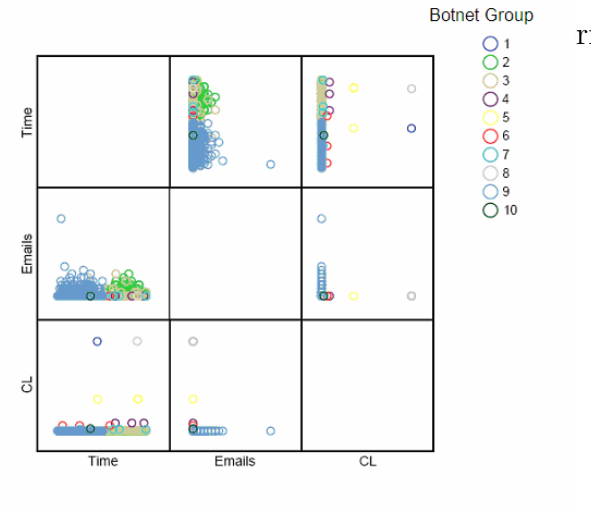

FIGURE 6.5. Cluster of spammers in corpus-I based on their association patterns

analytical study were satisfying. We were able to cluster the subjects in corpus-I with a precision of 91.86% with few false positives and false negatives.

6.3.4. Clustering

I performed K-means [60, 35, 61, 62] clustering to corpus 1. As a result, we clustered the spammers into three prominent clusters, (Table 6.2.) We also found 18 missing cases while performing K-means. Missing data values [63] can occur for 2 reasons: either a measurement is made and then lost or a measurement cannot be made at all. I signify missing values by using the symbol "N" in the data matrix. In the analysis, missing data is just 1.7% of the

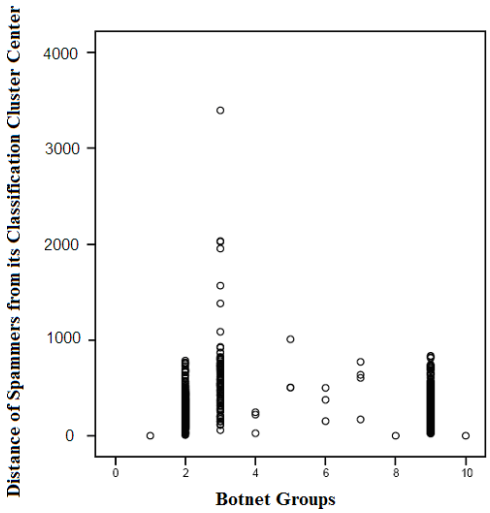

FIGURE 6.6. Results of the application of the K-means clustering algorithm for K=10

total matrix values, so it does not have impact on the clustering method. When I standardize the matrix, I act as if the N's are not present. That is, skip over them.

TABLE 6.2. Number of cases in the clusters

Cluster	Number of Spammers
1	1
2	412
3	96
4	3
5	3
6	3
7	4
8	1
9	498
10	1
Valid	1022
Missing	18

Figure 6.6 shows the number of clusters on the X axis. The Y axis shows the distance of spammers from their classification cluster's center. Spammers having their centroids close to a classification cluster are classified as a group (represented by a vertical line of spammers who are closely associated in figure 6.6). Thus we can say that the spammers having similar features will have their centroids exactly or close to the classification cluster.

We also have few outliers present in K-means clustering; These senders (outliers) have a pattern of spamming which does not fall into any of the clusters. Hence, they comprise a separate cluster, since it's an iterative process any new spammers will be associated with their corresponding clusters.

After performing K-means clustering we further analyzed the results and categorize individuals in each cluster based on similar time of arrival, content length and frequency. In addition, these individuals also shared other similarities such as possessing the same network ID and having a close (or the same) geographic location. Thus, I identified botnets which were present in each cluster. I also discuss the accuracy of these identifications in Sec. 6.6.

We have represented the association patterns based on the feature set, consider figure 6.5 showing categories of spammers having similar: time of arrival & content length, time of arrival & number of emails, content length & number of emails. Since we have set k=10 we will have 10 different cases which belong to one of the block having atleast 2 similar feature in the feature set.

6.3.5. Average Linkage Clustering

The average linkage clustering is a method of calculating distance between clusters in hierarchical cluster analysis. The linkage function specifying the distance between two clusters is computed as the average distance between spammer from the first cluster and spammer from the second cluster. The averaging is performed over all pairs of spammers, where one of the pair is an object from the first cluster, and the other is an object from the second cluster. The dissimilarity between clusters is calculated using cluster average values; of course there are many ways of calculating an average.

6.3.6. Agglomeration Schedule

An agglomerative, hierarchical classification starts of with each element as a separate cluster and goes on grouping based on the correlation until it forms one whole cluster. This method builds the hierarchy from the individual elements by progressively merging clusters.

Stage	Cluster # Combined		Stage Cluster First Appears		Next Stage	Error Coefficients
	Cluster 1	Cluster 2	Previous Stage for Cluster 1	Previous Stage for Cluster 2		
1	995	996	0	0	473	0.000
473	830	995	0	1	635	0.000
635	85	830	348	473	762	0.000
762	85	90	635	551	818	0.001
818	85	606	762	559	896	0.003
896	85	306	818	822	925	0.008
925	85	86	896	849	944	0.018
944	85	94	925	911	969	0.39
969	85	818	944	0	979	0.151
979	85	87	969	958	1002	0.264

FIGURE 6.7. Agglomeration schedule iteration

Figure 6.7 shows how the agglomeration algorithm groups them into clusters: That is at each stage which 2 senders are merged. The table also shows the previous stage the two cluster's which are currently being merged,appeared in their preceding stages. The next stage column indicates the next particular stage the agglomeration takes place. The agglomeration schedule shows the amount of error created at each clustering stage when two different objects (spammers) are brought together to create a new cluster. A large jump in value of the error term indicates the two different things have been brought together and there is a significant typology at that level of fusion.

6.3.6.1. *Hierarchical Clustering.* We also used a hierarchical clustering algorithm which produced similar results. The initial 5 hierarchical levels are depicted in figure 6.10. There are known advantages and disadvantages of using different clustering approaches.

Hierarchical clustering plays an important role in our observations, as it provides a tree-like structure called a dendogram which presents hierarchical relations between clusters. Using hierarchical clustering, we capture a concentric cluster,which is not the case in K-means clustering.

Hierarchical clustering is less efficient than K-means as one has to compute at least nxn similarity coefficients and,then, update them during the clustering process. If a data

set is very large, efficiency is a key issue. Because K-means was conceptually the simpler method, we used it first on the corpus-I as its results were often sufficient for the analysis. An additional problem associated with an hierarchical clustering approach is that it is not easy to define levels for clusters.

As for the dendograms produced, one must prune the tree structure as per the hypothesis; so, deciding the level might be difficult. K-means algorithm has low complexity $O(nkt)$ where t= no. of iterations. One basic disadvantage of the K-means algorithm is that we must specify k number of clusters to be formed initially. This may lead to erroneous results when we specify less than the cluster groups. Hence, clusters are sensitive to initial assignment of centroids.

Given the input set S, the goal is to produce a hierarchy (dendogram) in which nodes represent subsets of S. In the above dendogram, IP addresses of the senders represent the nodes of the tree. Features of the tree obtained:

- The root is the whole input set S
- The leaves are the individual elements of S
- The internal nodes are defined as the union of their children

Each level of the tree represents a partition of the input data into several (nested) clusters or groups. All the IP addresses which are closely related are nested under one level (k) , if the distance of the similarity increases then it jumps altogether to another level (say k=2) indicating less association between them.

- Based upon the correlation value between senders the dendogram is plotted
- If the correlation value between 2 senders is very close then they are grouped into one
- Similarly the mechanism goes on grouping based on the correlation values until we get one single cluster as a whole

The relation between objects is shown in proximity matrix (See figure. 6.9) in which rows and columns correspond to objects. In our research, using the euclidean distance measure, we computed the proximity matrix, which represents how close the senders are from each

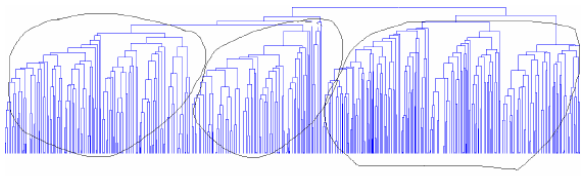

FIGURE 6.8. Representing compressed overview of dendograms of spammers

Proximity Matrix

Case	IP Adresses	Squared Euclidean Distance				
		1 xxx.xx.xxx.23	2 yyy.yy.yyy.235	3 zzz.zz.zzz.136	4 ppp.pp.ppp.230	5 qqq.qq.qqq.101
1	xxx.xx.xxx.23	.000	2.396	2.265	7.630	2.314
2	yyy.yy.yyy.235	2.396	.000	.003	6.760	.806
3	zzz.zz.zzz.136	2.265	.003	.000	6.719	.785
4	ppp.pp.ppp.230	7.630	6.760	6.719	.000	2.939
5	qqq.qq.qqq.101	2.314	.806	.785	2.939	.000
6	rrr.rr.rrr.20	1.560	.089	.067	6.554	.724
7	sss.ss.sss.229	2.569	.004	.010	6.817	.842
8	ttt.tt.ttt.210	3.271	.726	.734	3.189	.112
9	uuu.uu.uuu.229	5.040	11.968	11.712	7.070	8.149
10	vvv.vv.vvv.67	2.448	.001	.004	6.776	.817

FIGURE 6.9. Figure describes a sample of each IP address (Sender) and its proximity with all the other senders in the corpus

other. This is a matrix of: No. of senders × No. of senders. Where each sender shows his proximity with all the other senders in the corpus. It gives a dissimilarity matrix, i.e lower the value, closer the sender. Proximity values range from (0- 425) for example, the range of proximities for level=1 is from (0-0.818).

In the figure 6.10, height of the vertical lines and the range of the (dis)similarity axis give visual clues about the strength of the clustering. Long vertical lines indicate more distinct separation between the groups. Long vertical lines at the top of the dendogram indicate that the groups represented by those lines are well separated from one another. Shorter lines indicate groups that are not distinct.

Each level of the tree as shown in figure 6.10 represents a partition of the input data into several (nested) clusters or groups. All the IP addresses which are closely related are nested under one level (k) , if the distance of the similarity increases then it jumps altogether to another level (say k=2) indicating less association between them.

6.4. Clustering of Active Spammers and Timing Analysis

The most threatening spammers are the botnet spammers and within the botnet we want to catch the one's who are most active and are highly serious in spamming. We relate seriousness term to the one who is having high frequency of spamming within a given short

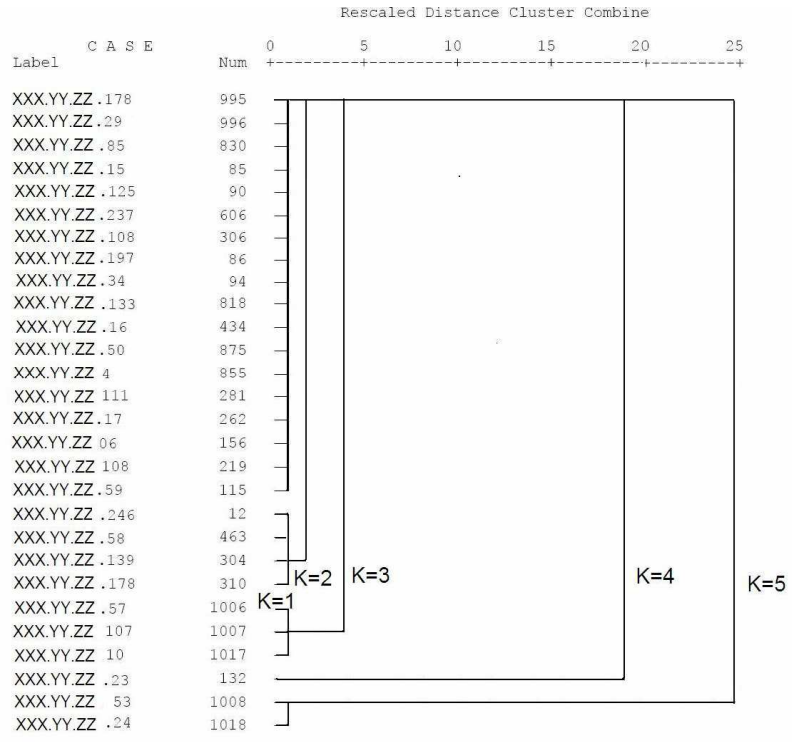

FIGURE 6.10. A sample of hierarchy levels of the algorithm up to level K=5

period of time. While it is difficult to estimate the total number of systems that participate in botnets at any point in time, but during the process of sub clustering, we could get hold of the number of spammers within the same botnet group who at the same time and in a similar pattern spam a domain. Active spammers are categorized as those senders who send spam email within the same range of time. So, spammers in the same group are active for same amount of time. Furthermore, the magnitude of the botnet threat is now widely recognized to be compounded by the emergence of an active botnet economy that is likely to be funded by an organized crime [64] [65]. We further analyze the structure and behavior of botnets, (e.g., the network effects and impact of multiple bots communicating and reacting to botmaster commands) will grow. We can analyze further from our results that within a botnet group we have sub-clusters and these sub clusters are having similar trend of spamming in terms of frequency and inter-active time, but their location of spamming is different. In our analysis the bot machines from US are spamming during the daytime whereas from a different sub cluster from the same botnet group is spamming in a similar trend from China/Australia. So one can say that though these guys have separate locations and separate active time they are managed by the same Bot Master which is sitting at a common place. We were also

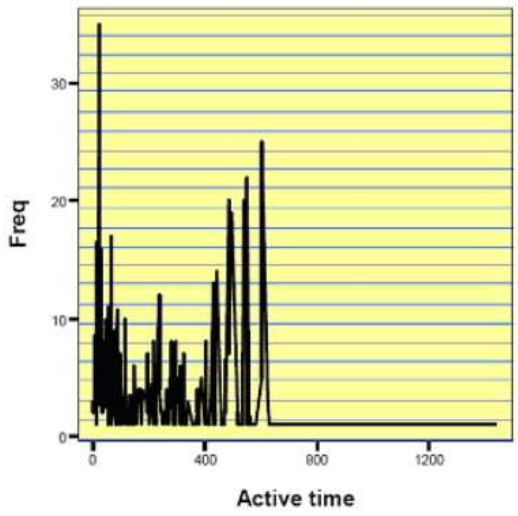

FIGURE 6.11. Analysis of active botnet groups in the spamming domain

able to track bots located at different places, by capturing their trend of spamming and by studying their timing patterns.

6.4.1. Botnet Propagation Based on Time of Arrival

The graph represents a particular botnets timing patterns based on their Time of spamming. Each bot is assumed to be a programmed machine or a compromised machine which will spam within a time slot. We see bots spamming in burst and then they are inactive throughout the day. But several bots are mastered together to keep spamming the entire day. Botnet 9 has a trend of sending most of emails within the time range of 10 am to 9 pm. There are few spam emails not falling in that category but they are treated as false positives and negatives in our case. Whereas on the other hand botnet groups 2 and 3 have patterns of spamming during late nights and early mornings so we assume them to be probably set up somewhere outside USA (usually we observe a trend of receiving spam mails early morning when we check our email in-box).

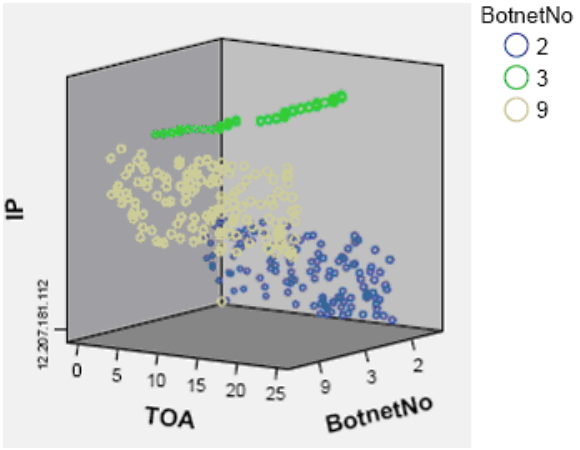

FIGURE 6.12. Timing analysis of the botnet groups

6.5. Sub-Clustering

For a particular botnet as shown in figure 6.13 we can see 3 sub clusters of activity formed. One can observe that within a botnet group we can think of sub-botnet groups. Take a closer look at botnet 9's activity.

We can observe that as the frequency increases the active time increases in case of botnet 9. Further we analyzed these spammers of botnet 9 to capture their sub botnets. We computed the distances between spammer based on their active time and frequency, and sub grouped botnets into different bots.

The distance metric which we selected was Pearsons Coefficient. In statistics, the Pearson product-moment correlation coefficient (sometimes known as the PMCC) (r) is a measure of the correlation of two variables X and Y measured on the same object or organism, that is, a measure of the tendency of the variables to increase or decrease together. It is defined as the sum of the products of the standard scores of the two measures divided by the degrees of freedom:

$$r = \frac{\sum Z_x Z_y}{n-1} \tag{26}$$

Note that this formula assumes the Z scores are calculated using standard deviations which are calculated using n - 1 in the denominator. Using this distance metric we tried

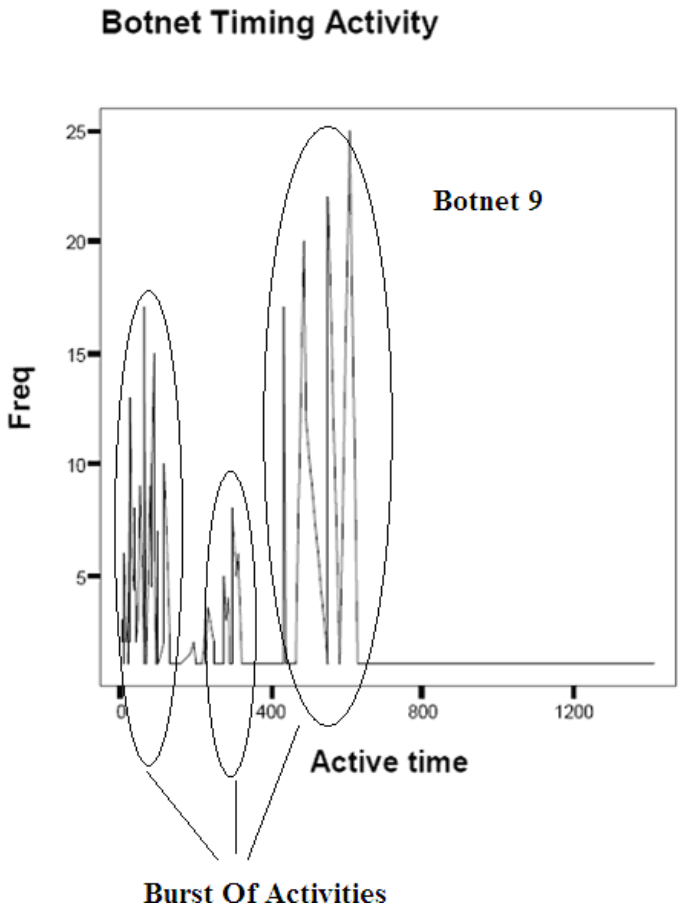

FIGURE 6.13. Botnet timing activity for Group 9 which shows burst of activities in 3 sub-groups

to analyze the sub-clusters within the same botnet group. In order to do that, we started identifying the botnet group into 3 sub-clusters. The cluster membership is shown as follows.

We can categorize looking at the scatter plot that all the botnet 9 spammers can be categorized into 3 groups or so and those subgroups have similar activity in terms of active time and frequency. For the spammers within the botnet 9 having a high active time that is the amount of time (Inter-arrival time) is high then their distance from the classification center is less. Thresholding is done to capture serious spammers from each sub-cluster of botnet group. The measurement of botnet is categorized using the distance metric.

The above graph correlates with the active time vs. frequency where within the first 300 minutes there is a similar activity and they all fall into sub cluster 1 and then the second one starts of with the active time mote than 800 minutes whereas there is no activity within

Cluster Membership

Case Number	IP	Cluster	Distance
1	65.175.70.104	1	167.112
2	69.6.11.149	1	166.104
3	8.10.32.133	1	163.113
4	216.66.66.178	1	161.113
5	71.6.130.100	1	160.128
6	70.47.60.13	1	159.113
7	8.10.16.240	1	158.116
8	204.15.226.7	1	157.104
9	209.128.85.120	1	156.113
10	209.73.178.130	1	156.113
11	69.30.217.238	1	154.104

FIGURE 6.14. A table consisting of spammers having cluster-membership and their distances from the centroid

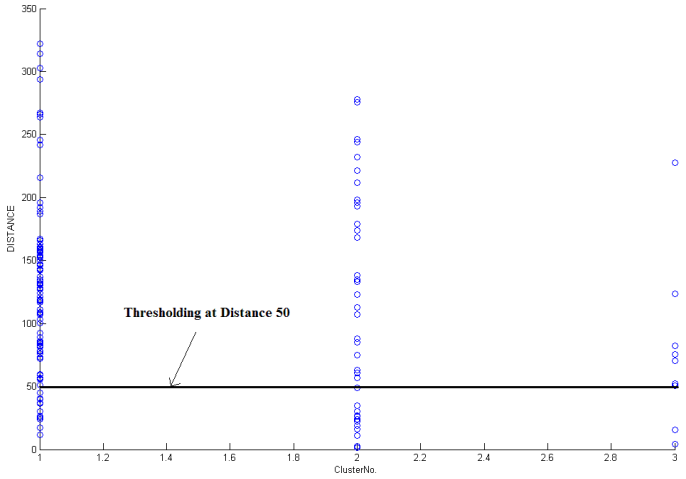

FIGURE 6.15. Thresholding active spammers within a botnet group based on their distance from the cluster centroid

the period of 400 minutes and 700 minutes during that time botnet 3 is the most active. So it is quite obvious that these 2 botnets are managed by the same bot master.

We can see a common trend between different groups of bots though with different active time. This proves that botnet masters are placed at one place at a different location and it controls other bots. These bots are automated machines most of the time and for e.g. a spam coming from a country like Australia during daytime might be night in USA. So the

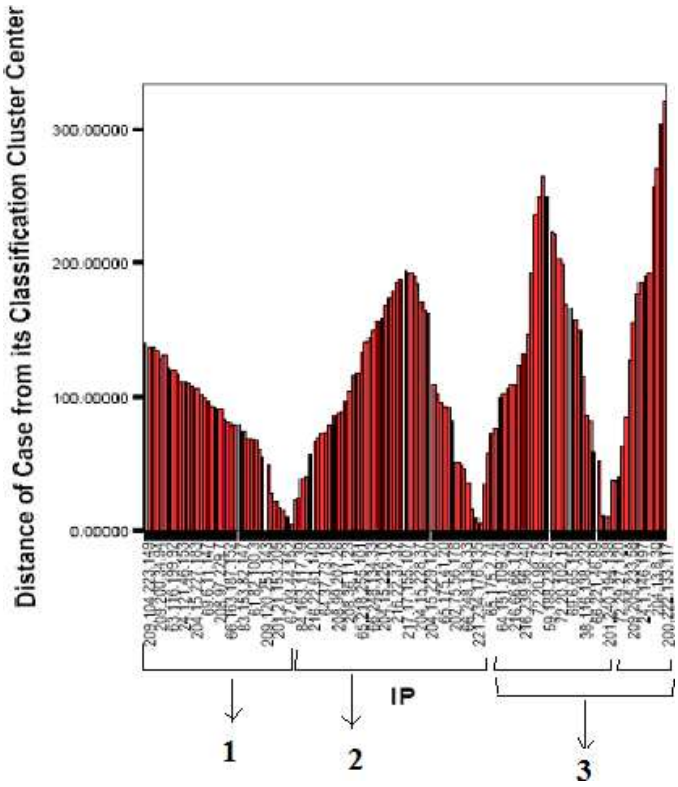

FIGURE 6.16. Categorization of the botnet group based on their distances

active time is more but the trend is similar. Thus we capture the trend in spamming using the timing analysis.

An interesting pattern each sub botnet follows it starts with a distance of 150 or higher and comes down to zero distance. It is highly possible that these two sub-bots are slaves and they are managed by a master sitting somewhere else.

6.6. Hand Labeling & precision

If an instance (here an, email) is unwanted (spam or phishing) and classified as unwanted, then, it is counted as true positive, "TP". If an instance is wanted (legitimate) and classified incorrectly as unwanted, it is counted as false positive "FP". Let "P" and "N" be the total number of positive and negative instances in a corpus; we determined the precision, true positive rate "tp_{rate}" and false positive rate "fp_{rate}" of researched classifier as:

$$Precision = \left\{ \frac{TP}{TP + FP} \right\} \tag{27}$$

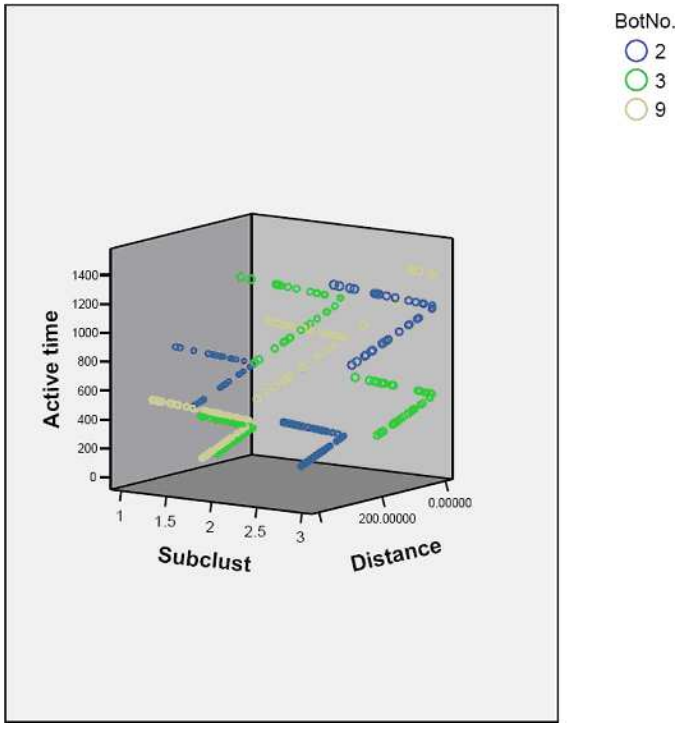

FIGURE 6.17. Figure depicts an interesting pattern of each sub-botnet group

$$tp_{rate} = \left\{ \frac{TP}{P} \right\} \tag{28}$$

$$fp_{rate} = \left\{ \frac{FP}{N} \right\} \tag{29}$$

To calculate the efficiency of our clustering technique and botnet classification, we required a means of measuring the precision. So, we hand labeled the spam data set and calculated the accuracy of the technique. The most significant feature for classification of botnet grouping was the network id and location from where the spam email was sent. We can observe from the relevant work [66, 67, 26] that bots possess a similar network Id and they may have same location of spamming, thus, because of the nature of bots, spamming usually inherits similar behaviors when spamming a domain. Table 6.3 describes hand labeling of botnet 9 and we can see that the spammers display similar behavior and characteristics. They have their spamming time very close to each other.

TABLE 6.3. Example: handlabelling of one of the botnet sub-group

IP Addresses	Location	Bot Group	Month	Day	Time	C.L
X.X.X.58	United States	b9	Aug	6	2:26:25	1086
X.X.X.75	Malaysia	b9	Aug	6	2:30:27	1250
X.X.X.98	Netherlands	b9	Aug	6	2:31:12	1076
X.X.X.35	United States	b9	Jul	23	5:10:36	873
X.X.X.66	United States	b9	Jul	23	5:10:36	880
X.X.X.17	United States	b9	Jul	23	5:11:41	713
X.X.X.23	China	b9	Jul	23	5:11:19	801
Y.Y.Y.95	United States	b9	Jul	23	5:08:41	736
Y.Y.Y.05	United States	b9	Jul	23	5:10:14	562
Y.Y.Y.29	United States	b9	Jul	23	5:07:23	736
Y.Y.Y.34	United States	b9	Jul	23	5:08:54	645
Y.Y.Y.27	United States	b9	Jul	23	5:11:01	1141
Y.Y.Y.93	United States	b9	Jul	23	5:11:14	407
Y.Y.Y.33	United States	b9	Jul	23	5:12:14	765
Y.Y.Y.69	United States	b9	Jul	23	5:11:14	1961

During hand labeling, we also correlated characteristics of spam emails, such as, country, city, network id, time of arrival, active time, content length and frequency. The following properties are observed during our experiments and hand labeling:

- Spammers send large numbers of emails in a short period of time.
- Spammers send the same content repeatedly (with different IDs).
- Spammers send the same number of emails in a given day.
- Spam from a botnet arrives at its destination at the same time although spammers are located in geographically distributed locations.
- Based on our corpus,major source of spam is from USA, followed by Malaysia and China. Moreover, due to the time difference, Asia is actively sending spam while USA is sleeping.
- A small number of botnets account for most spam.
- Spammers and legitimate users share SMTP paths and relays [31].
- In the email corpus of a single user, most spam was generated from a few botnets and the emails were highly correlated.

TABLE 6.4. Precision of analyses performed on corpus-I using K-means

Corpus-I	Cluster Analyses	True Positives	False Positives	False Negatives	Precision Hits
Botnet2	417	379	34	4	91.76%
Botnet3	97	81	11	5	88.04%
Botnet9	484	436	41	7	91.40%

TABLE 6.5. Precision of analysis performed on corpus-I using hierarchical clustering

Corpus-I	Cluster Analyses	True Positives	False Positives	False Negatives	Precision Hits
Botnet2	371	242	60	15	80.10%
Botnet9	257	222	30	5	88.09%
Botnet3	396	298	91	7	77.00%

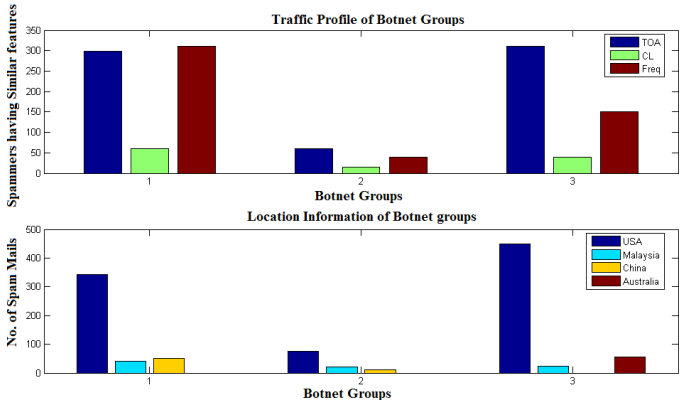

FIGURE 6.18. Email traffic based on the geographic locations of spammers in Clusters 2,3 & 9

- Hand labeling of botnets for a large corpus is humanly difficult and requires behavior-based automation (e.g., the techniques described in this paper).

We used the true positive rate and false positive rate during Receiver Operating Characteristics [68] analysis performed to optimize the performance. Tables 6.4 and 6.5 shows the precision of our filter.

6.7. Traffic Shaping of Spam Botnets

The most challenging aspect of dealing with spam has been the seriousness, threat and growth of 'botnets'. According to a recent survey, spammers sent an estimated 80% of email spam by using zombie PCs. One of the most common usages of botnet's is to launch

massive spams. Spam remains an annoying problem because a majority of spam filtering techniques focus on the content of an email, which is in complete control of the spammers. So, such techniques are not of use as their classification strategies depend upon the message's meaning. The approach avoids this limitation I base classification on the individual user's behavior. So one needs understanding of the network of botnet, its growth dynamics, threat level and an approach to avoid automated 'Bots'. Not much research on botnet is stated as yet on the behavior analysis, there are few papers related to this work but they have more to do with the IRC relays and their study.

An in-depth understanding of botnet behavior is a precursor to building effective defenses against this serious and fast growing threat in emails and in future it would be the Voice over IP (VoIP) applications. Using the technique I was able to perform a range of experimental study on new methods and tools for characterizing, comparing, identifying, tracking, dismantling, and preventing botnets.

The most threatening spammers are the botnet spammers and within the botnet we want to catch the ones who are most active and are highly serious in spamming. I relate seriousness term to the one who is having high frequency of spamming within a given short period of time. While it is difficult to estimate the total number of systems that participate in botnets at any point in time, but during the process of sub clustering, I could get hold of the number of spammers within the same botnet group who at the same time and in a similar pattern spam a domain.

Active spammers are categorized as those senders who send spam email within the same range of time. So, spammers in the same group are active for same amount of time. Furthermore, the magnitude of the botnet threat is now widely recognized to be compounded by the emergence of an active botnet economy that is likely to be funded by an organized crime.

I further analyze the structure and behavior of botnets, (e.g., the network effects and impact of multiple bots communicating and reacting to botmaster commands) will grow. I can analyze further from the results that within a botnet group it has sub-clusters and these

sub clusters are having similar trend of spamming in terms of frequency and inter-active time, but their location of spamming is different. In our case the bot machines from US are spamming during the daytime whereas from a different sub cluster from the same botnet group is spamming in a similar trend from China/Australia. So one can say that though these guys have separate locations and separate active time they are managed by the same Bot Master which is sitting at a common place. From the analysis I can track different bots located at different places, by capturing their trend of spamming and by doing timing analysis.

Main contribution of traffic shaping analysis

The main contribution is to develop a Traffic control mechanism which, categorize email senders into categories such as legitimate, suspicious and Bots. I analyze traffic from their behavior patterns and delay the traffic of their spamming which allows real time filtering techniques to be used without the risk of false positives. I set up a simple traffic shaping technique called Turing test in our analyzer for all the suspicious and spam mails, which will eliminate the use of automated machines to send spam mails. In addition, I was able to find the trace-ability of botnets based on Information theory.

6.8. Methodology

For this analysis I have considered a corpus of emails which includes both legitimate as well as spam emails. Based on the spammers' locations, I categorized the traffic profile of the botnet groups.Identifying the spammers' physical locations cannot be achieved using the originating location of the spam as spammers use compromised machines(Bots).

First step is to separate legitimate senders from spammers using principal component analysis (PCA) [30] [23]. From the spammers only traffic, I apply clustering techniques on their feature set and identify botnet groups [24] [69]. Our study only focuses on the header analysis which is not under the control of the spammer whereas a spammer can spoof the content of an email. Using the analysis I was able to separate legitimate and spammers traffic and also identify Bots with a precision of more than 90%.

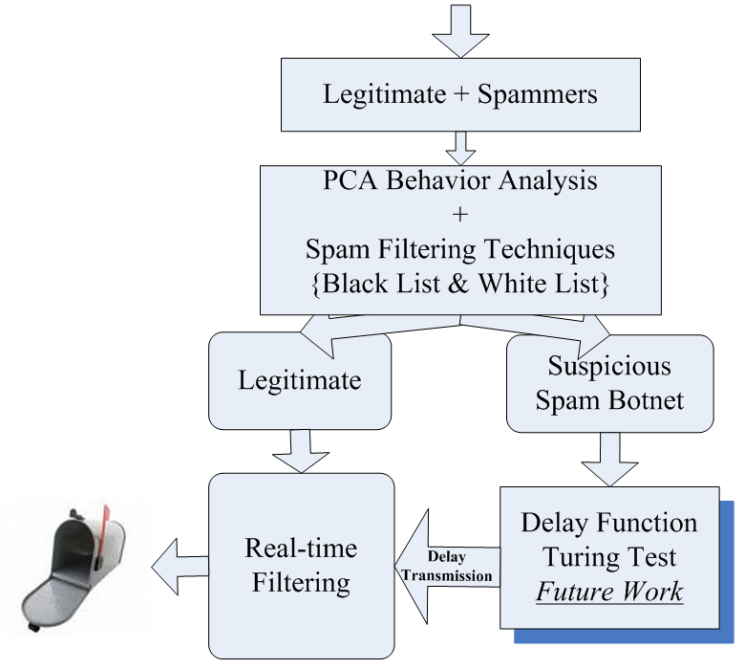

FIGURE 6.19. Flow diagram of pre-filtering analysis to avoid spam bots using traffic shaping techniques

After separating out the legitimate traffic from spammers or suspicious spam Bots, I pass the legitimate for advanced filtering techniques. Whereas suspicious and spam Bot traffic is passed ahead for a Turing Test which will delay the transmission. According to [27] 90% of spam mails are avoided if I delay their transmission channel. In our proposed approach, Turing test will not only delay spam traffic but also mitigate automatic use of zombie machines (bots) which are used to send large chunk of spam mails. Methodology of the proposed filter is a pre-filtering technique, which is controlled by the network administrator.

6.9. Conclusion

An in-depth understanding of botnet behavior is a precursor to building effective defenses against this serious and fast growing threat in emails and in future it would be the voice over IP applications. Using my technique I was able to perform a range of experimental study on new methods and tools for characterizing, comparing, identifying, tracking, dismantling, and preventing botnets.

In this work I investigated the clustering structures of spammers based on spam traffic collected over a period of 6 months. Our analysis shows that the relationship among spammers demonstrate highly clustering structures based on features such as content length, time of arrival and frequency of an email. I extracted many features like content type and storage time but did not use them because the eigenvalues were very low and these features were eliminated during Scree plot.

The inter-arrival time of spam from the same group of spammers exhibits long-range dependence in the sense that the spam from the same group of botnets often arrives in-burst. It was also observed that spammers associated with multiple groups tend to send more spam in the near future. I need to emphasize that group-based method may not be highly effective as a stand-alone approach as some groups may have only one member. Some botnet groups had 1-10 spammers and I categorized them as outliers in this analysis.

Using the described clustering techniques, I could accurately identify botnets as they usually inherit similar behavior when spamming a domain. By hand labeling I was able to identify that these botnets indeed fall into a particular cluster with a precision closet to 90%. I will continue to explore interesting properties of the clustering structures of telemarketing spammers as our future work and also deploy the above techniques as a complementary tool for existing anti-spam tools.

CHAPTER 7

BEHAVIOR ANALYSIS OF CALLEE FOR UNWANTED CALLS

Presence technology is going to be an integral part of the next generation of communication technology [70] [71] [72]. It can eliminate telephone tag between two parties (caller and callee), which will increase productivity of the parties and reduce unnecessary bandwidth usage of unwanted calls. In this chapter, I propose a willingness estimator that computes willingness level of a specified callee. By knowing the willingness value of the callee the caller can decide on proceeding with the call or not. The proposed willingness estimator is tested with real mobile user data, and these results are highly accurate. The present technology can use callees mobile history (time, location, day) to accurately estimate willingness level of callee and this could serve as one of the future presence based service.

7.1. Methodology

In our daily life, when we make a phone call, we often guess whether or not our call will be answered by the intended called party (callee). Most of the time, we want our call to be answered by the intended callee, however other times we want to leave a message instead of speaking to the callee. Therefore, when we make a phone call, I try to estimate our chance of being answered by the callee. I have based this estimation on

Time of the day: The callee is not likely to take a call during his/her busy hours or while he/she is sleeping at night but more likely during his/her free hours such as time before work or during his/her break or driving home after work. Therefore, I estimate the callee's willingness of taking a call based on when (time of call) we make the call.

Location: The callee is not likely to take any call while he/she is at work or in the theater but he/she more likely to take a call while he/she is at home or apartment. Therefore, I can also base an estimation of the callee's willingness on where (location) the callee is located when one makes a call.

Day of the week: Since we all have different schedules, most people go to work during weekdays and stay at home on the weekends. Thus, incoming calls during the weekends are more likely to be answered than during the weekdays. Likewise, I base the estimation of the callee's willingness on what day of the week we make the call.

As previously mentioned, the caller wants to know the callee's willingness of taking the call before the caller decides whether or not to make that call. *From the communication network controllers point of view, this can help traffic congestion since the caller knows the callee's willingness level so the caller might not initiate a call which can reduce the traffic in the network and also save the caller's available minutes.* Therefore, I propose the willingness estimator (WE) for computing the willingness level of the callee, which can be deployed at the base station. The basic architecture of the WE and its service flow diagram are shown in figure 7.1 and figure 7.2 respectively.

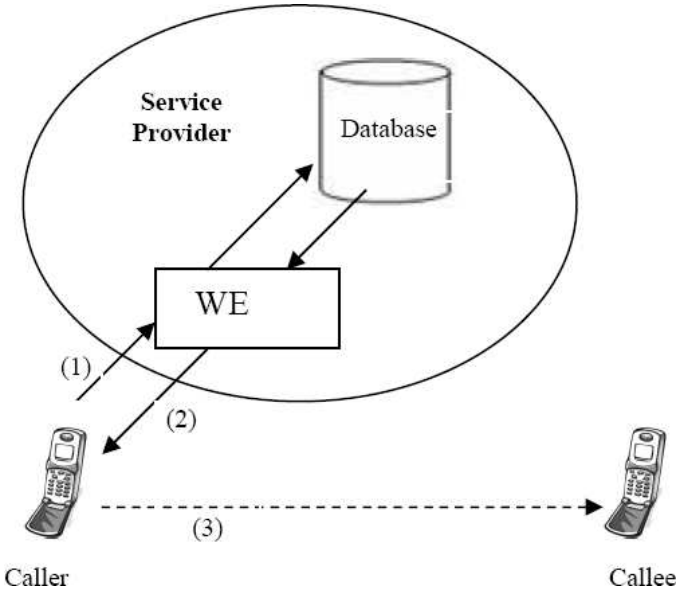

FIGURE 7.1. Basic service flow diagram

When the phone user makes a request to the WE for the callee's willingness level, the WE takes information of time of the call (current time), day of the call (current day) and the callee's location information from the service provider and callee's call history from database. The WE computes the willingness level based on time of the call, location of the callee, and day of the call. The principal component analysis (PCA) is applied to compute the amount of

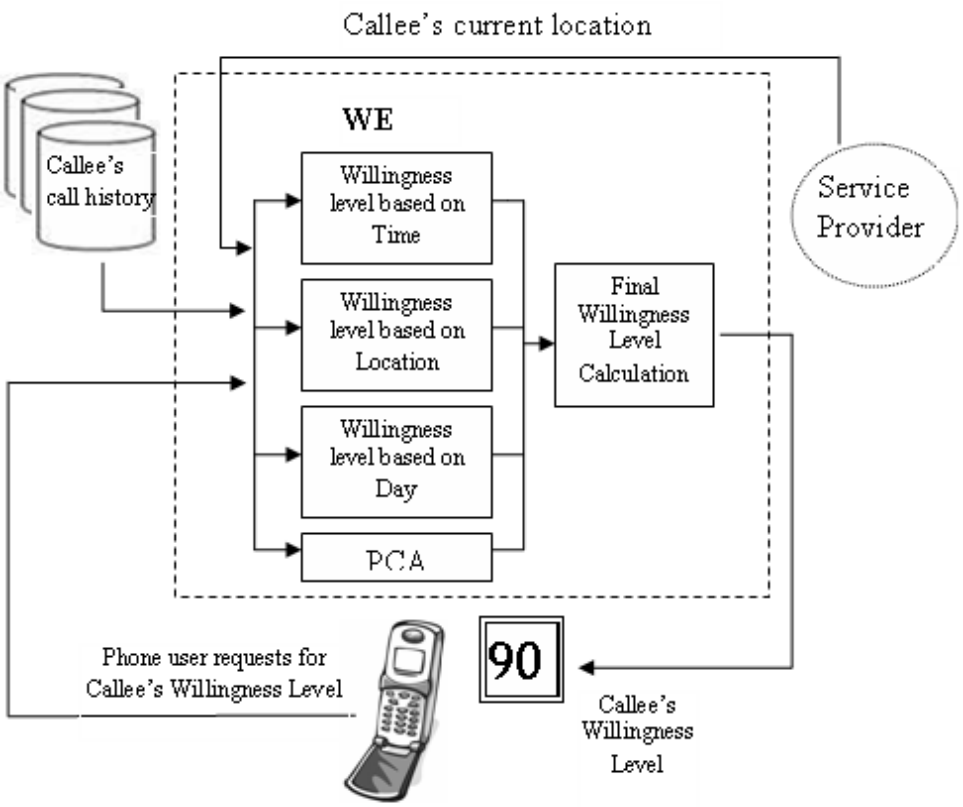

FIGURE 7.2. Architecture of the willingness estimator

contribution (importance) of those three input parameters have toward the final willingness level. The final willingness level is computed and forwarded to the phone user. The phone user then makes a decision whether or not to initiate the call.

7.2. Willingness Computation

Willingness can be defined as a wanted activity [73] [74]. This refers to receiving wanted calls in case of voice call. As described in section 7.1, willingness level of the callee depends on time of the call, location of the callee, and day of the call.

I define the willingness level based on time of the call as the sum of the call frequency at that particular time divided by the total number of calls, which is given in equation 30 where $W_T(t)$ is the willingness based on time of call of t^{th} hour, N is the total number of calls, $n_t(i)$ is the j^{th} call frequency at t^{th} hour, and j = 1, 2, 3, , m where m is the total

number of days of observation.

$$W_T(t) = \frac{1}{N}\sum_{j=1}^{m} n(i) \tag{30}$$

Similarly, the willingness level based on callee's location is defined as the sum of the number of calls (call frequency) that the callee has received at a particular location divided by the total number of calls, which is given in equation 31 where $W_L(l)$ is the willingness based on callees location of l^{th} location, N is the total number of calls, and n_l is the number of calls at l^{th} location.

$$W_L(l) = \frac{n_l}{N} \tag{31}$$

Likewise, the willingness level based on day of the week ($W_D(d)$) is defined as the sum of the number of calls that the callee has received each day of the week (nd) divided by the total number of calls (N), which is given by equation 32. The sample plots of W_T, W_L, and W_D are shown in figure 7.3.

$$W_D(d) = \frac{n_d}{N} \tag{32}$$

Therefore, the final willingness level (W) is given by equation 33.

$$W = C_T W'_T(t) + C_L W'_L(l) + C_D W'_D(d) \tag{33}$$

The final willingness level is the sum of the product of the normalized (rescaled to the maximum value) willingness level (W') based on time of call, callees location, and day of the call and the contribution coefficients C_T, C_L, and C_D corresponding to time of call, callees location, and day of the week respectively. These contribution coefficients are introduced here because I believe that these three parameters (time, location, and day) have different contributions that impact the final willingness level. To obtain the values of these coefficients, PCA is applied.

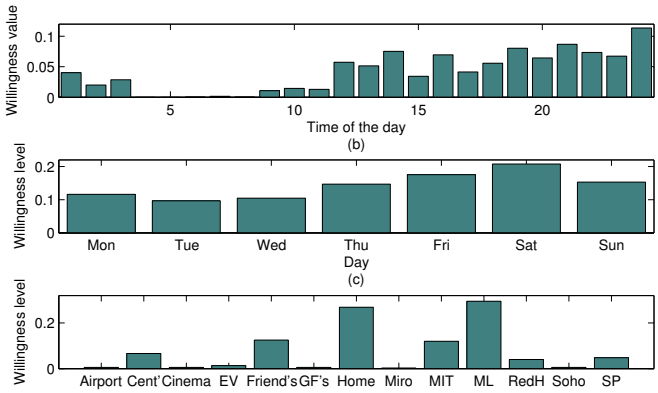

FIGURE 7.3. (a) A sample willingness level based on time of the call (b) A sample willingness level based on day of the week (c) A sample willingness level based on location

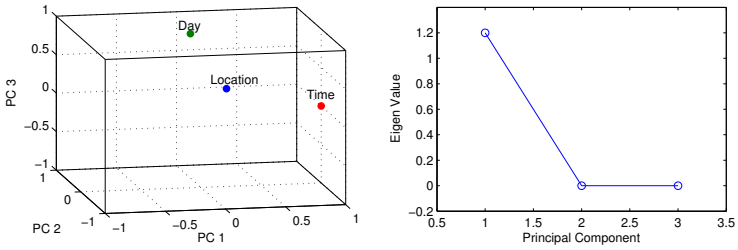

FIGURE 7.4. (a) Principal component plot where time of the call, location of the callee, and day of the call are represented with red, blue, and green dots respectively (b) Scree plot shows that the first principal component has the highest eigenvalue

I apply PCA analysis to find the contribution of each parameter towards the final willingness level based on time of the call, callees location, and day of the call. I convert the three dimensional dataset into three PCs, and further look into the first PC which captures maximum variance of the data. Figure 7.4 shows where the three parameters (time, location, and day) lay on the principal component plot.

The first PC captures maximum variance as it has the highest eigenvalue and the other two components can be eliminated from the analysis as referred to the scree plot in figure 7.4; so area of interest lies in the first PC. Table 1 shows that on the first PC, the orders of significance from high to low is time of the call, callee's location, and day of the call respectively. The physical significance of PCA is to find the underlying pattern in the dataset,

TABLE 7.1. Principle component matrix shows the values of time of the call, callees location, and day of the call on the principal components

Parameter	**PC1**	PC2	PC3
Time of the call	**1.00**	0.00	0.00
Callee's location	**0.40**	0.92	0.00
Day of the call	**-0.08**	0.04	0.99

and detect each fields contribution. Hence the numerical values of the contribution coefficient can be computed from Table I as the ratio of values the time, location, and day lied on the first principal component. For this sample the particular user has contribution coefficients of time, location, and day as 67.613%, 27.04%, and 5.206% respectively. Thus time factor plays a major role for this user as the majority of the calls are received during a particular time period as it can be seen from figure 7.3. So the contribution of time towards the willingness value for this user is high, whereas the location has the second highest contribution while day of the week has the lowest significance. Based on my analysis, I discover that PC values vary from person to person as PC captures individuals behavior.

7.3. Validation

To evaluate the accuracy of the model proposed, an experiment is conducted with actual call logs of 100 phone users (described in section 7.1) over a period of 8 months. I randomly selected 10 phone users who are students, professors, and staffs to represent all users. The first 6 months of data are used to compute the initial willingness level. The following 2 months are then used to validate the proposed Willingness Estimator (WE) where the willingness level is recomputed for each incoming call and verified against the result of the call (missed or accepted).

The accuracy is measured by the unwanted rate over the range of different willingness levels. The unwanted rate is the ratio of the number of missed calls to the total number of calls at the given willingness level. An assumption is made here that a missed call is considered as an unwanted call.

Figure 7.5 shows the resulting contribution coefficients by applying the PCA for 10 different users to compute the contribution that each field makes towards the final willingness

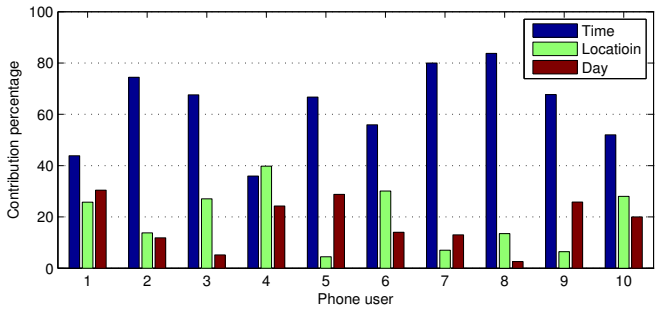

FIGURE 7.5. Contribution coefficient plot of 10 different phone users resulting of the PCA

TABLE 7.2. Experiment results

Phone User	Incoming calls	Missed calls	Unwanted rate(%)									
			Willingness levels(%)									
			0-10	10-20	20-30	30-40	40-50	50-60	60-70	70-80	80-90	90-100
1	541	117	15.38	48.71	19.65	15.38	6.83	3.45	0.00	0.00	0.00	0.00
2	438	106	24.52	20.18	13.20	9.43	14.00	10.00	6.60	2.07	0.00	0.00
3	210	28	28.57	0.00	14.28	28.57	14.30	14.28	0.00	0.00	0.00	0.00
4	620	176	14.00	29.00	11.50	6.25	10.50	15.00	10.00	3.75	0.00	0.00
5	134	46	26.08	32.78	32.78	0.00	0.00	0.00	8.36	0.00	0.00	0.00
6	88	30	20.00	60.00	13.30	0.00	6.70	0.00	0.00	0.00	0.00	0.00
7	104	59	5.09	57.63	18.64	0.00	18.64	0.00	0.00	0.00	0.00	0.00
8	861	128	12.50	14.06	55.47	17.97	0.00	0.00	0.00	0.00	0.00	0.00
9	170	28	7.21	28.50	64.29	0.00	0.00	0.00	0.00	0.00	0.00	0.00
10	169	17	80.00	10.00	10	0.00	0.00	0.00	0.00	0.00	0.00	0.00

level. I observe that the contribution coefficients are different for all users. It can be observed that the phone users 1, 4, 6, and 10 (who are students) have tendency of taking calls at any time of the day thus "time" contribution coefficients towards willingness are relatively less than those users 5, 7 and 9 (professors) and users 2, 3, and 8 (staffs).

The numerical results are shown in table 7.2 and the graphical representation is illustrated in figure 7.6. As one can see in table 7.2, unwantedness is higher in the low willingness regions (less than 50%) than the high willingness (more than 50%) regions. Again, the proposed WE only provides the willingness level of the specified callee; however it is up to the phone user to make a decision whether to initiate call. Nevertheless, the experimental results show high accuracy of WE. Further, results validate the hypothesis that when the estimated

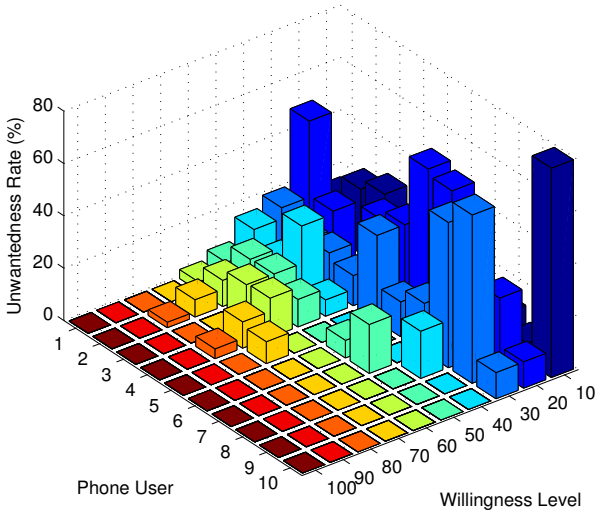

FIGURE 7.6. Experimental results show the unwanted rates over the range of willingness levels for 10 different phone users

willingness level is high, most of the calls are answered, whereas few calls are answered at the low willingness level.

7.4. Conclusion

In this thesis, I proposed a willingness estimator (WE) that computes the willingness level of the intended callee. I propose a special button on the phone called presence. Before making a call, one can press this button for finding out if the calleee is indeed available. Next, it is up to the phone user to make a decision whether to initiate a call. I believe that the callees willingness of accepting a call is influenced by time of the call, location of the callee, and day of the call. The willingness computations based on these three factors are carried out as a simple ratio of the call frequency at particular (time, location, day) to the total number of calls. I believe that contribution of these factors to the overall (final) willingness level are different. It is validated by the principal component analysis that these three factors have different relevant weights to the final willingness level. Hence, the final willingness level of the callee is the sum of the product of the willingness based on time of the call, callees location, and day of the call, and its corresponding contribution coefficients. The accuracy of the model is evaluated with the actual call logs of 10 phone users from

the MIT's reality mining data sets. The WE performs well with high accuracy as when computed willingness level is low; the unwanted rate is high and vice versa.

I believe presence will be a new service offered by service providers [75] [76]. If the willingness level is high, it is proved that the call will be answered but when the willingness level is low, the call is most likely to reach the callees voice mail. Most of the time, we want to speak to the callee but some other times we wan to leave a more meaningful message therefore this service will give the phone user the ability to predict the reaction of the callee after the phone ring. This service can also be useful for sales people to know when to make an important call to the customers. This service can also help the call traffic congestion by reducing unwanted communication traffic.

In reality, there are many other factors that affect the willingness of the callee such as mood (state of mind), social relationships (between callee and caller), situation or status of the callee (e.g., out of available minutes, out of battery, callee has a second line, callee is in emergency, etc.) To quantify these factors is a very challenging task. I am working on other estimation techniques for improving the accuracy of the WE. These issues, among others, will be be addressed by future work.

CHAPTER 8

BEHAVIORAL ANALYSIS USING ENTROPY

8.1. Introduction

In this chapter I propose a mechanism to classify incoming traffic into buckets of Legitimate and Suspicious spam. I computed the entropy values for parameters such as active time, time of arrival, frequency, and content length which are extracted using Header analysis. Based on these values, I sub-classified suspicious bucket into spammers and botnet spam. I have separated the legitimate and spam traffic. Next, I detected and grouped the spam mail into botnets based on the entropy values. There was a high correlation of entropy values for a group of bots and this information helped to classify the botnets further. In future, based on the classification, I plan to delay the traffic generated from these bots and I believe that this delay can prevent further spam from these bots. I would like to setup a turing test on all the suspicious and spam mails. This can further block the spam from these sources. To validate the effectiveness of the proposed mechanism I analyze traffic in three experiments. The experimental results reveal that the techniques are applicable for detecting botnets and further mitigate future actions with a precision of more than 95%.

8.2. Methodology

For analysis I have considered a corpus II of size 8000 emails which includes both legitimate as well as spam emails. Based on the spammers' locations, I categorized the traffic profile of the botnet groups. Identifying the spammers' physical locations cannot be achieved using the originating location of the spam as spammers use compromised machines(bots). The motivation of this work is to understand the behavior of spammers through a large collection of spam mails. To gain this understanding, I analyzed a data set collected over 2.5 years (corpus-I) and developed an algorithm which gives us the botnet Features and, then, classifies them into distinct groups. I use principal component analysis (PCA) to analyze the

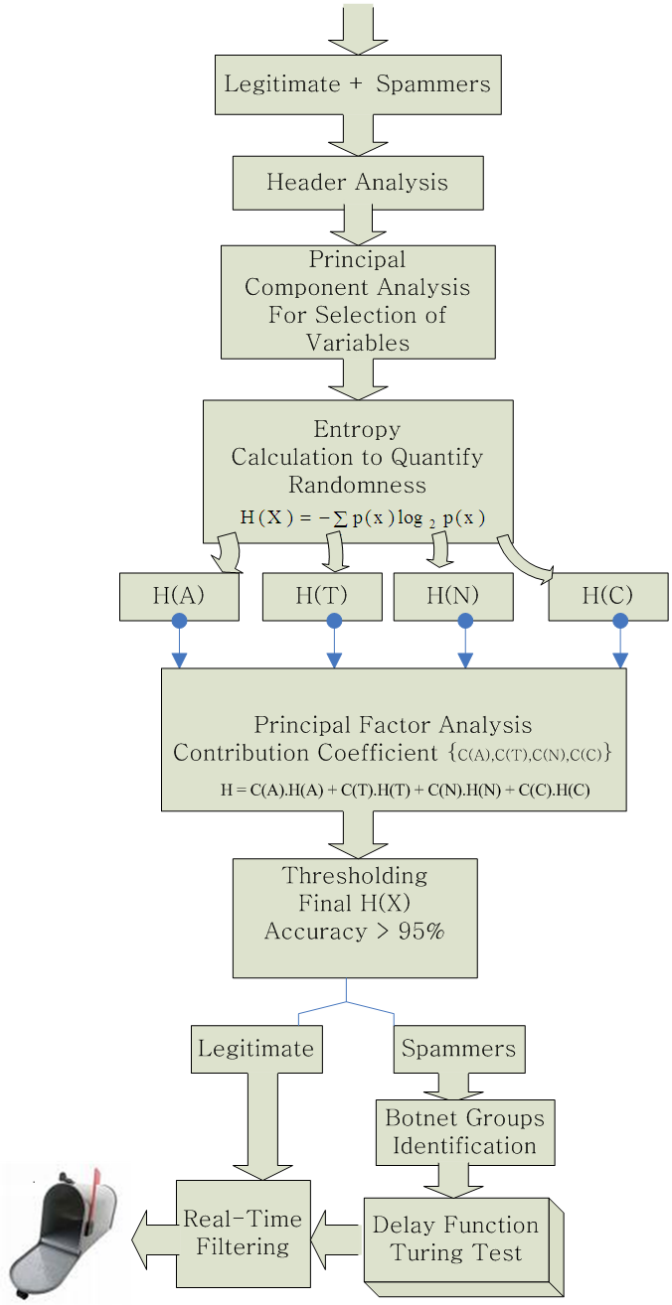

FIGURE 8.1. Flow diagram of entropy based classification spam bots using traffic shaping techniques

association patterns of groups of spammers and to analyze the behavior of individual spammers within a given domain. These analysis' are based on features which capture maximum variance of the information I have clustered.

Phase 1 : First phase contains incoming traffic of legitimate as well as spammers. I classified each user's behavior based on features of its header contents. And categorized each

email spammer based on features like IP address, content length, time of arrival, frequency of spamming and content type. For analyzing spammers' behaviors, other parameters such as reciprocity, read emails, and storage time do not apply, as it is assumed that users do not read telemarketing emails. At the first step, I develop a feature set for each incoming mail, as a data set matrix.

Phase 2 : I identify the underlying spamming patterns by applying Principal Component Analysis (PCA) [30] which allows the association patterns of groups of spammers and the individual behavior of a spammer in a given domain [77] [78]. With the aid of PCA it is possible to find association patterns such as,

- Spammers spamming at the same time of the day
- Spammers sending the same content over and over
- Spammers changing their email id's and spamming the same recipient
- Spammers sharing contact lists

Phase 3 : After extracting the features having maximum variance, I apply information theory techniques, to quantify randomness of users for legitimate as well as spam bot. In the analysis Time of arrival, Active time, content length and Frequency capture maximum variance(from phase 2). I found the randomness of all incoming traffic in terms of the above mentioned variables. Looking at the randomness with respect to different variables, I was able to detect a relationship of entropy values between these variables

Phase 4 : In phase 3 I get randomness based on individual variables. In order to get final randomness of senders we need a technique to quantify the contribution given by each variable towards total randomness. I use Principal-factor analysis to determine the contribution coefficient of each variable. The final Randomness level is the sum of the product of the normalized randomness level based on time of arrival, active time, content length, frequency and the contribution coefficients.

Phase 5 : In this phase I categorize traffic into buckets such as legitimate, spammers and bots based on the final randomness value. I set a threshold value which is an automatic receiver operating characteristic curve(ROC) for legitimate users, spammers and different

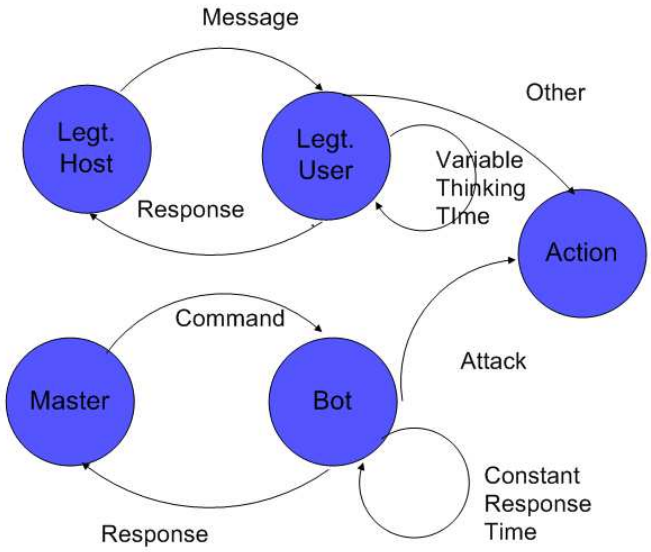

FIGURE 8.2. Comparison of human and bots behavior using a transition diagram

botnet groups. On handlabelling the data I found the precision of the analyzer for classification into buckets is more than 95%.

Phase 6: I set up a simple traffic shaping technique [25] called turing test in the analyzer for all the suspicious and spam mails, which will eliminate the use of automated machines to send spam mails. After separating out the legitimate traffic from spammers or suspicious spam bots, I pass the legitimate for advanced filtering techniques. Whereas suspicious and spam bot traffic is passed ahead for a turing test which will delay the transmission. According to [27] 90% of spam mails are avoided if we delay their transmission channel. In the proposed approach, turing test will not only delay spam traffic but also mitigate automatic use of zombie machines (bots), identified from previous phases, which are used to send large chunk of spam mails. However, traffic shaping does not block email, it is used as a prefiltering analysis to real-time filtering techniques to be used without the risk of false positives.

8.3. Randomness of Bot Structures

How easy it is to detect/trace botnet spam machines?

One has to distinguish between normal and malicious activities. In figure 8.2 one can see that after bot receives commands from their controller, they respond immediately and

accurately, which gives us a constant response time in terms of bot structure. In contrast, if a legitimate host receives a message, it responds or performs an action from a wide variety of possibilities, after a variable thinking time. A botnet machine we assume performs a preprogrammed set of activities, therefore, we have all the variables of header analysis used as metrics for botnet detection. To quantify the randomness or amount of predictable structure in an individual botnet group, the information entropy can be used. The information entropy or Shannons entropy is a measure of uncertainty of a random variable. The information entropy as given in 34 was introduced by Shannon [50].

$$H(X) = -\sum_{x} p(x) log_2 p(x), \tag{34}$$

where X is a discrete random variable, and the probability mass function $p(x) = Pr(X = x)$. The spam botnet pattern can be observed from the active time, time of arrival, frequency and content length of email spam. Let A, T, N and C be random variables representing active time, time of arrival, frequency and content length respectively. The entropy of Active time can be calculated by 35.

$$H(A) = -\sum_{t} p(a) log_2 p(a), \tag{35}$$

where the probability $p(a)$ is a ratio of the number of mails during t^{th} hour slot to the total number of mails of all time slots. The entropy of time of arrival can be calculated by 36.

$$H(T) = -\sum_{t} p(t) log_2 p(t), \tag{36}$$

where the probability $p(t)$ is a ratio of the number of mails during t^{th} hour slot to the total number of mails of all time slots. By the same token, the randomness in the spammers active time content length $H(C)$ and frequency $H(N)$, can also be quantified using information entropy, which is defined in 34.

Results of Entropy values shows that behavior structure of botnets are more predictable and less random as compared to normal spammers. Also, by comparing legitimate traffic

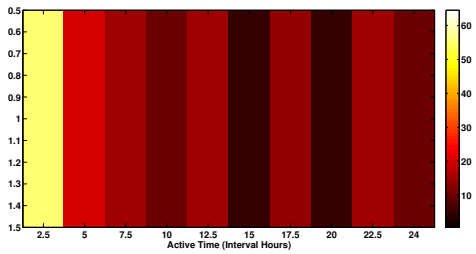

FIGURE 8.3. Example of a high entropic botnet group based on active time

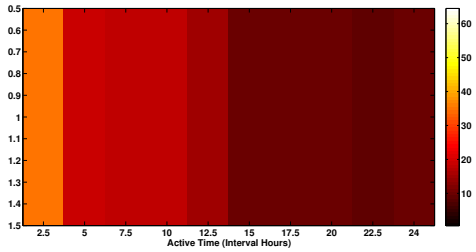

FIGURE 8.4. Example of a low entropic botnet group based on active time

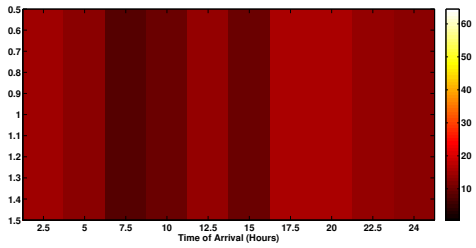

FIGURE 8.5. Example of a high entropic botnet group based on time of arrival

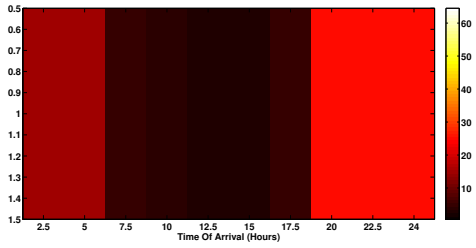

FIGURE 8.6. Example of a low entropic botnet group based on time of arrival

and spam bots, I conclude that they vary a lot in terms of their entropy value. Thus, one can easily group bots, spammers and legitimate from a given traffic analysis. Normal spamming patterns are quite random and, thus, they have a high entropy value, where as bots usually tend to have a similar structure of spamming a domain, which results it being

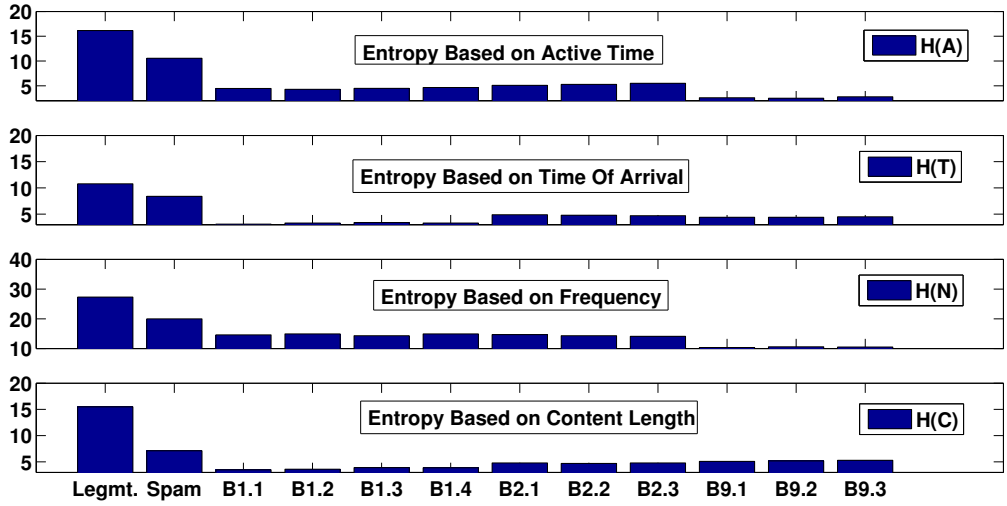

FIGURE 8.7. Figure shows entropy (randomness) values of legitimate, spammers and botnets

less entropic. For example, a low entropic botnet group is considered to have a similar structure of spamming in terms of its time of arrival, active time, frequency and content length. It is much easier to trace and identify low entropic bot in terms of its behavior pattern. In contrast, a high entropic bot having random patterns shows that it is changing its pattern of spamming regularly to avoid being traced. Such bots are difficult to trace in terms of their spamming pattern.

8.4. Principal-Factor Analysis to Determine the Relationship of the Randomness Levels

Principal components are successively chosen to have the largest possible variance, suppose the variance of the k_{th} PC is l_k, scree test involves looking at a plot (see figure 8.8) of l_k against k and deciding at which value of k the slopes of lines joining the plotted points are 'steep' to the left of k, and 'not steep' to the right. This value of k, defining an 'elbow' in the graph, is then taken to be the number of components m to be retained.

The scree graph defines a more-or-less straight line, not necessarily horizontal. The first point on the straight line is then taken to be the last component to be retained. If there are two or more straight lines formed by the lower eigenvalues, then the cut-off is taken at the upper end of the left-most straight line.

TABLE 8.1. Component matrix

	Component 1	Component 2
H(A)	0.648	-0.271
H(T)	-0.471	0.528
H(N)	0.478	-0.032
H(C)	-0.046	0.510

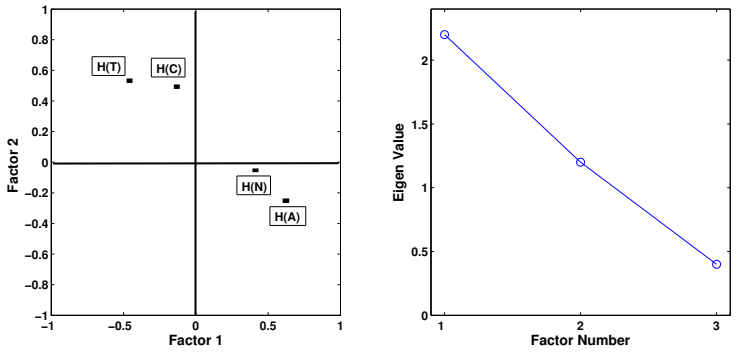

FIGURE 8.8. Principal factor plot: to determine the relationship between the variables; scree plot: to extract the components of maximum importance

Factor analysis, generally, is used to encompass both principal components and principal factor analysis. The eigenvalue for a given factor measures the variance in all the variables, which is accounted for by that factor as stated in table 8.1. If a factor has a low eigenvalue, then it is contributing little to the explanation of variances in the variables and may be ignored as redundant with more important factors.

Since the first and second factor are orthogonal i.e., uncorrelated, one can notice two established relations; (1) between entropy based on Time of arrival and content length (2) entropy based on frequency and active time. Looking at the orientation [see figure 8.8] of these variables towards the first and the second eigenvectors it detects a structural pattern of spamming which is going on between variables time of arrival and the content length of an email, and also between frequency of spamming and the active time. Using factor analysis I determine the trend noticed in figure 8.7.

Therefore, the final Randomness level (H) is given by Eq. 37.

$$H(X) = C_A H(A) + C_T H(T) + C_N H(N) + C_C H(C) \tag{37}$$

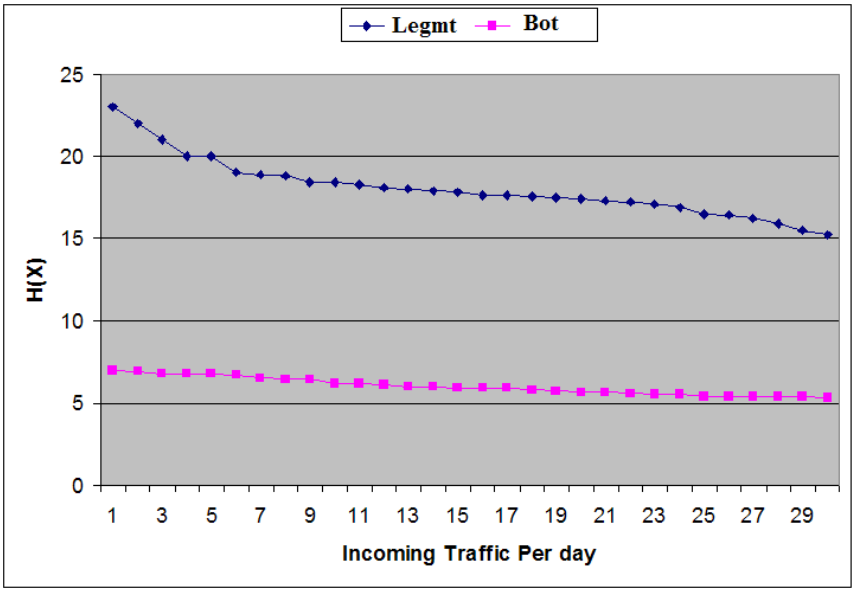

FIGURE 8.9. Entropy classification of spam bots and legitimate user

The final Randomness level is the sum of the product of the normalized randomness level (H') for a sender X, based on time of arrival, active time, content length, frequency and the contribution coefficients C_A, C_T, C_C, and C_N. These contribution coefficients are introduced here because I believe that these four parameters (Active time, time of arrival, content length and Frequency) have different contributions that impact the final randomness level. To obtain the values of these coefficients, PCA is applied. The values of these contribution coefficients are stated in Table 8.2. I use the above analysis to compute the final randomness of the traffic and further use their behavioral value to distinguish between legitimate and malicious traffic. I computed the final randomness for the entire corpus for 1 month of incoming traffic. I observe from figure 8.9 that the final randomness for legitimate traffic varies lot more than the randomness of botnet group. This proves the analysis that botnet share a similar behavior in trend over a period of time. One can notice that the slope of ranked $H(X)$ for Legitimate traffic is higher as compared to botnet spam, which shows that rate of change of randomness over a period of time is more in legitimate people than bot machines. This is an interesting result as one can distinguish the traffic into legitimate and botnet spam buckets based on their randomness values.

TABLE 8.2. Factor analysis results to final randomness

Contribution Coeff.(%)				Final Entropy (H)											
AT	TOA	Freq	Cl	Legt	Spam	B11	B12	B13	B14	B21	B22	B23	B91	B92	B93
30.7	33.3	17.3	18.7	16.8	11.4	5.5	5.8	5.9	5.9	7.9	7.7	7.9	3.7	4.2	3.4

TABLE 8.3. Precision of entropy based classification into buckets after handlabelling

Incoming Traffic	Accuracy								
	Legitimate			Spammers			Bots		
	T.P(%)	F.P(%)	F.N(%)	T.P(%)	F.P(%)	F.N(%)	T.P(%)	F.P(%)	F.N(%)
First Month	93	2	5	90	8	2	95	3	2
Second Month	95	1	4	95	4	1	95	3	2

8.5. Precision

To calculate the efficiency of the entropy based classification of spam botnets, I required a means of measuring the precision. So, I manually hand labeled the legitimate and spam data set and calculated the accuracy of the technique. I can observe from the relevant work [66, 67, 26] that bots possess a similar network Id and they may have same location of spamming, thus, because of the nature of bots, spamming usually inherits similar behaviors when spamming a domain. They have their spamming time very close to each other. There are few botnets which spoof their network id and they can have a new network id everyday, but their behavior of spamming still remains the same.

Table 8.3 contains handlabelling example of a particular corpus, which is divided in terms of first and second month of data. I have identified all the locations of the senders of a particular cluster, as in which country/city the sender is spamming plays an important role for hand labeling [refer to figure 8.10, 8.11, 8.12]. Then I analyzed his network id, also, by looking at the feature set I categorized it as in which bucket it falls into. Referring to table 8.3 I see that the accuracy of the filter, which includes entropy based classification, has precision of more than 95% to accurately identify botnet spam. My analysis also works well in identifying legitimate traffic as well with a high precision of 90%.

FIGURE 8.10. Geographic locations of the spam relays of a botnet group identified

FIGURE 8.11. Geographic locations of the spam relays of a botnet group identified

FIGURE 8.12. Geographic locations of the spam relays of a botnet group identified

8.6. Conclusion

Spammers are aggressive, smart and continue to use new techniques to send large volumes of spam [79] [80] [81] [82] [83]. The result is not only an enormous quantity of spam in users mailboxes, but also email servers that are brought down by excessive quantities of unwanted content using bots. This leads to an increasing investments in servers, network security administrators and software to keep up with the growing deluge of spam. Therefore, I propose a model that blocks spam based on spammers' readily identifiable behavior [84] [85] [86].

By applying a pre-filtering analysis of senders reputation, header based filters can operate much more efficiently, since the vast majority of spam is blocked before reaching email servers. However, traffic shaping does not block email, allowing real-time filtering techniques to be used without the risk of false positives. I propose a behavior based analysis technique to combat use of botnets to spam a particular domain. Using Information theory, it captured randomness based on variables such as time of arrival, content length, active time and frequency. A structural behavior pattern between the entropy values of these variables were captured using principal factor plot. Using factor analysis, I then computed the contribution coefficient of each variable towards final randomness, and then, using this total randomness value I distinguished legitimate and spam botnets into different buckets. I have a precision of more than 95% in capturing, classifying spam botnets using entropy.

Suppose a legitimate sender is identified as suspicious or bot based on his spamming pattern or when he/she is not on the receivers white mailing list, then that person will have to undergo the Turin test, which can be frustrating at times but the classifier has a precision of more than 90% in separating legitimate from spam traffic. Though I can have few false positives but wont let a spam bot pass through the classifier without going through the turing test.

To identify and mitigate VoIP bots is going to be the next challenging task. My future work will be concentrated more on identifying the structure of these VoIP spam bots, their possible exploitation in damaging VoIP network and how one should trace them based on their structural behavior. Also, I will be working on other traffic shaping techniques to reduce some legitimate delay, if any in future.

BIBLIOGRAPHY

[1] Cybercrime Primarily Originates in the U.S., Report Says,Symantec Corporation, *http://www.thenewsmarket.com/CustomLink/CustomLinks.aspx?GUID=053d4530-3ffd-4423-9089-ceae4ca598e9*, March 19, 2007.

[2] Microsoft security Bulletin ms04-011. http://www.microsft.com/technet/security/bulletin/ms04-011.mspx, Apr.2004.

[3] J. Mason, Spam Forensics: Reverse-Engineering Spammer Tactics, http://spamassassin.apache.org/presentations/2004-09-Toorcon/html/,Sept.2004.

[4] G. Keizer. Spam volume jumps 35. http://informationweek.com/news/showarticle.jhtml;?articleID=196701527.

[5] Message Labs. 2006: The year Spam raised its game and threats got personal, December 2006. http://www.messagelabs.com/publishedcontent/publish/about_us_dotcom_en/news__events/press_release/DA_174397.html.

[6] J. Rosenberg, "Programming IP Telephony Services with the Call Processing Language (CPL) and CGI," *OPENSIG*, 1999.

[7] D.M. Smith, M.C. Cain, L. Latham, B. Burton, "Develop a Strategy for Presence Technology," *Gartner Research*, 2006.

[8] D. Jiang, T.H. Yeap, L. Logrippo, R. Liscano, "Personaliztion for SIP Multimedia Communications with Presence," *IEEE ICSSSM*, 2005.

[9] V. Gehlot, A. Hayrapetyan, "A Formalize and Validated Executable Model of the SIP-Based Presence Protocol for Mobile Applications," *ACMSE*, Winston-Salem North Carolina USA, 2007.

[10] X. Shan, A. Shriram, "Enterprise Mobile Applications Based on Presence and Logical Proximity," *IWCMC*, Vancouver British Columbia Canada, 2006.

[11] Massachusetts Institute of Technology: Reality Mining. http://reality.media.mit.edu/.

[12] P. Graham, "Different Methods of Stopping Spam," http://www.secinf.net/anti_spam/Stopping_Spam.html, Oct. 2003.

[13] "Dealing Effectively with Spam", GFI Software, http://www.secinf.net/anti_spam/Dealing_Effectively_with_Spam.html, May 2003.

[14] "Blocking over 98% of Spam using Bayesian Filtering Technology," GFI Software, http://www.secinf.net/anti_spam/ Blocking_Spam_Bayesian_Filtering.html, Oct. 2003.

[15] "Spamfighting Overview FAQ," spamfaq.net, http://www.secinf.net/anti_spam/Spamfighting_Overview_FAQ.html, May 2003.

[16] B. Cormac O and C. Vogel, "Spam filters: Bayes vs. chi-squared; letters vs. words," In Proceedings of the International Symposium on Information and Communication Technologies, http://www.cs.tcd.ie/publications/tech-reports/reports.03/TCD-CS-2003-1%3.pdf, year =.

[17] T. Oda and T. White, "Developing an immunity to spam"' In Genetic and Evolutionary Computation-GECCO, Chicago, IL, USA. Lecture Notes in Computer Science, Vol. 2723, Springer, pages 231242. http://terri.zone12.com/doc/academic/spam gecco2003.pdf, 2003.

[18] "Increasing the accuracy of a spam-detecting artificial immune system," In Proceedings of the *Congress on Evolutionary Computation*, http://terri.zone12.com/doc/academic/spam cec2003.pdf, 2003.

[19] B. Cormac O and C. Vogel, "Comparing spamassassin with cbdf email filtering," In Proceedings of the 7th Annual CLUK Research Colloquium.

[20] O. Boykin P and V. Roychowdhury, "Personal email networks: An effective anti-spam tool," IEEE Computer, Vol. 38, April 2005.

[21] A. Ramachandran and N. Feamster, "Understanding the Network-Level Behavior of Spammers," In Submission.

[22] A. Kumar, V. Paxson, and N. Weaver, "Exploiting Underlying Structure for Detailed Reconstruction of an Internet-scale Event", In Proc. *ACM SIGCOMM Internet Measurement Conference,"* Berkley, CA, Oct. 2005.

[23] L. Fulu, M. Hsieh, "An Empirical Study of Clustering Behavior of Spammers and Group-based Anti-Spam Strategies," *CEAS*, 2006.

[24] H. Husna, S. Phithakkitnukoon, R. Dantu, "Behavior Analysis of Spam Botnets," *The 3rd International Conference on Communication System Software and Middleware (COMSWARE 2008)*, January 2008.

[25] H. Husna, S. Phithakkitnukoon, E. Baatarjav, and R. Dantu, "Traffic Shaping of Spam Botnets," *The 5th Annual IEEE Conference on Consumer Communications & Networking Conference (CCNC 2008)*, January 2008.

[26] A. Ramachandran, N. Feamster, D. Dagon, "Revealing Botnet Membership Using DNSBL Counter-Intelligence," *USENIX, SRUTI*, 2006.

[27] Osterman Research, Inc., "The Advantages of Using Traffic- Shaping Techniques to Control Spam," January 2007.

[28] N. Eagle and A. Pentland, "Eigenbehaviors:Identifying Structure in Routine," *MIT Media Laboratory.*

[29] B. Congleton and S. Nainwal, "Mining the Mine, Exploratory Social Network Analysis of the Reality Mining Dataset," *School of Information, University of Michigan, Ann Arbor*, Fall 2007.

[30] I.T. Jolliffe, *Principal Component Analysis*, 2nd ed. Springer Series in Statistics, 1986, New York, USA.

[31] S. Palla, "A Multi-Varaite Analysis of SMTP Paths and Relays to restrict spam and phishing attacks in Emails," Masters Thesis, Masters in Computer Science, *University of North Texas*, December, 2006.

[32] M. Turk, and A. Pentland, "Eigenfaces for Recognition," J. of Cognitive Neuroscience. Vol 3, Number 1., (1991) 71-86.

[33] M. McDermeit, R. Funk, M. Foss, M. Dennis, "Exploratory Factor Analysis with alpha method and varimax rotation," LI Analysis Training Series, 2000.

[34] A.K. Jain, M.N. Murty and P.J. Flynn, "Data Clustering:A Review," *ACM Computing Surveys*, Vol.31, No.3, September 1999.

[35] R. Dubes, A. Jain, *Algorithms for Clustering Data*, Prentice-Hall Advanced Reference Series, 1988.

[36] M.R. ANDERBERG, 1973. *Cluster Analysis for Applications.* Academic Press, Inc., New York, NY.

[37] E. DIDAY, AND J.C SIMON, 1976. Clustering analysis. In *Digital Pattern Recognition*, K. S. Fu, Ed. Springer-Verlag, Secaucus, NJ, 4794.

[38] V.L BRAILOVSKY, 1991. A probabilistic approach to clustering. *Pattern Recogn. Lett.* 12, 4 (Apr. 1991), 193198.

[39] C.T ZAHN, 1971. Graph-theoretical methods for detecting and describing gestalt clusters. *IEEE Trans. Comput.* C-20 (Apr.), 6886.

[40] J. MAO, AND A.K. JAIN, 1995. Artificial neural networks for feature extraction and multivariate data projection. *IEEE Trans. Neural Netw.* 6, 296317.

[41] P.H.A. SNEATH, AND R.R SOKAL, 1973. *Numerical Taxonomy.* Freeman, London, UK.

[42] B. KING, 1967. Step-wise clustering procedures. *J. Am. Stat. Assoc.* 69, 86101.

[43] J.H. WARD, 1963. Hierarchical grouping to optimize an objective function. *J. Am. Stat. Assoc.* 58, 236244.

[44] F. MURTAGH, 1984. A survey of recent advances in hierarchical clustering algorithms which use cluster centers. *Comput. J.* 26, 354359.

[45] R.A. BAEZA-YATES, 1992. Introduction to data structures and algorithms related to information retrieval. In *Information Retrieval: Data Structures and Algorithms,* W. B. Frakes and R. Baeza-Yates, Eds. Prentice- Hall, Inc., Upper Saddle River, NJ, 1327.

[46] G. NAGY, 1968. State of the art in pattern recognition. *Proc. IEEE* 56, 836862.

[47] W.H.E. DAY, 1992. Complexity theory: An introduction for practitioners of classification. In *Clustering and Classification,* P. Arabie and L. Hubert, Eds. World Scientific Publishing Co., Inc., River Edge, NJ.

[48] M.N. MURTY, AND G. KRISHNA, 1980. A computationally efficient technique for data clustering. *Pattern Recogn.* 12, 153158.

[49] G.H. BALL, AND D.J. HALL, 1965. ISODATA, a novel method of data analysis and classification. Tech. Rep.. Stanford University, Stanford, CA.

[50] C.E. Shanon, "A mathematical theory of communication," *Bell System Technical Journal,* vol. 27, pp.379-423 and 623-656, July and October 1948.

[51] S. Phithakkitnukoon, H. Husna, R. Dantu, "Behavioral Entropy of a Cellular Phone User," *The 1st International Workshop on Social Computing, Behavioral Modeling, and Prediction (SBP 2008).*

[52] A. Roy, S.K. Das, and A. Misra, "Exploiting information theory for adaptive mobility and resource management in future cellular networks," *IEEE Wireless Communications,* August 2004.

[53] IRONPORT INC. Spammers continue innovation. IronPort press release, June 28, 2006. http://www.ironport.com/ company/ironport_pr_2006-06-28.html

[54] V. MONGA, AND B.L. EVANS, Robust perceptual image hashing using feature points. In Procedings of the IEEE International Conference on Image Processing (ICIP04) (Oct. 2004), pp. 677 680.

[55] Messaging Anti-Abuse Working Group. MAAWG Issues First Global Email Spam Report. http://www.prnewswire.com/cgi-bin/stories.pl?ACCT=104&STORY=/www/story03-08-2006/0004316196, Mar.2006.

[56] T. MATHER, Net::DNSBLLookup Perl module. http://search.cpan.org/tjmather/ Net-DNSBLLookup-0.03/.

[57] K. Yeung, W. Ruzzo, "An empirical study on Principal Component Analysis for clustering gene expression data," *Bioinformatics*, November, 2000.

[58] R.B. Cattell, and S. Vogelmann, "A Comprehensive trial of the scree and KG criteria for determining the number of factors," *Mult. Behav. Res.*, vol.12, pp.289-325, 1977.

[59] C. Ding and X. He, "Cluster merging and splitting in hierarchical clustering algorithms," *IEEE International Conference on Data Mining (ICDM'02)*, Japan, 2002.

[60] H. Spth, "Cluster Analysis Algorithms for data reduction and classification of objects," Ellis Horwood, 1980.

[61] K. Xu, Z. Zhang, S. Bhattacharyya, "Profiling and Clustering Internet Hosts," *Sprint ATL Research Report RR05-ATL-020777*, Tech. Rep., February 2005.

[62] I.T, Jolliffe, B. Jones, T. Morgan, "Cluster analysis of the elderly at home: a case study," *Data analysis and Informatics*, pp.745757, 1980.

[63] H. Charles, *Cluster analysis for Researchers*, Lulu Press, 2004, North Carolina, USA.

[64] G. Erhard, Advances in System Analysis Vol. 4, Graphs as Structural Models: The Application of Graphs and Multigraphs in Cluster Analysis, Vieweg 1988.

[65] ZDNet Security News. Most spam genrated by botnets, expert says. http://news.zdnet.co.uk/internet/security/0, 39020375,39167561,00.htm, Sept. 2004.

[66] E. Cooke, F. Jahanian, D. McPherson, "The Zombie Roundup: Understanding, detecting, and disrupting botnets," *SRUTI Workshop*, July 7, 2005.

[67] D. Dagon, C. Zou, W. Lee, "Modeling Botnet Propagation Using Time Zones," *In Proceedings of the 13th Network and Distributed System Security Symposium NDSS*, February 2006.

[68] F. Tom, *Notes and practical considerations for researchers*, Kluwer Academic Publishers,The Netherlands, 2004.

[69] J. Erman, M. Arlitt, A. Mahanti, "Traffic Classification Using Clustering Algorithms," *SIGCOMM06 MineNet Workshop*, Pisa, Italy, September 2006.

[70] S. Zhao, Toward a taxonomy of copresence. Presence, 12(5):445455, October 2003.

[71] N. Eagle, and A. Pentland, "Reality Mining: Sensing Complex Social Systems," *Personal and Ubiquitous Computing*, 2006.

[72] A. Bar-Noy, I. Kessler, "Tracking mobile users in wireless communication networks," *IEEE Transactions* on Information Theory, 39(6): 1877-1886, November 1993.

[73] N. Eagle, A. Pentland, and D. Lazer, "Inferring Social Network Structure using Mobile Phone Data," *PNAS,* (In submission), 2007.

[74] A. Madan, A. Pentland(2006). "Vibephones: Socially Aware Mobile Phones, *ISWC06*, Martingy Switzerland Oct. 11-15 See TR602 http://hd.media.mit.edu

[75] A. Pentland, J. Gips, W. Dong, and W. Stoltzman, "Human Computing for Interactive Digital Media," MIT Media Technical Note No. 610, Proc. *ACM Multimedia,* Santa Barbara, CA, Oct 2006, pp.865-870.

[76] S. Wasserman and K. Faust, "Social network analysis," Cambridge: Cambridge University Press, pp. 188-191.

[77] W.J. Krzanowski, "Selection of Variables to Preserve Multivariate Data Structure, Using Principal Component Analysis," *Applied Statistics- Journal of the Royal Statistical Society Series C,* 1987, pp. 22-33 vol. 36.

[78] I. Cohen, Q. Tian, X. Sean, Z. Thomas and S. Huang, "Feature Selection Using Pricipal Feature Analysis,".

[79] S. Kandula, D. Katabi, M. Jacob, A. Berger, "Botz-4-Scale: Surviving Organized DDOS Attacks That Mimic Flash Crowds," in the proceeding of *USENIX NSDI*, May 2005.

[80] S. Katti, B. Krishnamurthy, D. Katabi, "Collaborating Against Common Enemies," in the proceeding of *ACM* IMC05, Oct. 2005.

[81] J. Synoradzki, P. Wawrzyniak, M. Zmudzinski, "Four Popular Anti-Spam Filters for Exchange Reviewed," http://www.secinf.net/anti_spam/Preventing_Spam _Antispam_Filters_MS_Exchange.html, May 2004.

[82] B. Laurie, R. Clayton, "Proof-of-Work Proves Not To Work," in the proceeding of the *Workshop on Economics and Information Security,* MN, May 2004.

[83] J. Graham-Cumming, "Tricks of the Spammer's Trade," in Hakin9 magazine, issue 3, 2004.

[84] J. Jung and E. Sit, "An Empirical Study of Spam Traffic and the Use of DNS Black Lists", in the proceeding of *ACM* IMC 04, Oct. 2004.

[85] L. H. Gomes, C. Cazita, "Characterizing a Spam Traffic," in the proceeding of IMC' 04, Oct. 2004.

[86] "Know Your Enemy: Tracking Botnets", the Honeynet Project and Research Alliance, http://www.honeynet.org, March 2005.

[87] H. Husna, S. Phithakkitnukoon, E. Baatarjav, and R. Dantu, "Quantifying Presence using Calling Patterns," *The 3rd International Conference on Communication System Software and Middleware (COMSWARE 2008)*, January 2008.

[88] H. Husna, S. Phithakkitnukoon, E. Baatarjav, and R. Dantu, "A study of Social Network Structure on Facebook," *ACM SIGCOMM 2008.* In submission.

[89] R. Dantu, H. Osterwijk, P. Kolan and H. Husna, "Securing Medical Networks," *Network Security,* June 2007.

[90] S. Biswas, R. Morris, "ExOR: Opportunistic Multi-Hop Routing for Wireless Networks," in the proceeding of *ACM SIGCOMM 05*, Philadephia, USA, Aug. 2005.