BOTNET DETECTION AND OPTIMIZATION USING HONEYPOT

A Major Project Report submitted in the partial fulfillment
of the requirements for the award of the degree of
MASTER OF TECHNOLOGY
(INFORMATION SYSTEMS)

Submitted By:
PRAKHAR SRIVASTAVA
(Roll No. 2K13/ISY/16)

Under the esteemed guidance of
Dr. N.S. RAGHWA
Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
DELHI TECHNOLOGICAL UNIVERSITY

BAWANA ROAD, DELHI-110042

SESSION: 2013-2015

CERTIFICATE

This is to certify that work entitled "**Botnet Detection And Optimization Using Honeypot**" submitted by **Prakhar Srivastava (2k13/ISY/16)**, to Delhi Technological University, Delhi for the award of the degree of Master of Technology is a bonafide record of research work carried out by him under my supervision.

The content of this thesis, in full or parts have not been submitted to any other institute or university for the award of any degree or diploma.

Dr. N.S.Raghava

Project Guide

Assistant Professor

Department of Computer Science and Engineering

Delhi Technological University

Shahbad Daultpur, Bawana Road, Delhi-110042

ACKNOWLEDGEMENT

I would like to thank my project guide, **Dr. N.S. Raghwa** for his valuable guidance and wisdom in coming up with this project. I humbly extend my words of gratitude to **Dr. O. P. Verma**, Head of Department, and other faculty members of IT department for providing their valuable help and time whenever it was required. I thank all my friends at DTU who were constantly supporting me throughout the execution of this thesis.

Special thanks to the Almighty Lord for giving me life and the strength to persevere through this work. Last but not least, I thank my family for believing in me and supporting me.

Prakhar Srivastava

Roll No. 2k13/ISY/16

M.Tech (Information Systems)

E-mail: prakhar42@gmail.com

Department of Computer Science and Engineering

Delhi Technological University

ABSTRACT

Botnet is the most extensive and thoughtful threat which follows commonly in today's cyber-attacks. A botnet is a collection of bargained computers which are at all controlled by hackers to launch various network attacks, such as Botnet attack, spam, click fraud, identity theft and information phishing. Botnet has developed a popular and creative tool overdue many cyber-attacks. The important characteristic of botnets is the use of command and control channels through which they can be efficient and absorbed. Lately malicious botnets change into HTTP botnets out of typical IRC botnets. This makes the detection of botnet command and control a stimulating problem. The Thesis then classifies and inspects problems common to many current packet sniffing applications, shows how these problems can effortlessly undermine the network administrator's intents and lead to a false intellect of security, and proposes solutions to these problems. Lastly, in this thesis accomplishes that botnet identification number follow by support vector classifier (SVM) that would distribute on the basis of header length such as h1, h2 and h3 although h3 would combine balance packet for h1 and h2 which currently a viable network security mechanism, but that its utility could be greatly improved with the extensions proposed in the Thesis.

LIST OF FIGURES

LIST OF TABLES

TABLE OF CONTENTS

Title **Page No.**

CHAPTER 1

INTRODUCTION

1. INTRODUCTION

1.1 OVERVIEW OF BOTNET

One of the most insidious cyber threats for security community is represented by diffusion of botnets, networks of infected computers (bots or zombies) managed by attackers due the inoculation of malware. The controller of a botnet, also known as botmaster, controls the activities of the entire structure giving orders through communication channels; the use of botnets is very commons in various IT contexts, from cybercrime to cyber warfare.
A botnet could be used to conduct cyber-attacks, such as a DDoS, against a target or to conduct a cyber-espionage campaign to steal sensitive information. There are various classifications of botnets, it's possible to discriminate them from the architecture implemented, the used network protocol or technology on which they are based. Bots are also called \zombies" because a computer (infected with a bot) performs a task given by its master.

A botnet herder (also called a botnet master) is a person or a group, who control the whole botnet: they can give instructions or upload data to the botnet. Botnets are used to perform a wide variety of tasks but some of the most popular ones are sending spam or coordinating a DDoS 1 attack. As the traditional" botnets are being shutdown (as IRC botnets can be easily detected because they have a single point of failure), the newer and more dangerous botnets are moving to more resilient architectures. The newer generation botnets are also using new techniques to hide their botnet and their tracks and are using a more sophisticated encryption method. This will make them ever harder to track and fight.

As we will see in the post the diffusion of botnet is increased due various factors such as the availability of unprotected mobile platforms and the presence in the underground market of cyber criminals that rent services and structures to compose the malicious systems. Infected machines receive commands from Command & Control (C&C) servers that instruct the overall architecture to operate to achieve the purpose for which it has been composed such as creation of SMTP mail relays for targeted spam campaign, implementation of a fraud scheme (e.g. Banking information gathering) or to launch a denial of service attack.
This thesis aims to research botnets and other related threats in order to develop an application designed to detect the possible activities of a botnet and, particularly, how it can be detected at an early stage.

Along with this the thesis will analyze the possible data remnants that a botnet agent can leave on a machine, so that it can be analyzed for future use.
Bots have been around since the late 90's and they were originally created to perform automated repetitive tasks. One of the main uses was on IRC chat servers providing statistical and administrative functions. Search engines, such as Google, also use bots to find new and updated websites making it possible to find the most up-to-date information easily. It has only been in the last few years that bots have become a tool used by criminals for malicious purposes such as to extract credit card, banking and other information for financial gain.

Bots accomplish this by running ambiguously on a user's computer. The bot is part of a collection of bots called a Botnet. Botnets can be made up of thousands, potentially millions of bots which are typically controlled by a bot master. They command the bots to perform various tasks from delivering spam to stealing banking account information. Botnets are getting more and more sophisticated and have infiltrated millions of personal and business computers including almost all of the Fortune 500 companies. The latest development is the discovery of the Stuxnet Botnet. It has concerned many security experts and has been called "groundbreaking" because it is so sophisticated. It appears to have been design specifically to disrupt power grids in Iran. One of the targets is the Iranian nuclear power stations. Many speculate that it must have been created with government backing such as Israel or the USA .

Botnet is a collection of internet-connected computers whose security defenses have been breached and control ceded to a malicious party, blackhat community. The concept of botnet refers to a group of compromised computers remotely controlled by one attacker or a small group of attackers working together called a „„botmaster". These large groups of hosts are assembled by turning vulnerable hosts into so-called zombies, or bots, after which they can be controlled from afar. A collection of bots, when controlled by a single command and control (C2) infrastructure, form what is called a botnet. The botmaster□s ability to carry out an attack from hundreds or even tens of thousands of computers means increased bandwidth, increased processing power, increased memory for storage and a large number of attack sources making botnet attacks more malicious and difficult to detect and defend against. Computer networks and the Internet, by their nature, support the free flow of information; nevertheless we have thousands of reasons to break this rule. Even for publishing website on the internet, we may not want everybody to read that. For small businesses or individuals, we can restrict our web contents accessible only to authorized users by deploying firewall or AAA server, but the simplest and cheapest solution to this general concern might be installing a traffic filter on the IIS web server.

In many circumstances, HTTP server can act as a portal to the internal network, thus packet filter even provides a basic level of security for controlling access to internal network.

1.2 PROBLEM IDENTIFICATION

In pure SVM there is some room for optimization, they are as follows

• Optimize binary SVM classification rules,

• Train conventional linear classification SVMs optimizing error rate in time that is linear in the size of the training data through an option, but corresponding formulation of the instruction problem [9]. This could be much faster than pure SVM for large training sets. In the previous IDS pure SVM is used, if we will use modified SVM with above mention point can improve overall performance of IDS.

1.2.1 Command and Control Server

"Command and Control" (C&C) servers are centralized machines that are able to send commands and receive outputs of machines part of a botnet. Anytime attackers who wish to launch a DDoS attack can send special commands to their botnet's C&C servers with instructions to perform an attack on a particular target, and any infected machines communicating with the contacted C&C server will comply by launching a coordinated attack.

Botnet C&C servers often exist in one of four structures each with pros and cons: star, multi-server, hierarchical, and random:

- **Star topology** botnets rely on one central C&C server, which sends commands to every bot in the botnet. This configuration allows for reliable, low-latency communication, but renders the botnet fairly easy to disable, as there is only one C&C server to take offline before the botnet is inoperable.
- **Multi-server** topology botnets are very similar to star topology botnets, except that the central "server" consists of a series of interconnected servers that allow for redundancy (preventing the single point of failure problem of star topology botnets); however, setting up multiple connected C&C servers may require more planning and overall be more difficult than just using a single server.
- **Hierarchical** topology botnets (involving a series of C&C servers in a hierarchy) allow for botnet owners to more easily divide their botnet up into "separate" chunks for re-sale or renting, as well as prevent researchers from enumerating the location of all other C&C servers and bots within a network with only a few captured C&C servers due to the restricted visibility of the entire botnet from lower hierarchy certain

servers. Additionally, commands that have to travel through a large hierarchy of C&C servers in order to reach bots may add to latency.

- **Attacker:** initiates and controls the attack Master: compromised host that controls the agents and invokes commands
- **Agent:** bots that perform P2P attacks on victims based on commands invoked by master

The contribution of our work is as follows:

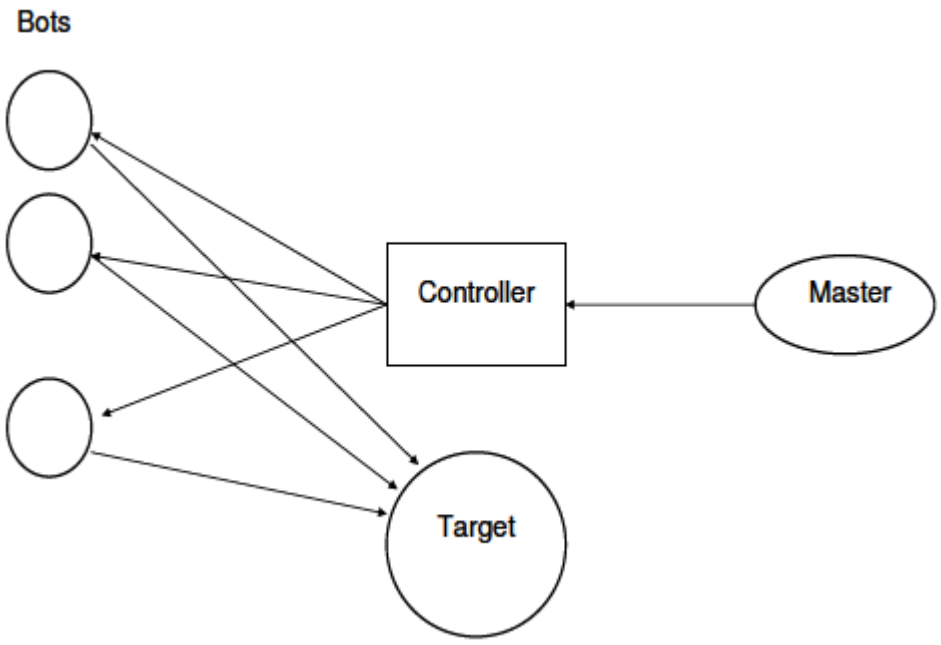

Figure 1.1 Bots and Target

- A two pronged approach towards detecting Botnets. An algorithm that based on standalone host activity analysis triggers a network analysis to find Botnets in the given network.
- A method for identifying similarities between filtered Bot traffic using Dynamic Time Warping (DTW) algorithm, K-means clustering and graphical analysis.
- We also present an experimental simulation of UDP flood attack and perform analytical calculations on UDP packet flows retrieved from the same, based on the Dynamic Time Warping (DTW) algorithm.

- Our solution does not assume the existence of a particular type of Bot and is hence generic. It can deal with both C&C and P2P Bots because of our two pronged approach. The advantage of our solution is that it can evolve to identify new Bot patterns thus

making it a learning based approach. The chances of false positives are also reduced because of the two pronged strategy adopted. The rest of the Thesis is structured as follows: Section 2 explains the data structure that we use for network traffic flow collection, Sections 3, 4 explain our solution which comprises of Stand Alone and Network Algorithm. These sections also include required experimental analysis. In Section 5 we summarize our work followed by the future work and references in Section 6.

1.3 RESEARCH OBJECTIVES

The final aim of the thesis is to develop a system which can detect the activity of botnet .
To meet this aim, the following five main objectives must be met:

1. There have to panel such as admin and user.
2. Admin detect the activity of all users PC and he will get the activity of the other system.
3. Design an agent host-based system that will be able to successfully detect a bot on a machine.
4. Admin will detect the no of packet transfer to other system using distributed environment.
5. Implement and evaluate the detection software created in objective 2.
6. Research into botnet taxonomy, current botnets and related threats. As well as detection methods and evaluation techniques for testing the proposed detection application.

1.4 RESEARCH METHODOLOGY

1.4.1 SVM (support vector machine)

A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyper plane. In other words, given labeled training data (supervised learning), the algorithm outputs an optimal hyper plane which categorizes new examples. For a linearly separable set of 2D-points which belong to one of two classes, find a separating straight line.

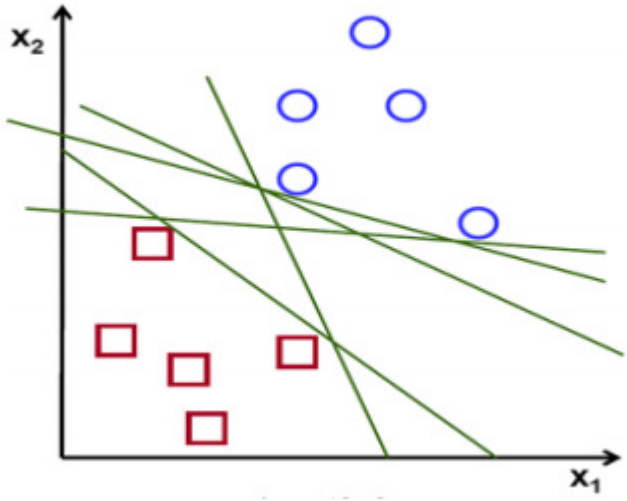

Figure 1.2 Multiple solutions to one problem

In the above picture you can see that there exists multiple lines that offer a solution to the problem. If any of them better than the others, we can intuitively define a criterion to estimate the worth of the lines: A line is bad if it passes too close to the points because it will be noise sensitive and it will not generalize correctly. Therefore, our goal should be to find the line passing as far as possible from all points. Then, the operation of the SVM algorithm is based on finding the hyper plane that gives the largest minimum distance to the training examples. Twice, this distance receives the important name of margin within SVM's theory. Therefore, the optimal separating hyper plane maximizes the margin of the training data.

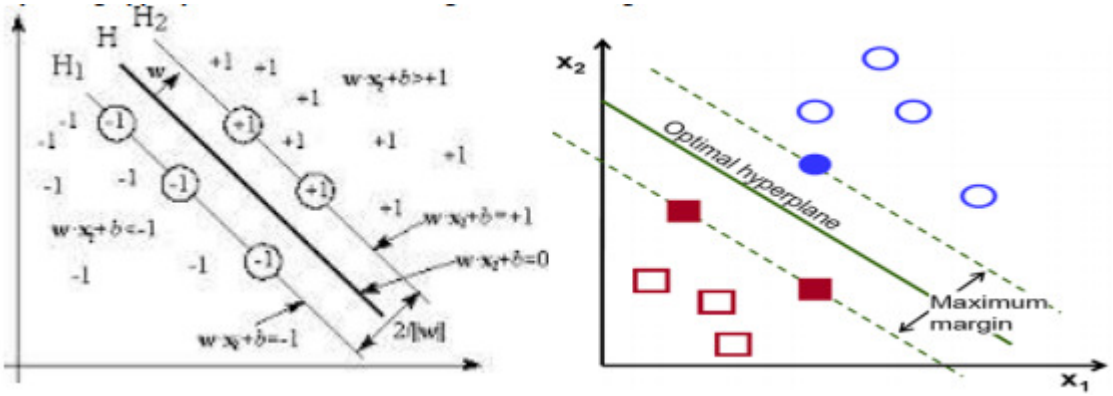

Figure 1.3 Hyperplane

1.5 THESIS ORGANIZATION

This thesis is split into five main chapters. They are described as follows:

1. **Chapter 1** – Introduction: This chapter provides the thesis overview and background to the subject of botnets. The key aim and objectives of the thesis are also defined along with the thesis structure.

2. **Chapter 2** – Literature Review: This chapter identifies trends and provides taxonomy of Botnets that currently exist. It will also investigate detection methods, such as Intrusion system by administrator panel and he having the right to detect all activity.

3. **Chapter 3** – Design: Based on the findings found in the literature review, this chapter introduces a design for the botnet detection software together with the synthetic bot that will be required for the evaluation to be carried out on the detection software.

4. **Chapter 4** – Methods: This chapter will implement and document the introduction of the detection software.

5. **Chapter 5 –** Result: This chapter will use an evaluation technique researched in the Literature Review to determine the performance of the detection software.

6. **Chapter 6** – Conclusion: This chapter will conclude the thesis reviewing the aims and objectives and provide a critical analysis and comment on possible future work.

CHAPTER 2

LITERATURE SURVEY

2. LITERATURE SURVEY

Barford et al. [1] present an in-depth analysis of Bot software source code. They reveal the complexity of Botnet software, and discuss implications for defense strategies based on the analysis. BotTracer [21] detects three phases of Botnets with the assistance of virtual machine techniques. Three phases include the automatic startup of a Bot without requiring any user actions, a command and control channel establish with its Botmaster, and local or remote attacks. **Binkley et al. [2]** propose an anomaly-based algorithm for detecting IRC-based Botnet meshes. Using an algorithm, which combines an IRC mesh detection component with TCP scan detection, they can detect IRC Botnet channel with high work weight hosts.

Ramachandran et al. **[25]** develop techniques and heuristics for detecting DNSBL reconnaissance activity, whereby Botmaster perform lookups against the DNSBL to determine whether their spamming Bots have been blacklisted. This approach is derived from an idea that detects DNSBL reconnaissance activity of the Botmaster but it is easy to design evasion strategies.

Zhuang et al. [31] develop techniques to map Botnet membership using traces of spam email. To group Bots into Botnets they look for multiple Bots participating in the same spam email campaign. They apply the technique against a trace of spam email from Hotmail web mail services.

Karasaridis et al. [19] propose an approach using IDS-driven dialog B correlation according to a defined Bot infection dialog model. They combine heuristics that assume the network flow of IRC communication, scanning behavior, and known models of Botnet communication for backbone networks.

BotHunter [12] models the Botnet infection life cycle as sharing common steps: target scanning, infection exploit, binary download and execution, command-and-control channel establishment, and outbound scanning. It then detects Botnets employing IDS-driven dialog correlation according to the Bot infection life-cycle model. Malwares not conforming to this model would seemingly go undetected.

BotSniffer [13] is designed to detect Botnets using either IRC or HTTP protocols. BotSniffer uses a detection method referred to as spatial-temporal correlation. It relies on the assumption that all Botnets, unlike humans, tend to communicate in a highly synchronized fashion. BotSniffer has a similar concept with BotGAD in respect of capturing the synchronized Botnet communication. Different from BotGAD,

BotSniffer performs string matching to detect similar responses from Botnets. Botnet can encrypt their communication traffic or inject random noise packets to evade.

BotMiner [11] presents a Botnet detection method which clusters Botnet□s communication traffic and activity traffic. Communication traffic flow contains all of the flows over a given epoch including flows per hour (fph), packets per flow (ppf), bytes per packet (bpp), and bytes per second (bps). The activity traffic identifies hosts which are scanning, spamming, and downloading any Portable Executable binary. Clustering algorithms are applied and performed cross-plane correlation to detect Botnets. Hence this section some existing approach to detect Botnet specially HTTP Botnet. Now in next section we have to present our proposed framework to detect HTTP Botnet using AIS.

Dredze et al .[8] proposes a new and simple methodology to detect phishing emails utilizing Confidence-Weighted Linear Classifiers. They use the contents of the emails as features without applying any heuristic based phishing specific features and obtain highly accurate results compared to the best that have been published in the literature. Phishing is a criminal mechanism employing both social engineering and technical subterfuge to steal consumers' personal identity data and financial account credentials. Dredze et al. recently proposed confidence weighted linear classifiers (CWLC), a new class of online learning method designed for Natural Language Processing(NLP) problems based on the notion of parameter confidence. Online learning algorithms operate on a single instance at a time, allowing for updates that are fast, simple and make few assumptions about the data, and perform well in wide range of practical settings. Online algorithm processes its input piece-by-piece in a serial fashion, i.e., in the order that the input is fed to the algorithm, without having the entire input available from the start.

Lee et.al [7] in his Thesis, for spam detection, proposed parameter optimization and feature selection to reduce processing overheads with guaranteeing high detection rates. In previous Thesiss, either parameter optimization or feature selection, but not both. Parameters optimization is to regulate parameters of spam detection models to figure out optimal parameters of the detection model. Feature selection is to choose only important features or feature set out of all the features. Feature selection enables to eliminate irrelevant features to avoid processing overheads.

Razmara et.al in [4] his work, present a novel solution toward spam filtering by using a new set of features for classification models. These features are the sequential unique and closed patterns which are extracted from the content of messages. After applying a term selection method, we show that these features have good performance in classifying spam messages from legitimate messages. The achieved

results on 6 different datasets show the effectiveness of our proposed method compared to close similar methods. Authors outperform the accuracy near +2% compared to related state of arts. In addition this method is resilient against injecting irrelevant and bothersome words. This method is outlined as the following steps:

- Preprocessing and stemming datasets
- Selecting best discriminating terms based on a term selection method
- Looking for frequent sequential patterns in corpus
- Using patterns as features
- Feature selection and classification The vector model for representation of texts has been offered in Salton's works. In the elementary case, the vector model assumes comparison to each document of a frequency spectrum of words. The dimension of space is reduced by rejection of the most common words that increases thereby percent of the importance of the basic words in more advanced vector models. The possibility of ranging of documents according to similarity in vector space is the main advantage of vector model. Applied Computational Intelligence and Soft Computing Clustering is one of the most useful approaches in data mining for detection of natural groups in a data set. The up-to-date survey of evolutionary algorithms for clustering.

M. Mangalindan et al [3]., proposed a complex-network, which is based on SMS filtering algorithm that compares an SMS network with a phone- calling communication network. Although such comparison can provide some new features, that obtains well-aligned phone-calling networks and SMS networks that can be aligned perfectly is difficult in practice. In this Thesis, we present an effective SMS anti-spam algorithm that only considers the SMS communication network. We first analyze characteristics of the SMS network, and then examine the properties of different sets of meta-features including static features, temporal features and network features. We incorporate these features into an SVM classification algorithm and evaluate its performance on a real SMS dataset and a video social network benchmark dataset. We also compare the SVM algorithm to a KNN based algorithm to reveal the advantages of the former. Our experimental results demonstrate that SVM based on network features can get 7%-8% AUC (Area under the ROC Curve) improvement as compared to some other commonly used features.

Borg et.al in [26]his Thesis presents a method that can use several social networks for detecting spam and a set of metrics for representing OSN data. The Thesis investigates the impact of using social network data extracted from an E-mail corpus to improve spam detection. The social data model is compared to traditional spam data models by generating and evaluating classifiers from both model types. The results in this Thesis show that accurate spam detectors can be generated from the low-dimensional social data

model alone, however, spam detectors generated from combinations of the traditional and social models were more accurate than the detectors generated from either model in isolation. Online Social Networks (OSNs) contain more and more social information, contributed by users. OSN information may be used to improve spam detection.

McCord et.al [27]in his Thesis discuss some user-based and content-based features that are different between spammers and legitimate users. These features are then used to facilitate spam detection. Using the API methods provided by Twitter, they crawled active Twitter users, their followers/following information and their most recent 100 tweets. Then, detection scheme is evaluated based on the suggested user and content-based features.

Zhang et al., [28]2008 describes a genetic programming approach to feature extraction for a cost-sensitive classification task of spam. The fitness used comprised three objectives: an approximation to the Bayes error, misclassification cost and number of tree nodes used to encode a particular solution. The solution proposed in (Zhang et al., 2008) is the most analogous to the one presented in this Thesis, since an EA is used for the feature selection.

Dudley et al., [29]2008 proposed an EA to analyse different configurations for Spam Assassin, a widely-used open source spam filter. Their approach consisted in using an EA to achieve an optimal setup, at a personalized level, for the set of weights that is used to infer if a given message is spam. In this case, the EA minimized the number of false positives and false negatives.

Araujo et.al [30] in his Thesis, present an efficient spam detection system based on a classifier that combines new link-based features with language-model (LM)-based ones. These features are not only related to quantitative data extracted from the Web pages, but also to qualitative properties, mainly of the page links. He considers, for instance, the ability of a search engine to find, using information provided by the page for a given link, the page that the link actually points at. This can be regarded as indicative of the link reliability. He also checks the coherence between a page and another one pointed at by any of its links. Two pages linked by a hyperlink should be semantically related, by at least a weak contextual relation. Thus, he applies an LM approach to different sources of information from a Web page that belongs to the context of a link, in order to provide high-quality indicators of Web spam. They have specifically applied the Kullback–Leibler divergence on different combinations of these sources of information in order to characterize the relationship between two linked pages. The result is a system that

significantly improves the detection of Web spam using fewer features, on two large and public datasets such as WEBSPAM-UK2006 and WEBSPAM-UK2007.

Lee et.al in [20] his Thesis, for spam detection, planned parameter optimization and has choice to cut back process overheads with guaranteeing high detection rates. In previous Thesiss, either parameter optimization or feature selection are used, however not each. Parameter optimization could be a method that regulates parameters of spam detection models to work out optimum parameters of the detection model. Feature selection could be a method that chooses solely necessary options or feature commenced of all the options. Feature selection allows eliminating orthogonal options to avoid process overheads.

Wan et.al [22] in his Thesis proposed a spam detection method that uses Sobel operators for edge detection and a multiple filter using Sobel operators and OCR. As spam filters easily catches text, so spam senders uses images to send spam instead of text. Traditional image spam filters have weaknesses in scanning documents and photographs. However, to transmit information, text is always used in image spam. Therefore, in this study, author classifies mail images by the configuration of letters and images.

A. PREVAILING ANALYSIS AND DETECTION TECHNIQUES

[23] **Kirti Mathur in April**, 2013 highlighted the prevailing analysis and detection techniques that are used for obfuscated malicious code. The major threat to computer system security is various malware that do the malicious actions. AV Scanners, Intrusion Detection System, and Firewalls are the various solutions that are used to detect these threats. Traditionally, all of these solutions for detection of malware detect their presence in our system by using malware signatures. But malware authors employ some obfuscation methods for which these methods proved unsuccessful.

B. THREE DATA MINING ALGORITHMS TO PRODUCE NEW CLASSIFIERS

[24] Milan Jain in August, 2014 proposed three data mining algorithms to produce new classifiers with separate features. The three algorithms are RIPPER, Naïve – Bayes and a Multi Naïve Bayes Classifier. The author also compared these three algorithms. Three phases comprising it is root kit data collection, pre-processing of data, then its classification and evaluation of performance. With the growth in high-speed Internet connections, malware are spreading very rapidly. Therefore, it is very essential to detect and delete benign malware in an effective way.

C. DEFINITION, TYPES, PROPAGATION OF MALWARE, AND THEIR DETECTING TECHNIQUES

[14] Mohsen Damshenas, Ali Dehghantanha, Ramlan Mahmoud in 2013 closely looked into the concept of malware, to know the definition, types, propagation of malware, and their detecting techniques so as to enhance the method of protection and security. The security experts practice all promising techniques, strategies and methods to halt and eliminate the threats whereas the malware authors exploit new types of malwares that bypass employed security features.

D. SEVERAL MALWARE DETECTION METHODS

[15] Vinod P. focused on several malware detection methods like signature based detection methods, reverse engineering of obfuscated code, for detecting malicious codes. Malwares are malicious software's. They are intended to harm computer systems devoid of the knowledge of the owner of the system. Software's that came from trustworthy vendors also contain malicious code which disrupts the system and discloses private information to remote servers. Malware's consist of computer viruses, spyware, ad-ware, Trojans etc. [16] Nwokedi Idika and Aditya P. Mathur in February 2007 had examined 45 malware detection methods and provided a chance to compare them with one another which would help in the process of decision making involved in the development of a secure application. The survey also provided a comprehensive bibliography to assist the researchers in malware detection. Malware detectors are the chief tools that provide protection against malware. The technique used by such malware detectors determines their effectiveness. Therefore it is very important to study malware detection techniques and recognize their merits and demerits.

E. CLASSIFICATION OF MALICIOUS EMAILS USING NAÏVE BAYES CLASSIFIER

[17] B.V.R.R.Nagarjuna in July 2013 explained how the malicious emails are classified and how these are deleted and how to know the contents of messages. The author used Bayesian spam filtering, Email filtering and J48 process for classification which overcomes the difficulties arose in linear C-Support Vector Machine (C-SVM). This machine had given the correct results when compared to the existing one. There are three steps to examine first one is to detect the malicious email, next step is to apply classifier to classify according to the emails received and send to trash automatically and delete directly.

[18] Ion Androutsopoulos, John Koutsias, Konstantinos, Constantine D. Spyropoulos and V. Chandrinos in Aug, 2000 proposed a method in which a Naïve Bayesian classifier is automatically trained to identify spam messages. The author tested this method on a large set of personal e-mail messages, which are available in encrypted form publicly. The author presented proper cost-sensitive measures. The author also examined the influence of training data size, lemmatization, size of attribute set, stop lists and the issues that had not been explored till then. Lastly, the Naive Bayesian classifier is compared to a filter makes use of keyword patterns to find its effectiveness.

F. Data mining applications and several classification algorithms

[5] Vishnu Kumar Goyal in April, 2014 had worked with diverse data mining applications and several classification algorithms. The algorithms had been applied on different dataset to find out the effectiveness of the algorithms. Classification is an important technique of data mining. It has wide applications in classifying the various kinds of data used in almost every field of human life. The author analysed the five main classification algorithms: Decision Tree (DT), Decision Stump (DS), k-nearest neighbourhood (KNN), Naive Bayes (NB) and Rule Induction (RI) and compared their performance. The results were verified on five datasets namely Golf, Iris, Weighting, Deals and Labor using Rapid Miner Studio.

Technique	Approach	Based on	Effective in	Pros	Cons	Accuracy
Signature Based Detection	Actively compare and match current behaviour against a large collection of signatures.	Predefined rules for known attacks	Detection of attacks having a fixed behaviour pattern.	Signature can be used as a standalone system. Works well for known signatures.	Signature extraction and distribution is a complex task. The signature generation involves manual intervention. The growing size of signature repository.	Lower false alarm rates Low false positives
Behavior Based Detection	identifies the action performed by Malware. Single behaviour signature can identify various samples of malware.	Signatures of malicious behavior	Detection of mutants of malware	May detect a wide range of novel attacks Can be cheap to deploy and monitor	Usage patterns may change often. Post-facto, attack already occurred Easy to evade once known	Higher false alarm rates Low false positives
Anomaly Based Detection	Detects behaviors that fall outside the predefined or accepted model of behavior.	Notion of normality	Detection of new unknown attacks	Can detect potentially a wide range of novel attacks.	May miss known attacks High overhead	High false alarm rates High false positives
Specification Based Detection	Leverage some specification or rule set of what is valid behavior. Programs violating the specification are considered malicious.	Statistical machine learning	Detection of attacks having fixed specification.	Approximates the requirements of application or system. Address the high false alarm rate problem.	it is very difficult to accurately specify the behaviour of the system or program.	Low false alarm rate High false positives

Table 2.1: Comparison of various malware detection techniques

CHAPTER 3

BOTS AND BOTNETS

BOTS AND BOTNETS

3.1 BOTS AND IRC HISTORY

IRC stands for Internet Relay Chat, which provides a way of communication with connected users in a real time [84][113][159][168]. It is mainly designed for group (many-to-many) communication in discussion forums called channels. In addition, IRC allows uni-cast communication [118]. The user can monitor a conversation between multiple users and can participate in the conversation. IRC was created in late August 1988 by Jarkko Oikarinen to replace a program called MUT (Multi-User Talk) on a BBS called OuluBox in Finland and to allow a maximum of 100 users to communicate concurrently [84][118][168]. In 1993, the first IRC protocol was defined by RFC1459. Later on, it was updated to include RFC2810, RFC2811, RFC2812, and RFC2813 [79].

IRC Operations Once a user connects to the IRC server, s/he can join a channel where other users are already there as shown in Figure 2.1. If s/he is the first user who joins the channel, s/he will be the channel operator. Any user submits a message to the server publicly, the other users can see her/his message on the channel [84][113][26].

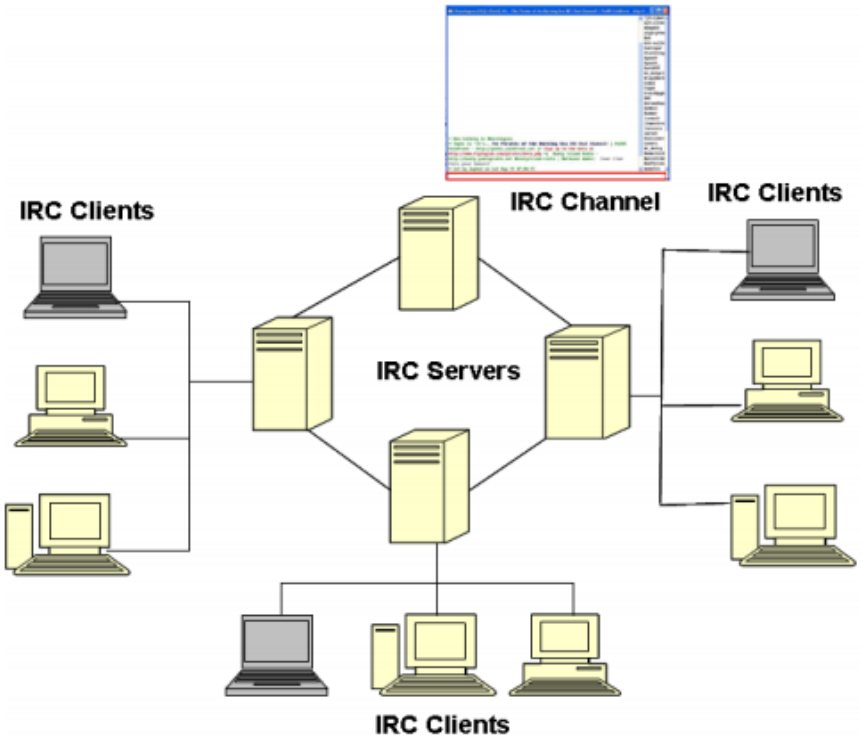

Figure 3.1: Basic IRC Operation

IRC has multiple channels. Therefore, if any user wishes to join a channel s/he should have the IRC client, know the IP server and also should know the channel name. A user can connect to IRC server through predefined ports (6660-7000/tcp). The most common port of the IRC server is 6667/tcp. Each channel has one or more operators to manage the channel. The channel operator's privileges can be obtained by one of the following methods. The first method is that the user creates that channel and becomes its operator. The second method to obtain operator privileges is through an Approved Channel Operator (AOP) list. In addition to sharing text messages between users on the channel, IRC has other functionalities. For example, IRC can allow file transfer between users, execute peer-to-peer capabilities, and run an automated program, termed 'a bot', to monitor IRC channels. Moreover, IRC hasmany commands that can be used by users. In this section, we will define the most common commands, which are used by users. • NICK and USER: are used to label a user and user's host equivalent to an ID. • PASS: set or send a password. • JOIN: enter a channel, often secret. • MODE: modify channel settings (eg. invisible). • PING and PONG: maintain the connection to IRC server. • PRIVMSG: send a message to a channel or user. • DCC SEND: transfer files from one user to another.

3.2 TYPE OF BOTNETS

There are many types of botnet; hub-leaf and channel. The Hub-leaf botnet is made by installing two bots on the victim's machine. The first bot is configured as a hub while the other bot is configured as a leaf. Additional bots can be configured as leaves, which connect to the hub bot. The resulting connection will form a star architecture. Hub-leaf botnets do not typically communicate through an IRC, but rather on configurable ports. Another type of botnet is called a channel botnet as shown in Figure 2.2, which allows bots to communicate through an IRC channel. Once a bot is configured on a victim's machine, it joins a predefined channel. The botmaster issues commands by posting messages to the IRC server. Bots read these messages, interpret and react to them. Another type of bot includes AOL bot which logs on to a set of AOL servers to receive commands. In addition, P2P bot uses peer-to-peer file sharing applications to spread, or to communicate with other bots.

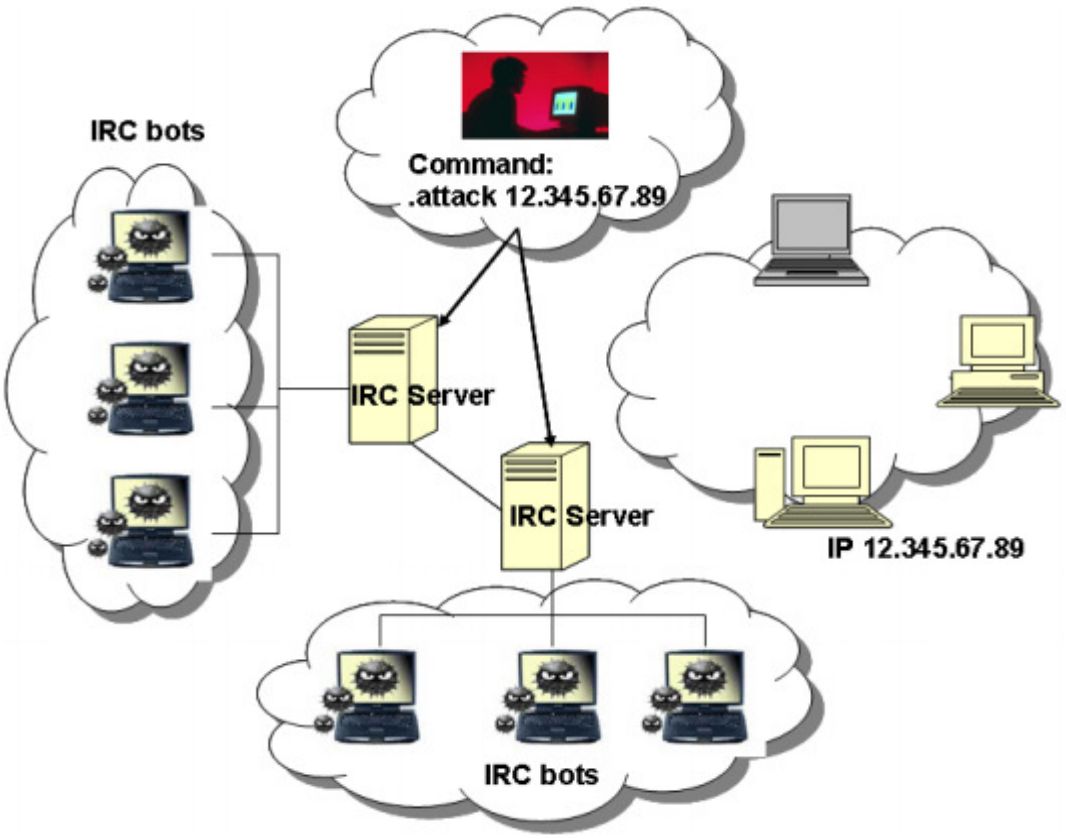

Figure 3.2: Structure of an IRC Channel Botnet.

3.3 DETECT THE BOTNETS

This can for example be obtained via an analysis of captured malware. Afterwards one can hook a client in the networks and gather further information. In the first part of this section we thus want to introduce our techniques to retrieve the necessary information with the help of honeypots. And thereafter we present our approach in observing botnets.

3.4 GETTING INFORMATION WITH THE HELP OF HONEYNETS

As stated before, we need some sensitive information from each botnet that enables us to place a fake bot into a botnet. The needed information include:

- DNS/IP-address of IRC server and port number
- (optional) password to connect to IRC-server
- Nickname of bot and ident structure

- Channel to join and (optional) channel-password.

Using a GenII Honeynet containing some Windows honeypots and snort_inline enables us to collect this information. We deployed a typical GenII Honeynet with some small modifications as depicted in the next figure:

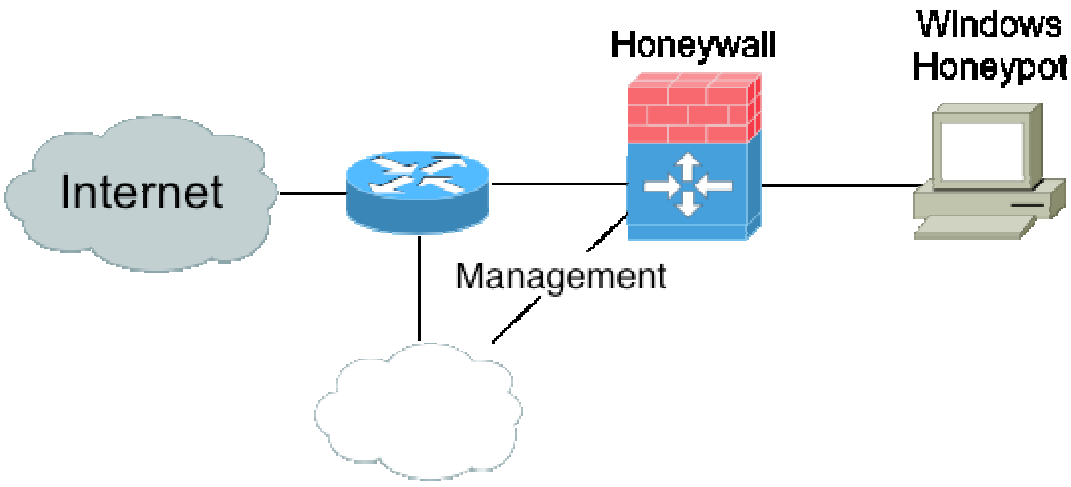

Figure 3.3: Deploy a GenII Honeynet

The Windows honeypot is an unpatched version of Windows 2000 or Windows XP. This system is thus very vulnerable to attacks and normally it takes only a couple of minutes before it is successfully compromised. It is located within a dial-in network of a German ISP. On average, the expected lifespan of the honeypot is less than ten minutes. After this small amount of time, the honeypot is often successfully exploited by automated malware. The shortest compromise time was only a few seconds: Once we plugged the network cable in, an SDBot compromised the machine via an exploit against TCP port 135 and installed itself on the machine.

3.5 OBSERVING BOTNETS

Now the second step in tracking botnets takes place, we want to re-connect into the botnet. Since we have all the necessary data, this is not very hard. In a first approach, you can just setup an irssi (console based IRC client) or some other IRC client and try to connect to the network. If the network is relatively small (less then 50 clients), there is a chance that your client will be identified since it does not answer to valid commands. In this case, the operators of the botnets tend to either ban and/or DDoS the suspicious client. To avoid detection, you can try to hide yourself. Disabling all auto response triggering commands in your client helps a bit: If your client replies to a "CTCP VERSION" message with "irssi 0.89 running on

openbsd i368" then the attacker who requested the Client-To-Client Protocol (CTCP) command will get suspicious. If you are not noticed by the operators of the botnets, you can enable logging of all commands and thus observe what is happening.

But there are many problems if you start with this approach: Some botnets use very hard stripped down IRCds which are not RFC compliant so that a normal IRC client cannot connect to this network. A possible way to circumvent this situation is to find out what the operator has stripped out, and modify the source code of your favorite client to override it. Almost all current IRC clients lack well written code or have some other disadvantages. So probably you end up writing your own IRC client to track botnets. Welcome to the club - ours is called ***drone***.

- No Threading: Threaded software defines hard to debugging Software.
- Non-blocking connecting and DNS resolve
- poll(): Wait for some event on a file descriptor using non blocking I/O we needed an multiplexer, select() could have done the job, too
- Written in C# since OOP offers many advantages writing a Multi-server client
- Modular interface so you can un/load (C#) modules at runtime
- libcurl: This is a command line tool for transferring files with URL syntax, supporting many different protocols. libcurl is a library offering the same features as the command line tool.
- Perl Compatible Regular Expressions (PCRE): The PCRE library is a set of functions that implement regular expression pattern matching using the same syntax and semantics as Perl 5. PCRE enable our client to guess the meaning of command and interact in some cases in a "native" way.

Drone is capable of using SOCKS v4 proxies so we do not run into problems if it's presence is noticed by an attacker in a botnet. The SOCKS v4 proxies are on dial-in accounts in different networks so that we can easily change the IP addresses. Drone itself runs on a independent machine we maintain ourselves. We want to thank all the people contributing to our project by donating shells and/or proxies.

3.6 BOTNET DETECTION AND PREVENTION OF PHISHING ATTACKS:

Phishing is a new type of network attack where the attacker creates a replica of an existing web page to fool users in to submitting personal, financial, or password data to what they think is their service provider’s website. The concept is a end-host based anti-phishing algorithm, called the link guard, by utilizing the generic characteristics of the hyperlinks in phishing attacks. The link guard algorithm is the

concept for finding the phishing emails sent by the phisher to grasp the information of the end user. Link guard is based on the careful analysis of the characteristics of phishing hyperlinks. Each end user is implemented with link guard algorithm. After doing so the end user recognizes the phishing emails and can avoid responding to such mails. Since link guard is characteristics based it can detect and prevent not only known phishing attacks but also unknown ones. The project uses the java technologies and oracle xe. Link guard is light-weighted.
In other words, You use the languages you know. Basically you need to be able to parse emails as they come in, perhaps by intercepting the traffic on your network. I would expect that it's a case of looking for things like links in mails that link to a different address to the one shown, and that are generally suspect in the way they are formed, which would be through rules that you would consider and write. Get a gmail account, wait a month and look through the spam folder.

3.7 BACKGROUND AND TERMINOLOGY

Malicious botnets are networks consisting of large numbers of bots. Bot is actually short for robot Symantec [39]. Bots are similar to worms and Trojans, but earn their unique name by performing a wide variety of automated tasks on behalf of their master.

Bots are also called \zombies" because a computer (infected with a bot) performs a task given by its master. A botnet herder (also called a botnet master) is a person or a group, who control the whole botnet: they can give instructions or upload data to the botnet. Botnets are used to perform a wide variety of tasks but some of the most popular ones are sending spam or coordinating a DDoS 1 attack.

There are different ways to infect a computer with a bot (see table 1 for an overview). Often it searches the Internet to look for \unprotected" computers to infect by exploiting known software vulnerabilities but it can also being sent through email, or hidden in another program or installed by a malicious website, this software is also known as malware.

A peer-to-peer botnet incorporates a P2P protocol in his code. Although such a protocol already existed for a couple of years it is but recently that we have encountered this in botnets. Some peer-to-peer bots have used existing peer-to-peer protocols while others have developed custom protocols [14].

Threat	Description
Email	Malware in email messages
Exploit	To exploit a particular vulnerability in a system
File shares	Network shares provide a transport mechanism for malware
Instant messaging	Through sharing files with other users malware can spread
Dictionary attack	Guessing a user's password to get access
Internet downloads	Downloaded directly from Internet Web sites
Removable media	Malware can spread when using this media on different computers
Network scanning	Scan for computers that have open ports or attack randomly IP addresses
P2P networks	Install malware after downloading innocent looking files

Table 3.1: Malware threats (Source: Microsoft.com [1])

3.8 BOTNET ARCHITECTURES

There are a few topologies described in this chapter. They differ from each other in how big they can get how easy they can be detected and disrupted and how they exchange messages and commands (command and control).

3.8.1 Centralized

The oldest type of topology is the centralized form, see figure 1 on the following page. There is one central point that forwards commands and data between the botnet herder and his botnet clients. The big advantage is that there is little latency. The biggest drawback is that they can be more easily detected then decentralized, since all the connections lead to a few nodes. Also when a IRC server gets disrupted or disconnected, an entire branch or perhaps the whole botnet could be taken offline because the botnet herder cannot pass any messages to the bots anymore. Passing commands and data in centralized systems go through a central point: in most cases they all connect to a central C&C node or a central IRC server where they will receive their commands.[27] These IRC servers can be setup on hacked computers, or the botnets could use public IRC servers. When the IRC server has been taken offline, the botnet will become \lost", meaning that the botnet herder will not be able to send commands and data to the bots, rendering his bots useless. The more modern IRC botnets are more resilient because they use a list of IP addresses of alternate IRC servers, which they will use in case an IRC server has been taken offline.

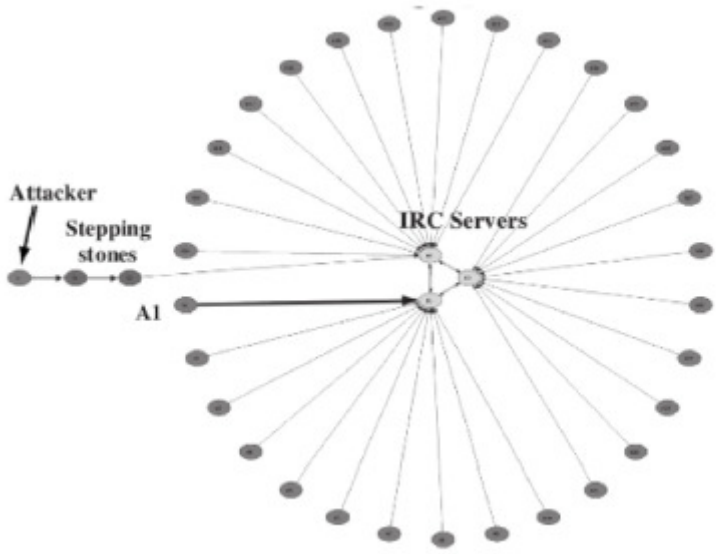

Figure 3.4: Centralized topology

3.8.2 Design

The aim of the thesis is to create a detection application that will monitor network and host activity and also to create an application that will mimic a malicious Botnet. In the Literature review the basic models of Botnets have been researched, enabling the design component of this thesis to begin.

The programming language used is important for the creation of both the detection software and the Botnet. Following research on the subject it was found that the majority of botnets are Windows-based and most of these are Window XP [78]. The language that will be used is C# and .NET which allows good integration with the operating system and will work with Windows XP, Vista and Windows 7.

Section 3.2 details the scenario for a working model of the synthetic botnet and describes the different types of attacks that will be emulated. The design of the bot will be described in Section 3.3. In Section 3.4 the design of the detection software will be described and will be split into two parts: network-based and host-based detection.

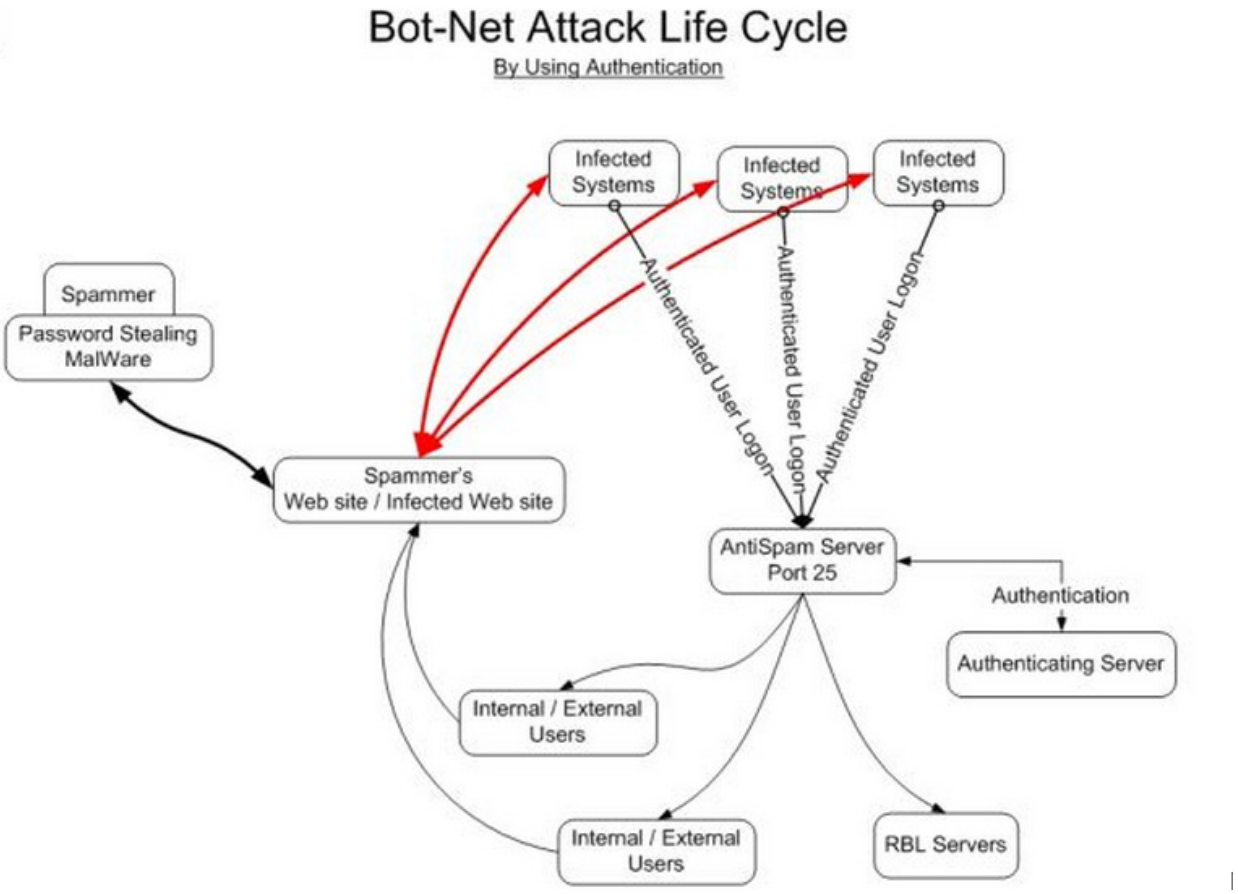

Figure 3.5 Bot-Net Attack Life Cycles

- Initial setup of configuration settings of the Bot parameters such as infection vectors, payload, stealth, C&C details
- Register a dynamic DNS (DDNS)
- Register a static IP
- Bot Herder infect PC's with the Bot(s)
  - Bot propagates the infection according to the configuration settings
  - Bot scans for vulnerabilities that it may encounter
  - Idle
  - Performs actions received by other Bots above it in the chain of command
  - Bot dies:
    - Bot may be taken over by another Botnet
    - The owner of an infected PC with a Bot realizes the PC is a zombie so it kills the Bot.
    - The chain of command may be compromised above the level.

This invasion technique has evolved not only to an illegal way of profit, but also to a way of violating people's information privacy privilege. What have been Botnets being used for? Is the whole concept of Botnet Evil? What actions are being taken to control this terrible plague? On the following sections, this article will discuss how has this threat being used (mainly as one effective malicious way of breaking into data integrity, availability and confidentiality of a computer network) and ways to detain it.

3.9 FRAMEWORK FOR BOTNET/BOTS DETECTION

To detect a botnet or an individual bot, we designed a framework which is mainly based on monitoring API function calls executed by a process to determine if that process is anomalous or not, as shown in Figure 3.4. The framework consists of two modules. The first module is responsible for detecting the botnet while the second module is responsible for detecting an individual bot on the system. From Figure 3.4, three main API function calls are monitored, K for key logging function calls, C for communication function calls and F for file access and registry function calls. The botnet module uses the APITrace tool to generate log files from different systems. Each file is monitored by another program which produces the change of log file data. The data is processed and analyzed to detect similar changes from different log files which may indicate botnet activity. The bot detecting module also uses the APITrace tool to monitor the same API function calls. To begin with, we use a simple correlation algorithm (i.e the Spearman's Rank Correlation - SRC) to detect an individual bot on a host by correlating different activities generated by a process. A more intelligent way (i.e. the Dendritic Cell Algorithm - DCA) of correlating different activities is used later to detect the existence of an individual bot on a system. Next, we will describe the framework in more details hosts. The aim of using log correlation is to develop a host-based algorithm that is capable of detecting similar activities of similar type of IRC bots by monitoring the change of behaviours in log file sizes across several hosts and find the correlation between these changes. The detection technique is based on monitoring the change of behaviours from one state to another state in each host and observes the common actions or responses generated by bots in all hosts. This is due to the fact that bots are responding to the commands simultaneously which produce the same rate of change in each log file. The advantage of applying this approach is that this detection technique does not require searching for specific patterns when analysing network traffic. Therefore, the amount of processing time required to detect botnets is reduced. In addition, this technique does not monitor standard ports and can deal cryption techniques or hidden from the attacker. We also assume that the attacker cannot modify or delete these files remotely as they require administrator privileges.

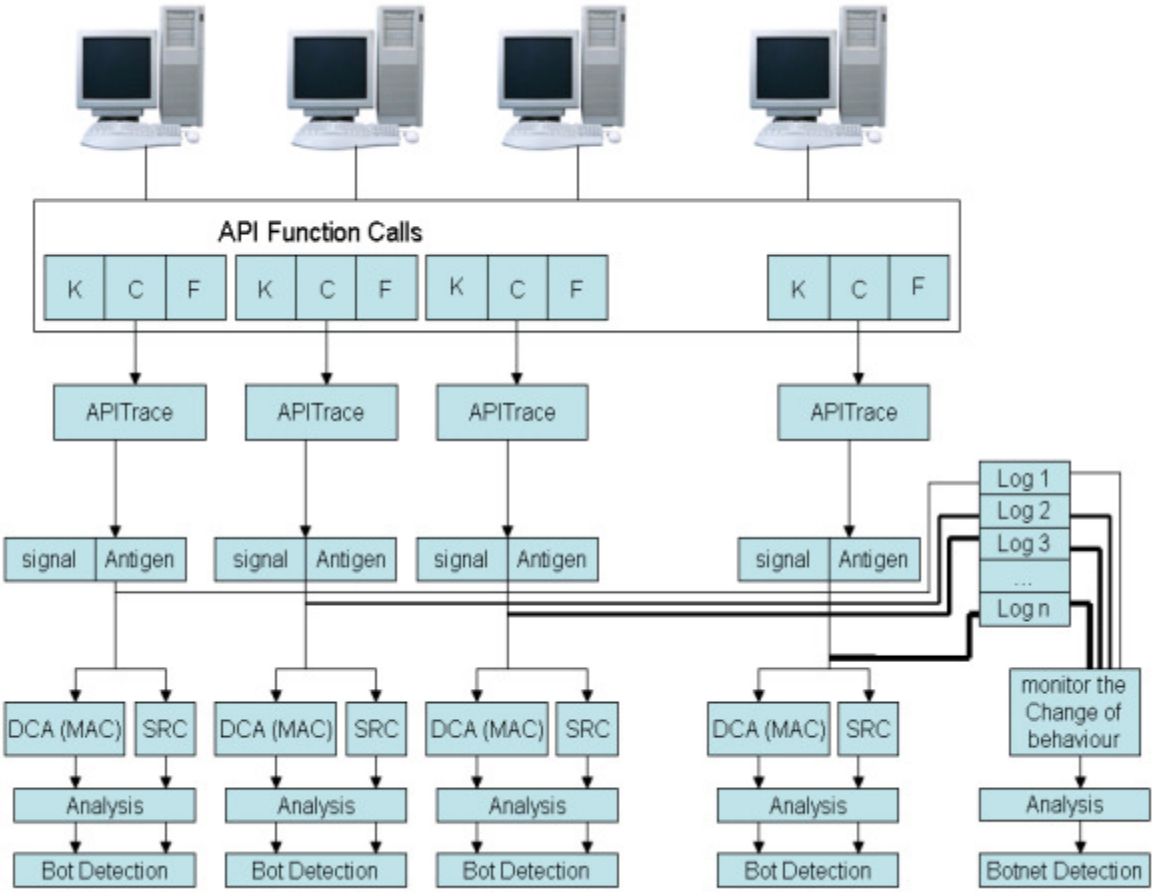

Figure 3.6: Framework for detecting botnet/bot

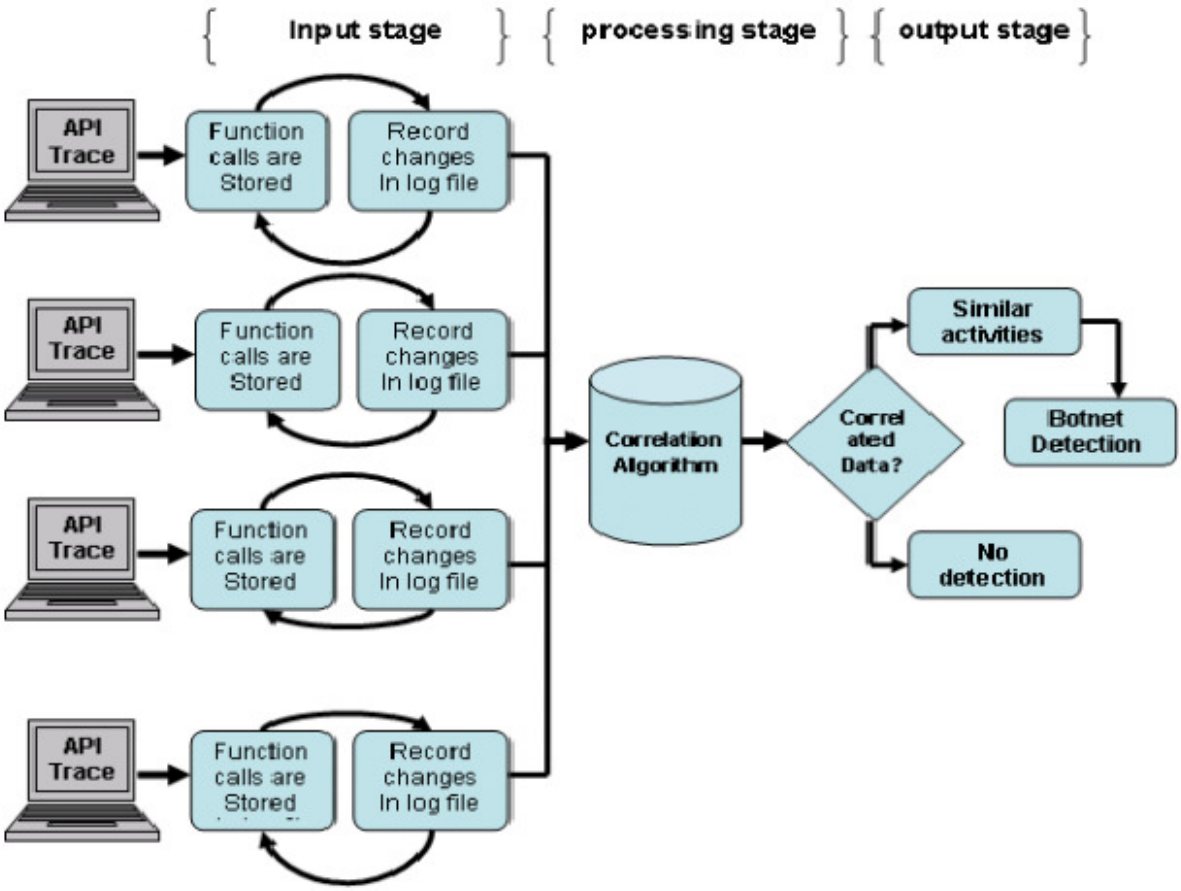

Figure 3.7: Botnet Model for detecting bots

3.10 CATEGORIES OF MALWARE

Malware code can be classified in to various classes:

3.10.1 Viruses

A small program having malicious intent and the ability to create copies of itself is termed as virus. Virus code attaches itself with other programs and then replicates [3]. It works by inserting virus code in an executable file. Virus code gets executed when the file is run. Metamorphic viruses evolve into new variants by changing itself. A program in which virus has inserted itself is denoted as the host for the virus. A virus requires an existing host program so as to cause damage.

3.10.2 Worms

The software which makes its copies by executing its own code regardless of any other program is known as worm. It sends copies of itself to other systems in the network unnoticeably without

the authorization of the user. Worms consume the bandwidth of the network to halt it. Worms spread through network linkages and have the aim to infect most of the computer systems associated with the network. Worms do not require the existence of any file unlike virus. It could encrypt files, delete files or send junk email.

3.10.3 Spyware

The software which monitors and collects private information of the user is termed as spyware. It usually gathers data like the key pressed by user, email address, pages regularly visited, credit card number etc [4]. This can happen when users download free or trial software. The users are monitored by scouts in this type of malware therefore their account numbers, passwords and every second personal element become exposed.

3.10.4 Adware

Adware also known as advertising-supported software presents, plays, or downloads advertisements in a computer system automatically once malicious software is installed or application is used. The code of this malware is normally embedded into free software. Its main objective is to monitor the activities of the user of the system. Free games and peer-to-peer clients are some examples of the common adware programs.

3.11.5 Trojans

A Trojan horse is a malware injected by its designer in an application or system. The application or system is accomplishing some illegal action but seems to carry out certain valuable function. Remote hijackers use this malware to launch their attack and they use our system for their own purpose. They could acquire our passwords, monitors what is happening on our system or corrupt the system files.

3.12.6 Botnet

Botnet is a type of malware that takes the control of our system distantly and sends spam or spyware. Mostly, botnets are like zombie and work under the command of the party who runs it. Bot doesn't wait for a command from the third party by sitting around the infected machine. On the other hand, it looks for the communication having like occurrences of bots awaiting instructions. Simple and hierarchical are the two types of botnets.

3.11 MALWARE DETECTION TECHNIQUES

Malware detection is referred as classifying the code in to two classes: genuinely benign and malicious. Malware detection technique should have the capability of handling obfuscated malware efficiently for robust malware detection. The main idea behind code obfuscation is that it modifies only malware syntax but preserves its proposed behaviour [1]. A system that tries to recognize malware using signatures and other heuristics factors is known as malware detector. Four methods employed for malware detection are:

3.11.1 Signature-Based Detection

Signatures are typically a sequence of bytes which the various antivirus scanners seek inside the malware code to state that the program scanned is malicious in nature. By observing the disassembled code of malware binary, these signatures are designed. Several debuggers and disassemblers are present for disassembling the portable executable. After that features are extracted by analyzing the disassembled code. These features are then used in the creation of the signature of specific category of malware.

3.11.2 Behavior-Based Detection

Behavior based detection focuses on the actions performed by the malware instead of the binary sequence. The programs having identical behavior but different syntax are recognized. Consequently several illustrations of malware are recognized by a single behavior. The main aim is to analyse the behaviour of well-known or unfamiliar malwares. Several elements such as source and destination address of malwares, attachments types and other statistical features constitute the behavioral parameters. These forms of detection mechanisms are helpful in discovering those malwares that continuously generates novel mutants as system resources and services are used by them in a similar way.

3.11.3 Specification-based Detection

Specification-based detection is derived from anomaly based detection. Specification-based detection estimates the necessities of an application or a system rather than estimating their execution. There is a training phase in specification-based method in which we try to acquire information about the legal behavior of a program or system which is being examined[4].The chief drawback of specification based technique is that it is very challenging to correctly specify the behavior of the system or program.

3.11.4 Anomaly-based detection

The anomaly detection technique is based on the idea of a baseline of the network behavior. The baseline specifies the accepted network behavior, which is stated by the network administrators. Any behavior that does not correspond with the predefined or accepted model of behavior causes events in an anomaly detection engine. It is a two-step approach that involves first training a system with data to create certain conception of normality and then use the established profile on actual data to report deviations. This approach gives anomaly-based IDSs the power to detect new attacks which are new and for which signatures are not present.

Why are Botnets dangerous today

Botnets today are one of the most dangerous species of network-based attack because they use large, coordinated groups of hosts to execute both brute-force and subtle attacks. A collection of bots, when controlled by a single command and control (C&C) infrastructure, forms a botnet [11], [18], [19]. Since the bots work together in large groups taking orders from a centralized botmaster, they can cripple a large-scale network in a short time.

A lot of work has been done trying to mitigate the efforts of botnets to avoid data and financial loss. However hard the industry works towards patching the known vulnerabilities in hosts and networks, there are always more unpatched or unknown vulnerabilities that malicious developers and cyber criminals may exploit.

CHAPTER 4

METHODOLOGY

METHODOLOGY

4.1 BOTNET DETECTION

Generally, a botnet applies for a area name for each of bots and allocates the domain names (usually in the form of URLs) via various channels, such as spam mails or web blogs. Though, if a machine is in down time, the bot cannot be measured and the URL will be provisionally inaccessible.
Furthermore, control of the bot could be misplaced due to removal of the malicious software. In this case, the bot herder will not gain any more assistances from the domain name except it is re-mapped to another IP address (of another bot).

A TCP based botnet (called a *botnet* for short), solves the above-mentioned problems because of two architectural innovations:
1) The mapping between domain names and IP addresses, and
2) The method sincere users' requirements are processed.

First, in a botnet, *a domain name is mapped to a number of IP addresses* (possibly hundreds, or even thousands) rather than a single IP address. As a result, if the mapping is fingered properly, i.e., a domain name is always determined to a manageable and live bot, the productivity (in terms of the access rate of malicious services) will be higher than that of a traditional botnet. In adding, if it is known that a bot has been detected, the area name's link to the bot can be finished directly so that their association cannot be discovered.
Second, *legitimate users' requests are indirectly handled by other machines called motherships, rather than the bots the users contact*. In other words, when a genuine user accesses a facility providing by a botnet via a URL, the bot that the URL connects to and receives needs from does not handle the requirements itself. Instead, it helps as a proxy by giving the requests to a mother-ship, and then advancing the mother-ship's responses to the user.
By so doing, bot herders can update a malicious service (and the content it offers) anytime because they have more control over the mother-ship and the number of mother-ship nodes is relatively small likened to that of bots. In addition, since malicious services do not reside on bots, it is easier for bot herders to reduce the footprint of the malicious software so that it is less likely to be noticed by anti-malware solutions.

- **Returned DNS records at time t -**

;; ANSWER SECTION:

f07b42b93.com. 300 IN A 68.45.212.84

f07b42b93.com. 300 IN A 68.174.233.245

f07b42b93.com. 300 IN A 87.89.53.176

f07b42b93.com. 300 IN A 99.35.9.172

f07b42b93.com. 300 IN A 116.206.183.29

f07b42b93.com. 300 IN A 174.57.27.8

f07b42b93.com. 300 IN A 200.49.146.20

f07b42b93.com. 300 IN A 204.198.77.248

f07b42b93.com. 300 IN A 207.112.105.241

f07b42b93.com. 300 IN A 209.42.186.67

Returned DNS records at time t+300 second -

;; ANSWER SECTION:

f07b42b93.com. 300 IN A 64.188.129.99

f07b42b93.com. 300 IN A 69.76.238.227

f07b42b93.com. 300 IN A 69.225.51.55

f07b42b93.com. 300 IN A 76.10.12.224

f07b42b93.com. 300 IN A 76.106.49.207

f07b42b93.com. 300 IN A 76.127.120.38

f07b42b93.com. 300 IN A 76.193.216.140

f07b42b93.com. 300 IN A 99.35.9.172

f07b42b93.com. 300 IN A 200.49.146.20

f07b42b93.com. 300 IN A 204.198.77.248

Above is an example of how a botnet rapidly changes the mapping of IP addresses to its domain names. These two consecutive DNS lookups are 300 seconds apart.

To unclear the link between a domain name and the IP addresses of available bots, botnets often employment an approach that resolutions a domain name to different sets of IP addresses over time. For

example, we experiential that the malicious service f07b42b93.com, which hosts a phishing webpage that two-times users by getting them to disclose their iPhone serial numbers, adopts this strategy.

As shown in figure during a DNS query at time t, the domain's DNS server replies with 10 A records, any of which will lead users to the phishing webpage. The short time-to-live (TTL) value, i.e., 300 seconds, designates that the records will expire after 300 seconds, so a new DNS query will then be compulsory. At t+300 seconds, we re-issued the same query and obtained another set of IP addresses.

In total, there are 19 IP addresses with one duplication in the two sets, which indicates that the bot herder presently owns a minimum of 19 bots. The duplication could occur because the DNS server returns IP addresses randomly, or the bot herder does not have enough bots and cannot deliver any more unseen IP addresses.

A single botnet domain name may be determined to a huge number of IP addresses. For example, we observed a total of 5,532 IP addresses by resolving the domain name nlp-kniga.ru between October 2009 and March 2010. The larger the IP address pool, the higher will be the "productivity" of the botnet. As a result, the link between any two bots that serve the same bot herders will be less clear, which is exactly what the bot herder's longing.

4.2 A BRIEF DESCRIPTION OF SVM ALGORITHMS

For small problems, customary algorithms from optimization theory exist. Examples include conjugate gradient decent and interior point's approaches. For larger problems, these algorithms do not work well because of the big space requirements to store the kernel matrix. Often they are slow since they do not make use of the features of real-world SVM problems, for example that the number of support vectors are usually sparse.

Classification Data

Pairs of observations (xi , yi) generated from some distribution P(x, y), e.g., (blood status, cancer), (credit transaction, fraud), (profile of jet engine, defect)

Task

Approximation y given x at a new location.

Modification: find a function f(x) that does the task

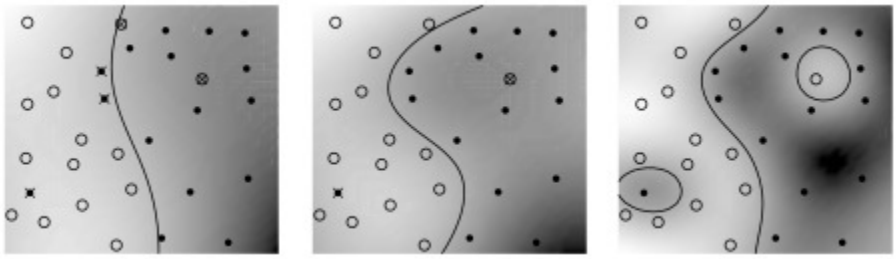

Figure 4.1 SVM classification

Though the SVM can be functional to various optimization problems such as regression, the classic problem is that of data classification. The basic idea is shown in figure 4.1. The data points are recognized as being positive or negative, and the problem is to find a hyper-plane that separates the data points by a best margin.

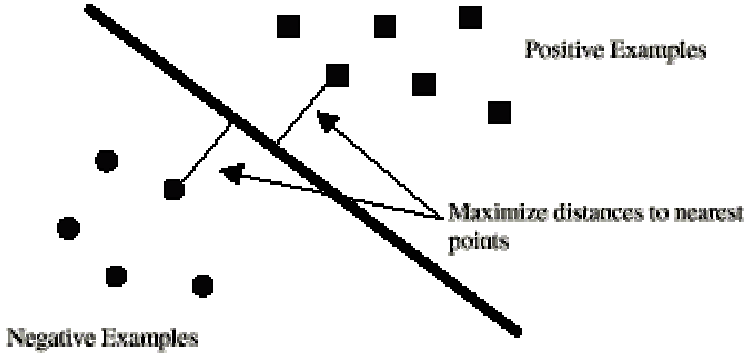

Figure 4.2: Data Classification

The overhead figure only shows the 2-dimensional case where the data points are linearly separable. The mathematics of the problem to be solved is the following:

$$\min_{\vec{w},b} \frac{1}{2}\|w\|,$$

$$s.t \quad y_i = +1 \Rightarrow \vec{w}\cdot\vec{x}_i + b \geq +1$$

$$y_i = -1 \Rightarrow \vec{w}\cdot\vec{x}_i - b \leq -1$$

$$s.t \; y_i(\vec{w}\cdot\vec{x}_i + b) \geq 1, \quad \forall i \tag{1}$$

The identification of the each data point x_i is y_i, which can take a value of +1 or -1 (representing positive or negative respectively). The solution hyper-plane is the following:

$$u = \vec{w} \cdot \vec{x} + b$$

(2)

The scalar b is also termed the bias.

A standard method to solve this problem is to apply the theory of Lagrange to convert it to a dual Lagrangian problem. The dual problem is the following:

$$\min_{\alpha} \Psi(\vec{\alpha}) = \min_{\alpha} \tfrac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} y_i y_j (\vec{x}_i \cdot \vec{x}_j) \alpha_i \alpha_j - \sum_{i=1}^{N} \alpha_i$$

$$\sum_{i=1}^{N} \alpha_i y_i = 0$$

$$\alpha_i \geq 0, \quad \forall i$$

(3)

The variables α_i are the Lagrangian multipliers for corresponding data point x_i.

4.3 SNIFFERS (PASSIVE ATTACKS)

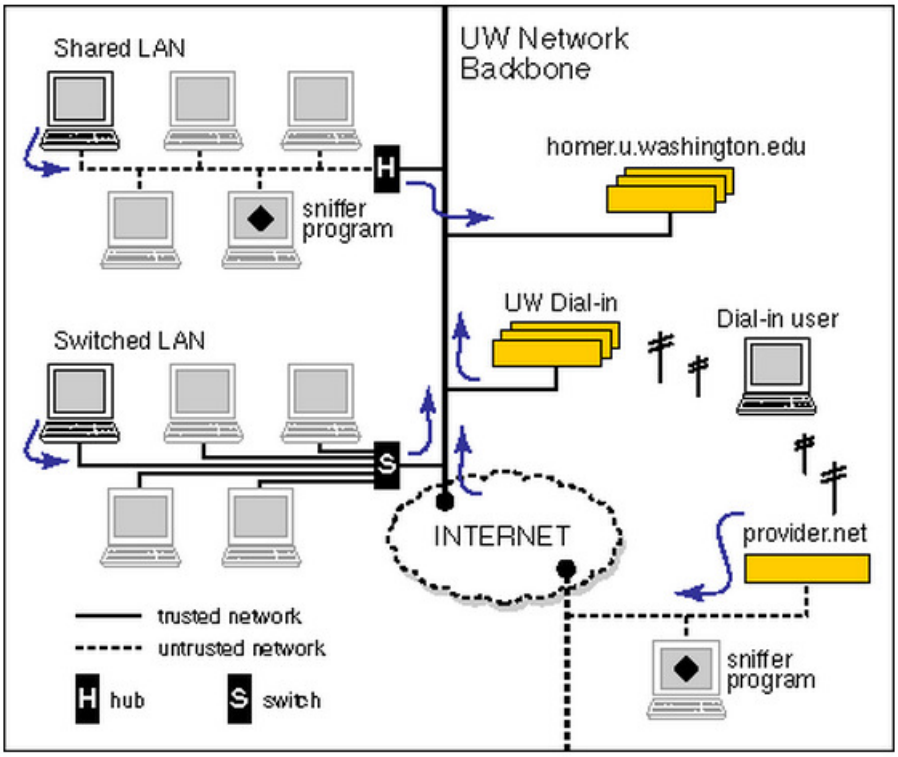

Figure 4.3 Botnet sniffer

- Passively watches for 3-way handshake
- Vulnerable services include telnet, ftp ,rlogin, IMAP, POP ...
- Logs *N* packets, or until FIN, RST, or timeout
- Stuffs everything into a log file
- Newer sniffers unlink themselves, unlink their log files, send logged data to collectors in ICMP packets

4.4 BOTNET PACKET ANALYZER

This protocol allows the compilation of information concerning the information of IP traffic of a network for its following analysis. The routers and switches that support this protocol not only allocate traffic through the network but also send metadata from the above-mentioned traffic (IPs, ports…) to a specialized compilation server.

This server transmits out the consistent analysis aimed at classifying anomalous traffic patterns, communications with other ports that are in theory filtered, connections to malicious IPs, excesses of a

certain type of traffic, sessions that go on for too long…which can specify whether a network or some of its members have been cooperated.

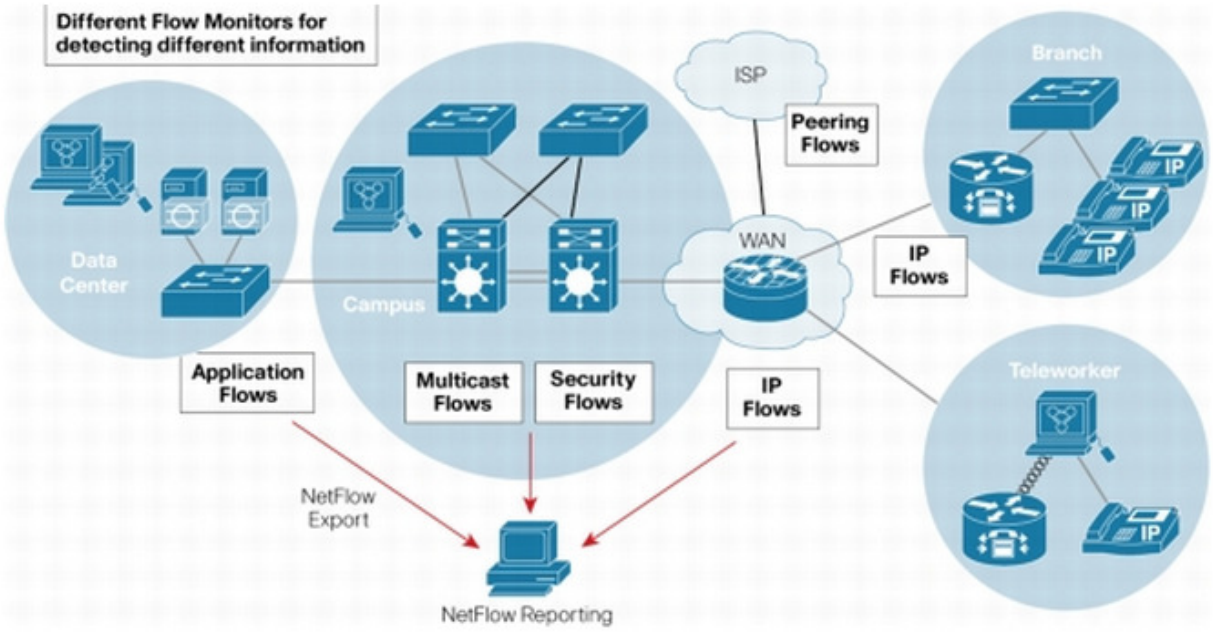

Figure 4.4 Example of Net-Flow architecture

The analysis of network traffic flow can detect both unidentified threats, by detecting anomalous conduct such an unidentified botnet, and before recognized threats, such as known IPs from the central C&C of a botnet.

Another advantage of the analysis technique is the option of storing the reduced metadata from communications, thanks to which retrospective analyses of threats or attacks can be carried out, with the aim of reviewing the system's response and improving the current counter-measures. On the other hand, this technique makes a considerable overflow of traffic, since the routing devices send all the metadata from communications to the server which is accountable for storing and processing this information.

4.5 PORT SCANNER

Port scanner implement can be used to classify obtainable services running on a server, it uses raw IP packets to find out what ports are open on a server or what Operating System is running or to checkered if a server has firewall allowed etc.

The service can also detect uptime of a host if the host is running one of the known Operating Systems which the scanner can analyze to conjecture uptime To scan a host just enter the host name or the IP

address in the box above and give a range of ports to scan, if the host has firewall permitted then you can try a dissimilar type of scan in the advance mode.

The optional scan type is "connect()", however other scan types can also be useful contingent on the network of the target host, a "SYN Stealth" scan type can also show to be valuable when there's a firewall blocking the ports.

CHAPTER 5

RESULTS AND IMPLEMENTATION

5. RESULTS AND IMPLEMENTATION

In this method we have to classify the botnet detector /intruder that will detect host-name, IP address. Finally we mature an application in C# using some networking outline and some related networking programming which will identify the botnet detector, using botnet attack will evaluate the operative, and that will monitor the Intruder activity and this tool also measure of capture the concept of intruder conduct, and it denial at the 2 time of intruder time until it will monitor by administrator panel

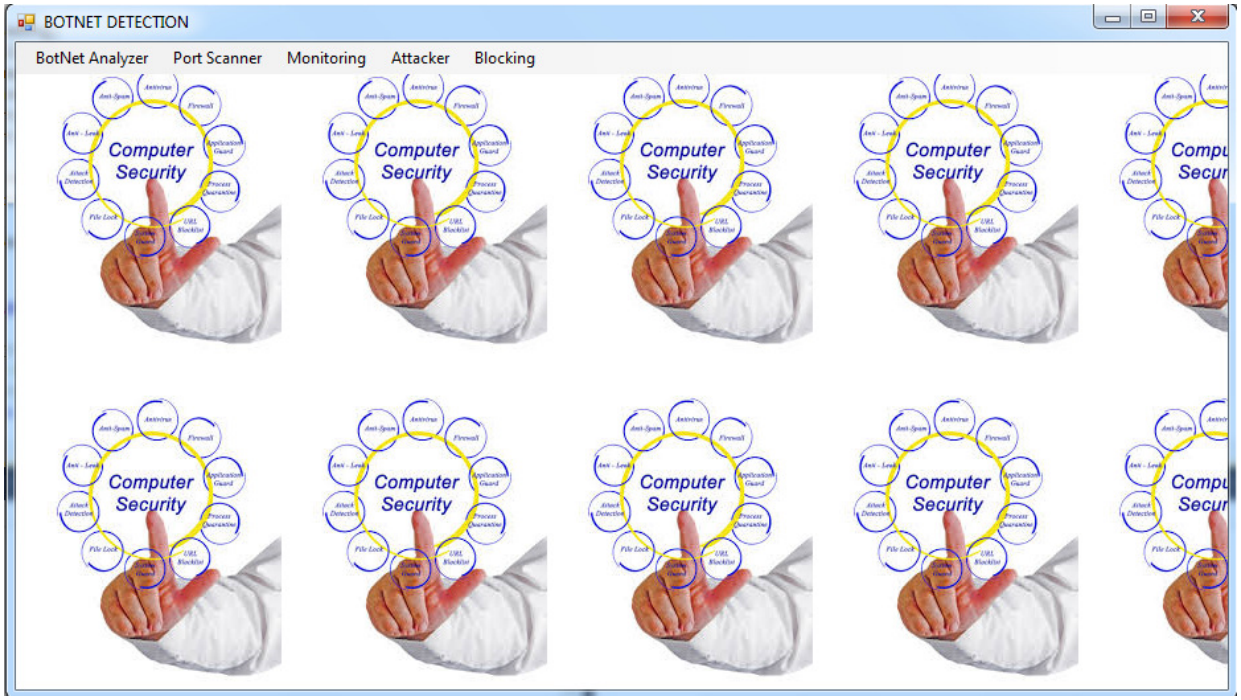

Figure 5.1: Module for BOTNET detection and Prevention

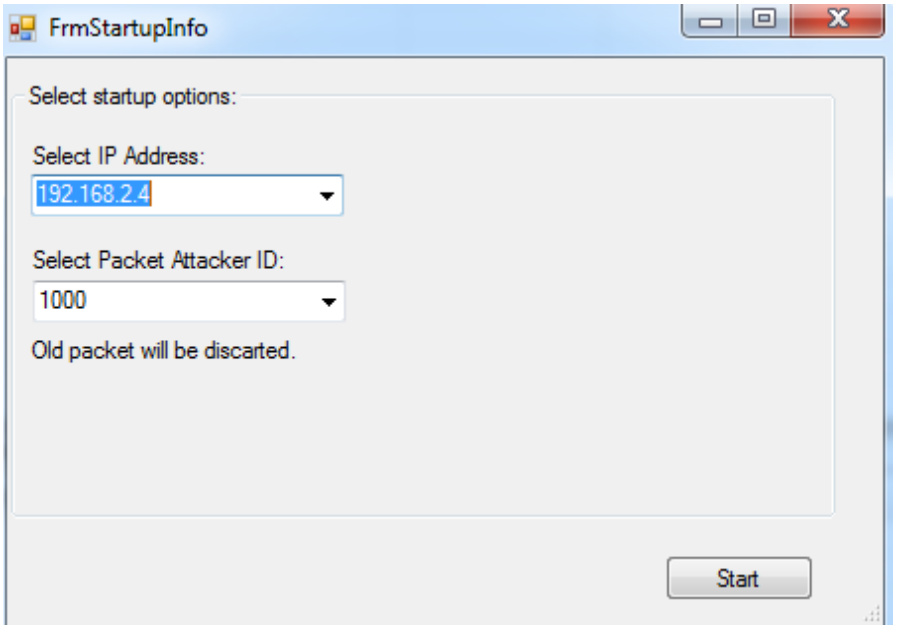

Figure 5.2: Enter Host IP Address with no of packet for BOT network

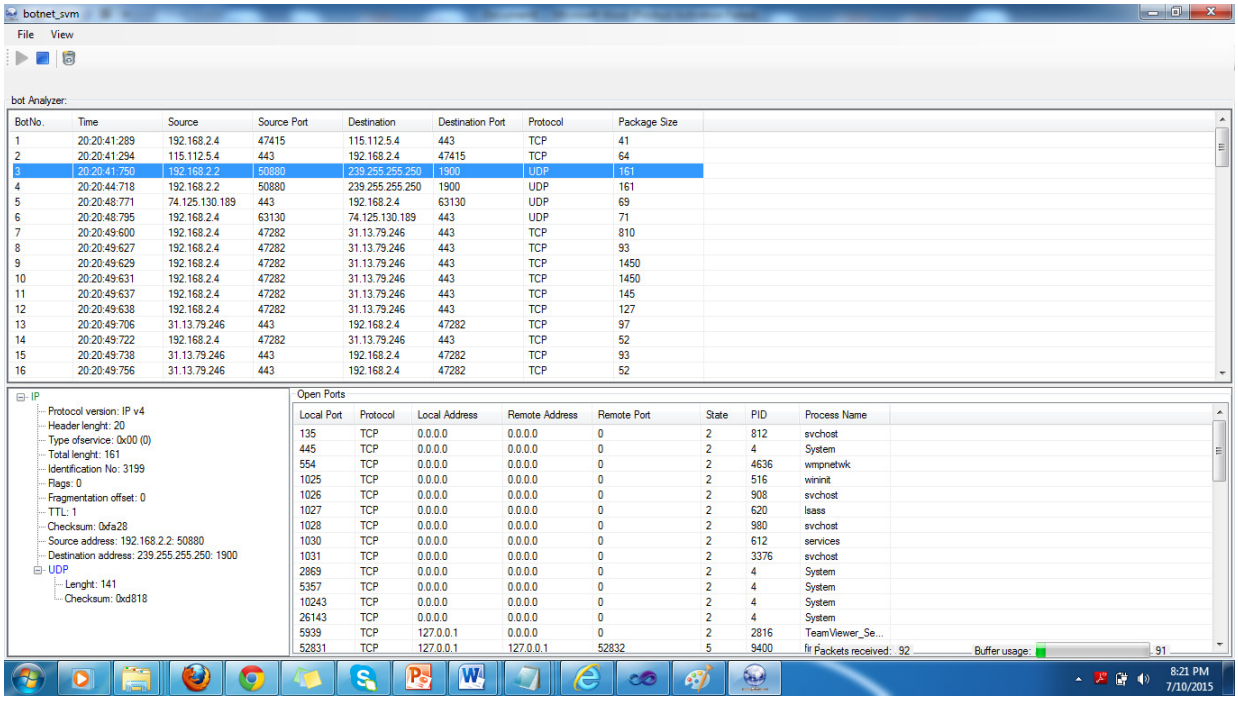

Figure 5.3: Bot analyzer with its packet length and botnet identification number from source to destination

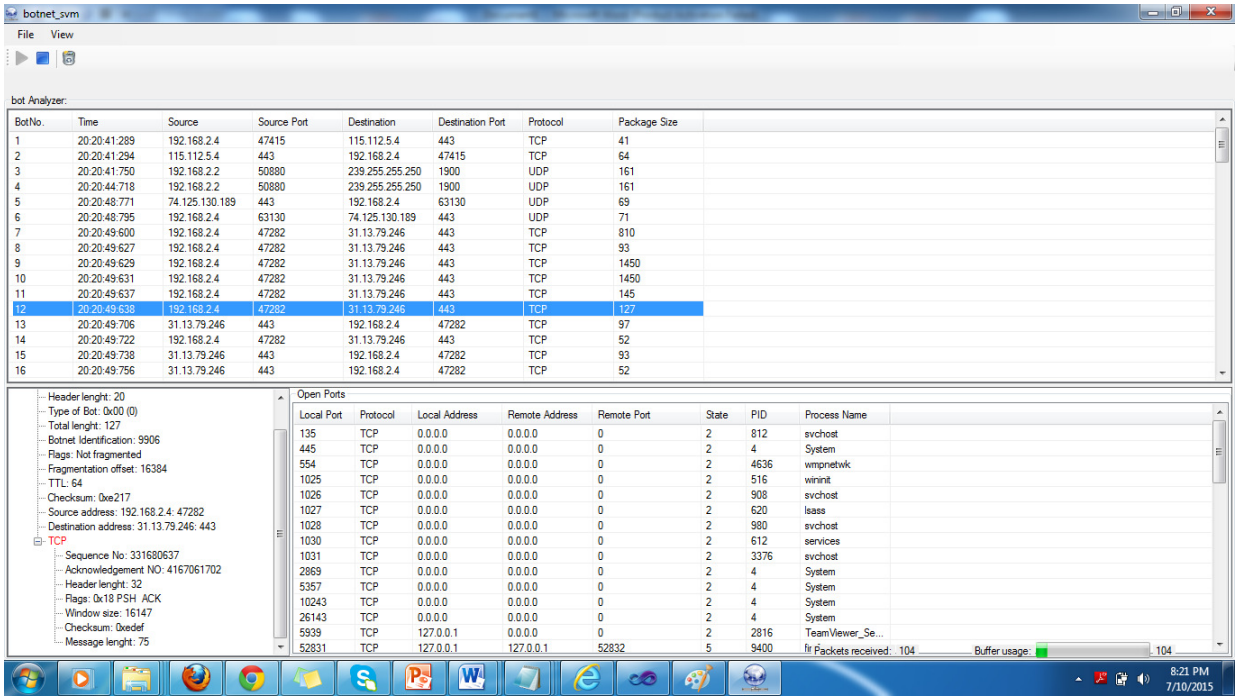

Figure 5.4: Bot analyzer with its packet length and botnet identification number in TCP tracing

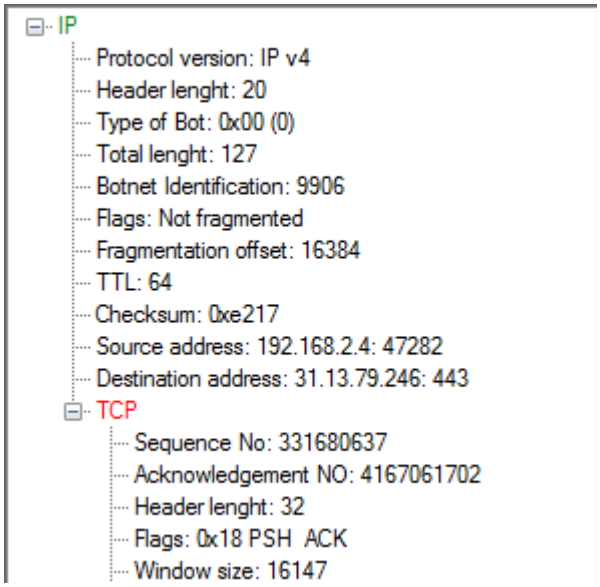

Figure 5.5: Different sequence number for Botnet identification for 32 header length

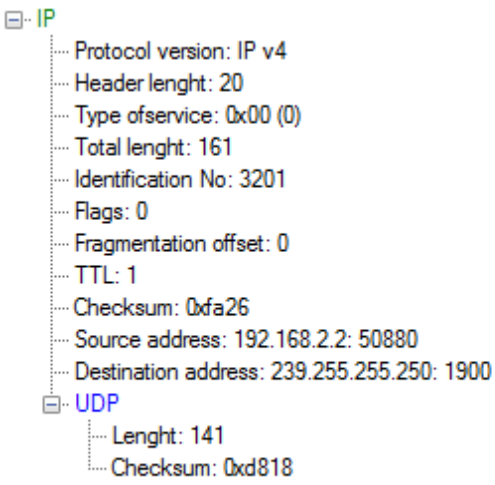

Figure 5.6: Different sequence number No- Botnet identification for 141 header length and ofservice is 0x00

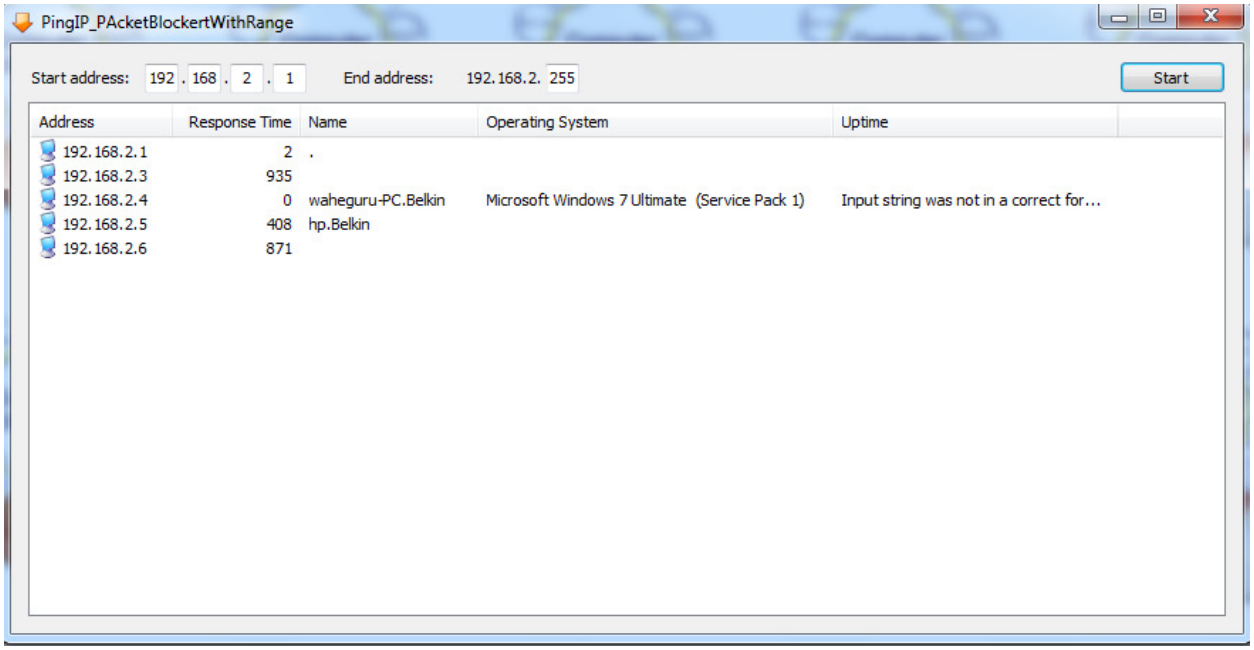

Figure 5.7: BOTNET prevention module from theft IP Pinging Blocker

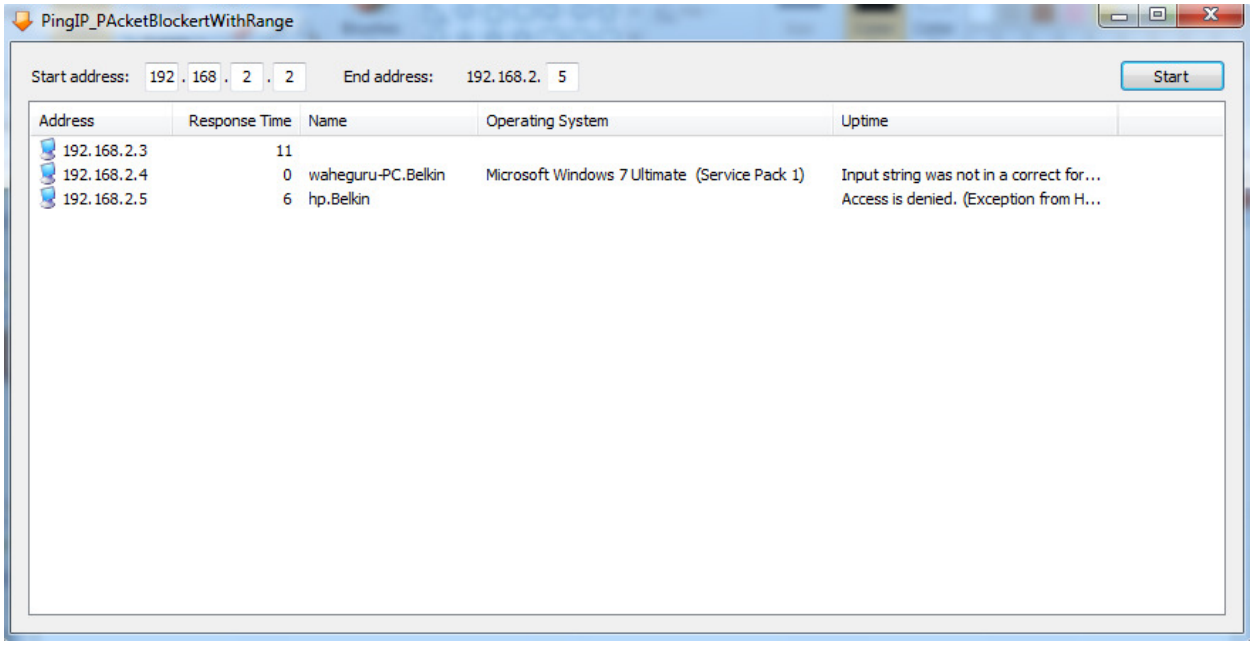

Figure 5.8: BOTNET prevention module from theft IP Pinging Blocker towards selected range

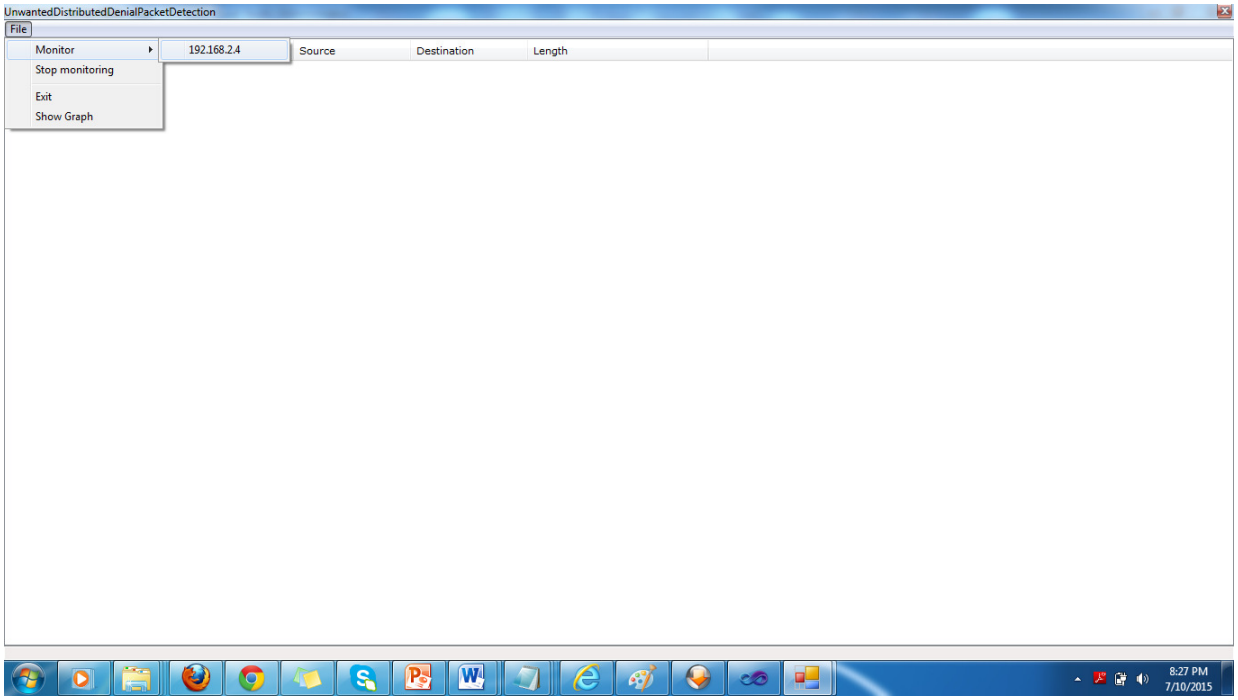

Figure 5.9: Start monitoring from host IP

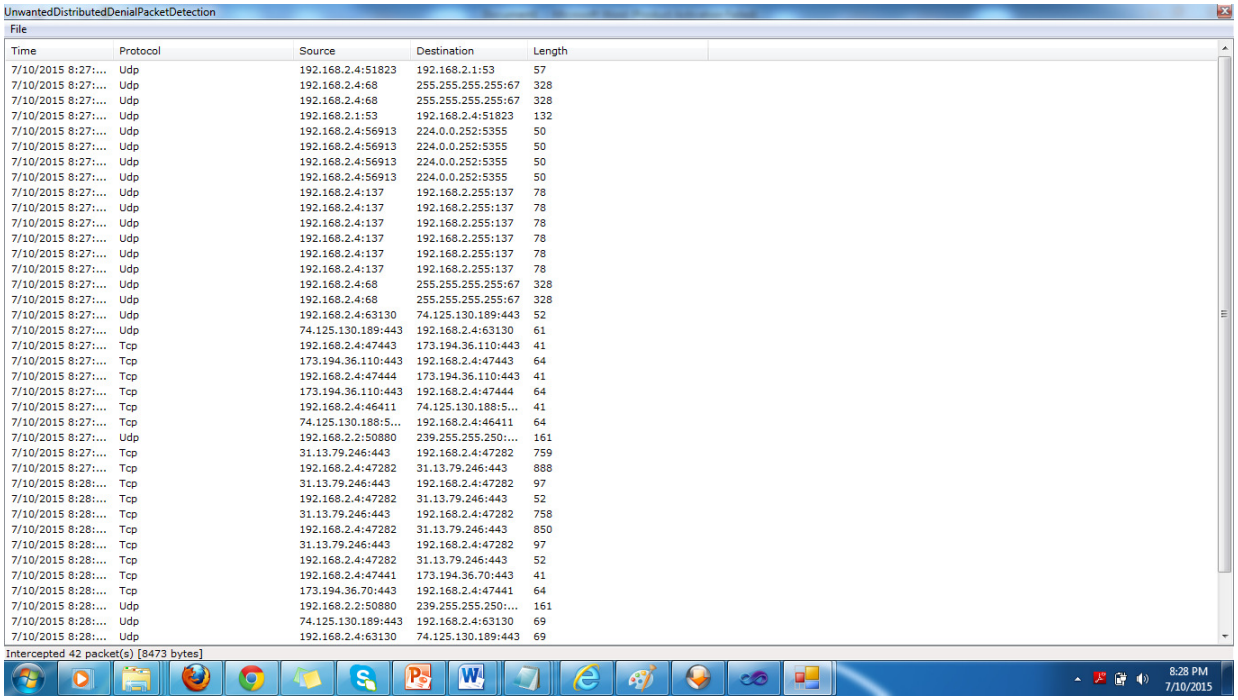

Figure 5.10: Total number of packet monitoring under TCP/UDP

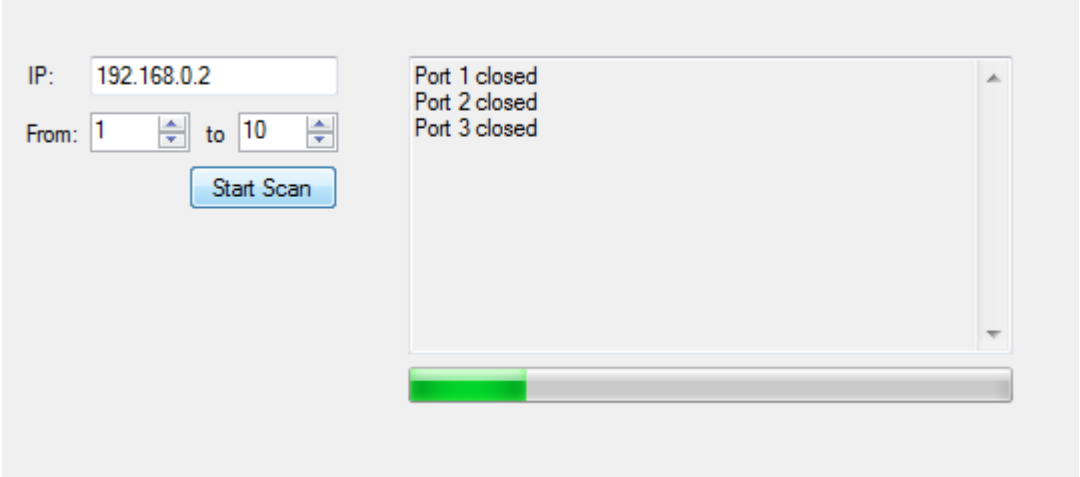

Figure 5.11: Scan port then close to prevent directory of support h1, h2 and h3

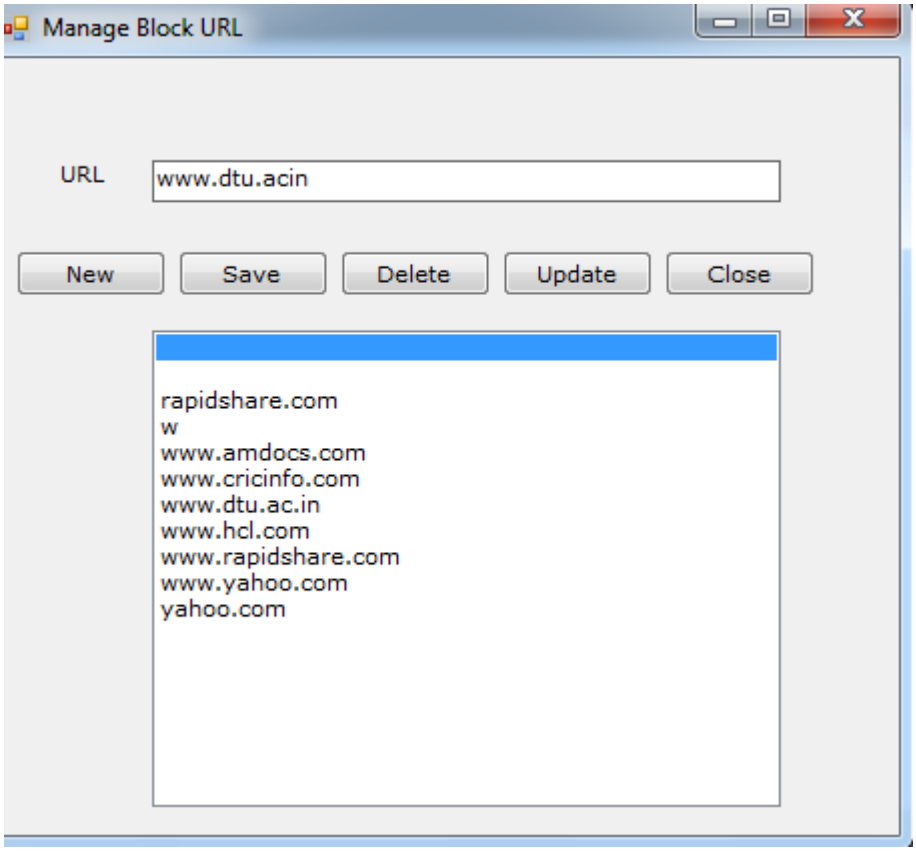

Figure 5.12: Block unauthenticated access from web browser

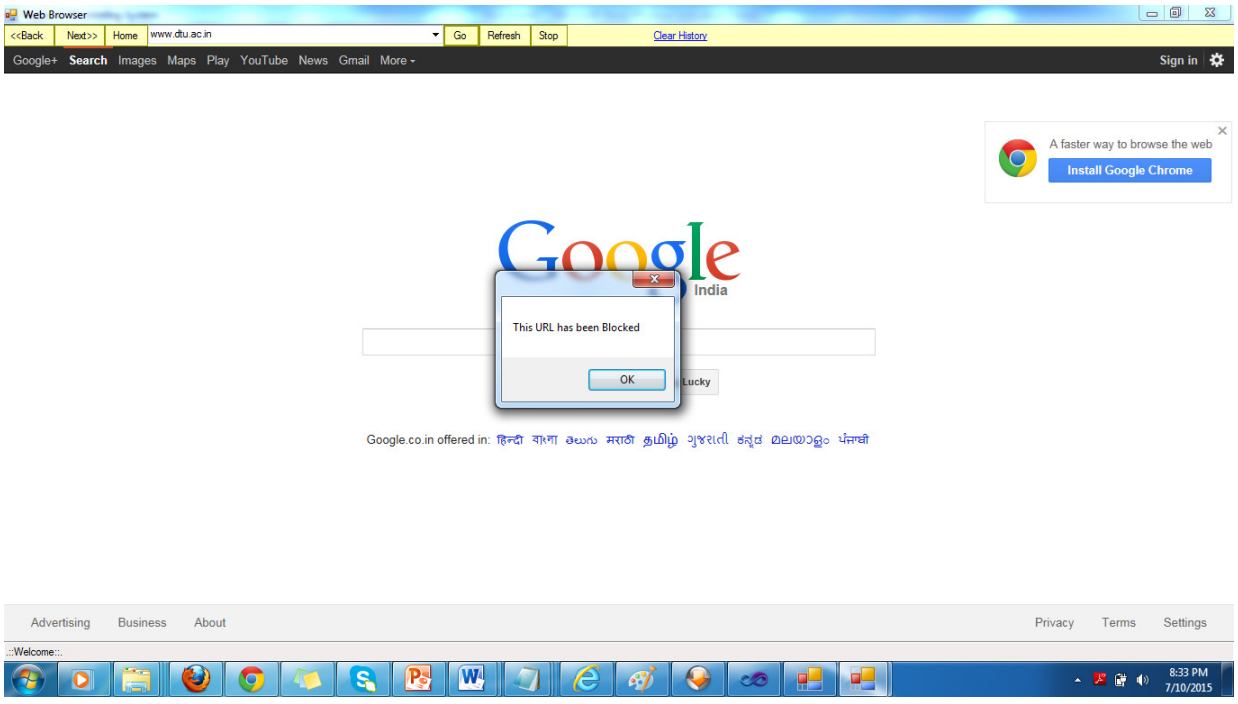

Figure 5.13: block URL from web browser for preventing

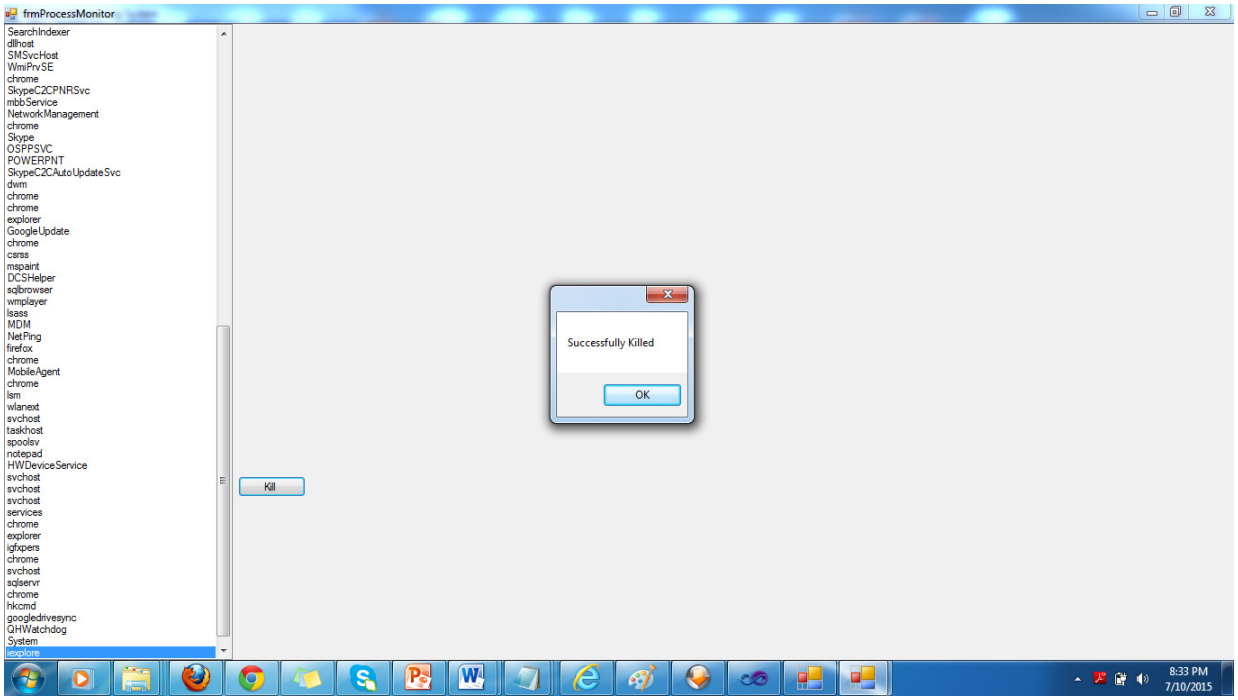

Figure 5.14: kill process which enter in bot that provide by support vector machine

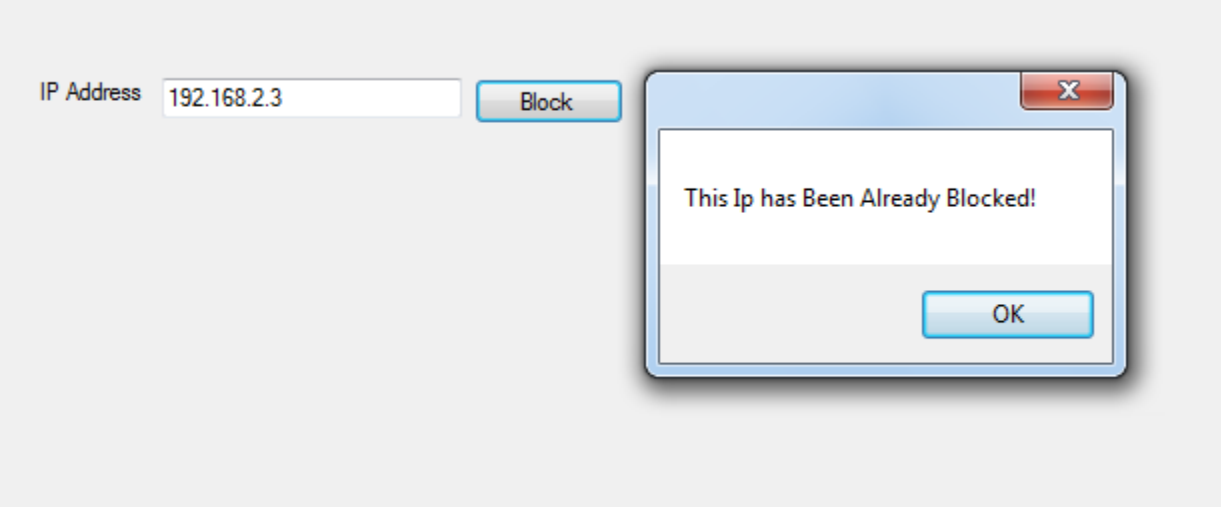

Figure 5.15: IP address already in SVM container

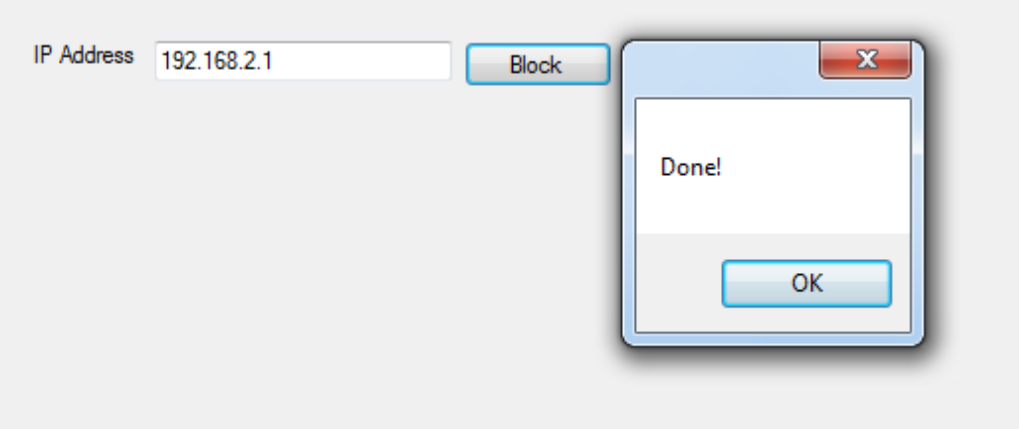

Figure 5.16: IP contain in SVM cluster

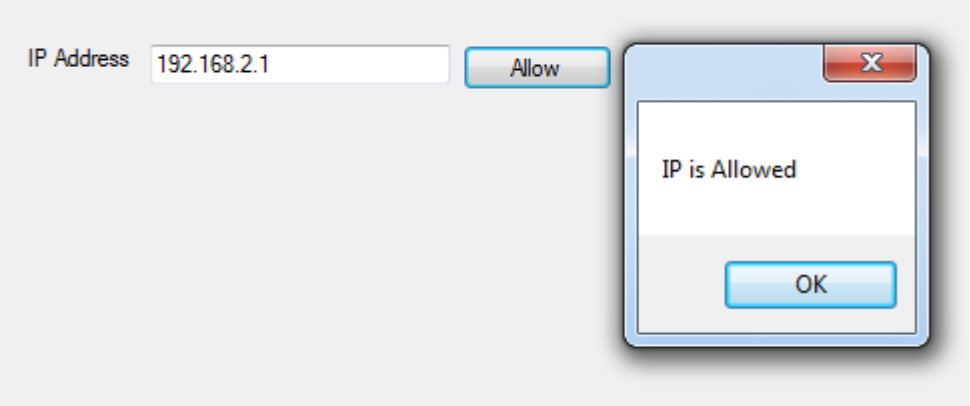

Figure 5.17: SVM ready to allow to execute the directory

We syndicate to application which is window application and another web application, window application will monitor the admin side and intruder movement and remaining intruder activity rest web application which is www, major activity timely updating on admin side.

CHAPTER 6

CONCLUSION
AND
FUTURE WORK

CONCLUSION AND FUTURE WORK

6. 1 CONCLUSION AND FUTURE WORK

Botnet detection based on machine learning SVM classifier has been the subject of interest of the research community resulting in the numerous detection methods that are based on different botnet heuristics that target different types of botnets using diverse machine learning algorithms and that consequently provide varying performances of detection. This Thesis presents a review of some of the most prominent contemporary botnet detection methods that use machine learning as a tool of identifying botnet-related traffic. The presented study addresses 20 detection methods, proposed over the last decade. The methods have been analyzed by investigating bot-related heuristic assumed by the detection systems and machine learning techniques used in order to capture botnet-related knowledge. Furthermore, the methods have been examined by analyzing their characteristics, performances, and limitations. The analysis of these detection approaches indicates a strong ability of this class of approaches to be used for identifying botnet network traffic. However, the study also indicates several aspects of machine learning-based approaches that could be further improved.

In addition, our proposed algorithm also outperformed the state-of-the-art CW algorithm when applied to large-scale botnet data obtained from an ISP. Future work will include more detailed textual analysis of the botnets' domain names.

REFERENCES

[1] Zeidanloo, H.R.; BT Manaf, A.; Vahdani, P.; "Botnet Detection Based on Traffic Monitoring", International Conference on Networking and Information Technology, 2010.

[2] Hossein Rouhani Zeidanloo, Azizah BT Abdul Manaf et.al "A proposed framework to detect P2P Bots", IACSIT International Journal of Engineering and Technology, Vol.2, No.2, April 2010 ISSN: 1793-8236.

[3] M. Mangalindan, ―For bulk E-mailer, pestering millions offers path to profit, Wall Street Journal, November 13, 2002.

[4] Qian Xu, Evan Wei Xiang and Qiang Yang, ―SMS Spam Detection Using Non-Content Features publication in IEEE Intelligent Systems, Nov.-Dec. 2012 (vol. 27 no. 6)pp. 44-51.

[5] Vishnu Kumar Goyal, Dept. of Computer Engineering. "A Comparative Study of Classification Methods in Data Mining using RapidMiner Studio". International Journal of Innovative Research in Science & Engineering, April 2014.

[6] Sang Min Lee, Dong Seong Kim, Ji Ho Kim, Jong Sou Park, ―Spam Detection Using Feature Selection and Parameters Optimization, pp. 883-888, 2010,IEEE.

[7] Ram B. Basnet, Andrew H. Sung, ―Classifying Phishing Email Using Confidence-Weighted Linear Classifiers, pp. 108-112, 2010 IEEE.

[8] Keisuke Takemori1, Masakatsu Nishigaki2 et.al, "Detection of Bot Infected PCs Using Destination-based IP and Domain Whitelists during a Non-operating Term", IEEE, 2008

[9] L.N. de Castro, J. Timmis. "Artificial Immune Systems: A New Computational Intelligence Approach" Springer, 2002.

[10] Fu, H., Yuan, X. and Hu, L. “Design of a four-layer model based on danger theory and AIS for IDS”, International Conference on Wireless Communications, Networking and Mobile Computing. IEEE, 2007.

[11] Mohsen Damshenas, Ali Dehghantanha, Ramlan Mahmoud. “A survey on malware propagation, analysis and detection”. International Journal of Cyber-Security and Digital Forensics (IJCSDF), 2013.

[12] Vinod P. “Survey on Malware Detection Methods”. Department of Computer Engineering, Malaviya National Institute of Technology.

[13] Nwokedi Idika and Aditya P. Mathur. “A Survey of Malware Detection Techniques”. Research supported by Committee on Institutional Cooperation, February 2007.

[14] B.V.R.R.Nagarjuna, V. Sujatha. “An Innovative Approach for Detecting Targeted malicious E-mail”. International Journal of Application or Innovation in Engineering & Management (IJAIEM), Volume 2, Issue 7, July 2013.

[15] Ion Androutsopoulos, John Koutsias, Konstantinos V. Chandrinos, and Constantine D. Spyropoulos. “ An experimental comparison of naive Bayesian and keyword-based anti-spam filtering with personal e-mail messages”. In Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval, Aug 2000.

[16] S. Hofmeyr, S. Forrest, “Architecture for an artificial immune system”, Evolutionary Computation. 2000. 7 (1) 1289–1296.

[17] Sang Min Lee, Dong Seong Kim, Ji Ho Kim, Jong Sou Park, ―Spam Detection Using Feature Selection and Parameters Optimization□, pp. 883-888, 2010,IEEE.

[18] Tao Wang , Shun-Zheng Yu “Centralized Botnet detection through traffic aggregation”, IEEE International Symposium on Parallel and Distributed Processing with Applications, 2009.

[19] Peng Wan, Uehara, ―Multiple Filters of Spam Using Sobel Operators and OCR□ IEEE, pp.164-169,July,2012

[23] Kirti Mathur. “A Survey on Techniques in Detection and Analyzing Malware Executable”. International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 4, April 2013.

[24] Milan Jain. “Malicious Detection Using Multiple Classification Algorithms & Their Comparison Using Different Clustering Techniques”. International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 4, April 2013.

[25] Oda, T.White, T. “Increasing the accuracy of a SPAM-detecting artificial immune system”. In: Proceedings of the IEEE CEC, 2003, vol. 1, pp. 390 396.

[26] Anton Borg, Niklas Lavesson, ―E-mail Classification using Social Network Information Seventh International Conference on Availability, Reliability and Security,IEEE,2012

[27] M. McCord and M. Chuah, ―Spam Detection on Twitter Using Traditional Classifiers pp. 175–186, 2011.© Springer-Verlag Berlin Heidelberg 2011

[28] Zhang, Y., Li, H., Niranjan, M., and Rockett, P. (2008). Applying cost-sensitive multi objective genetic programming to feature extraction for spam e-mail filtering. In Proc. of the 11th European conference on Genetic programming, pages 325–336. Springer-Verlag.

[29] Dudley, J., Barone, L., and While, L. (2008). Multi objective spam filtering using an evolutionary algorithm, pages 123–130. IEEE.

[30] Lourdes Araujo and Juan Martinez-Romo, ―Web Spam Detection: New Classification Features Based on Qualified Link Analysis and Language Models IEEE Transactions On Information Forensics And Security, Vol. 5, No. 3, September 2010

[31] J. Greensmith, U. Aickelin, and S. Cayzer.” Introducing dendritic cells as a novel immune-inspired algorithm for anomaly detection”, In ICARIS-05, LNCS 3627, pages 153–167, 2005.