Detection of P2P Botnets Based on Support Vector Machine: Case Study

Dr. Nemir Ahmed Al-Azzawi
Engineering College, University of Baghdad / Baghdad
mailto:N_azzawi@yahoo.com
Dr. Shatha Mizhir Hasan
Iraqi Commission for Computers and Informatics

Received on: 3/4/2013 & Accepted on: 15/8/2013

ABSTRACT:

Botnet is a general term referring to a group of automated software robots that run without human intervention (malware code). Nowadays, Botnets produces a major threat to the cyber security (Information Assurance) of computing assets. Therefore, you need to protect our huge confidential and personal information through the use of web interfaces such as online passwords, corporate secrets, online banking accounts, and social networking accounts like Facebook. Network traffic analysis is an important component in the management and security of current networks and in the design and planning on future networks. This study enables the researcher: (a) to study botnet topologies, behavior and lifecycle events and actions (b) to combine normal web traffic and normal P2P traffic for binary classification; (c) to produce simulated network flow data comparable to the activities of a botnet controller or "bots," and hosts under attack (testing samples); and (d) to detection and identifies P2P botnet framework using Support Vector Machine (SVM) based on statistical features.

Keywords: Botnet, Peer to peer (P2P); Support vector machine (SVM).

كشف الروبوتات في إطار P2P باستخدام آلة داعم المتجهات (SVM) : دراسة حالة

الخلاصة:

الروبوتات هو مصطلح عام يشير إلى مجموعة من الروبوتات الآلية البرمجة التي تعمل دون تدخل بشري (برمجة ضارة). في الوقت الحاضر، فان هذه الروبوتات تنتج خطرا كبيرا على أمن المعلومات الموجودة في الحوسبة. لذلك، نحتاج إلى حماية المعلومات السرية والشخصية الضخمة التي لدينا من خلال الاستخدام الاعتيادي لواجهات الويب، كمثل على ذلك كلمات المرور على الانترنت، أسرار ومعلومات الشركات، الحسابات المصرفية عبر الإنترنت، الشبكات الاجتماعية مثل حسابات الفيسبوك. أن تحليل شبكة حركة المرورهي عنصر أساسيفي إدارة أمن الشبكات الحالية، وكذلك في تخطيط وتصميم شبكاتالمستقبل.هذه الدراسة تمكن الباحث من: (أ) دراسة دورة حياة الروبوتات أحداث وأفعال، (ب) الجمع بين حركة المرور العادي على الشبكة، وحركة المرور P2P وذلك للتصنيف الثنائي، (ج) إنتاج محاكاة تدفق بيانات الشبكة للأنشطة المماثلة لتدفق سير الروبوتات الموجه والمستضيف المصاب (عينات الاختبار)، (د) الكشف

https://doi.org/10.30684/etj.32.5A.12
2412-0758/University of Technology-Iraq, Baghdad, Iraq

وتحديد الروبوتات في إطارP2P باستخدام آلة داعم المتجهات (SVM) على أساس السمات الإحصائية.

INTRODUCTION

Today, our reliance on the internet has developed multiple. So has the need to guard our vast personal information accessible via web interfaces such as online passwords, corporate secrets, online banking accounts, and social networking accounts like Facebook.

The appearance of botnets in the internet scene over the last decade, and their ever changing behavior has caused real challenges that cannot be easily remedied. According to literature, a botnet is defined to be a set of infected hosts (also called bots or zombies) that run autonomously and automatically, controlled by a bot master (bot herder) who can co-ordinate his/her malicious intentions using the infected bots. Some of the prominent malicious tasks that can be credited to botnets include DDoS (Distributed denial-of-service), spam, phishing, ransom wares and identity theft. in Figure 1, in regards to how each actor is contributing to the botnet problem (or alternatively, mitigating it); and how botnets are in return, affecting that actor.

In the beginning, most botnets used a centralized approach for managing botnets [1]. This was done using the IRC (internet relay chat) protocol or modified versions of the protocol using freely available sources such as Unreal IRCd[2]. As per [2], the main reasonsfor using IRC were its interactive nature for two way communication between server-client; readily available source code for easy modifications; ability to control multiple botnets using nicknames for bots and password protected channels; and redundancy achieved by linking several servers together.

A botnet can be defined as a coordinated group of compromised machines controlled via Command & Control (C&C) communication channels that are connected to some C&C servers/peers managed by botmasters/botherders. Bots in a botnet are generally involved in performing various malicious activities like sending spam mails, distributed denial-of-service (DDoS) attack, phishing and click fraud. In the centralized approach IRC servers are often used to manage Command and Control functions to allow the botmaster to control the collection of compromised computers and issue commands. The prevalence of IRC servers (which use a centralized C&C model) and their simplicity of use continue to make them an attractive mechanism for botnet control (Bacher et al., 2005). A typical botnet using IRC for Command and Control is illustrated in Figure 2 [3]. Most common botnet topologies available in wild are either centralized or distributed in structure. Most early botnets are based on centralized C&C architecture (e.g. IRC). Botnets with centralized C&C suffer from single -point-of-failure problem i.e. if C&C is detected and taken down the botnet cripples. However, IRC botnets with their source codes widely available and their setup and maintenance simple and relatively easy, are still most popular among bothered[4]. Some recent botnets have used distributed C&C architecture (e.g. P2P), mainly to avoid single-point-of-failure problem. Also, newer P2P botnets are using advanced techniques like Root-kits, Fast-flux etc. to avoid detection [5]. Use of a distributed C&C infrastructure by the botnet operators started in the year 2007 with the introduction of Storm [6] botnets, followed by Nugache [6] and more recently Waledac[7]. While Storm spreads via legitimate hosts running Overnet clients, Nugache spreads over AIM (AOL Instant Messenger). All these three bots carry a list of P2P hosts used as “first point of contact”. While Nugache

always opens TCP port 8 to establish communication, Storm uses a protocol called Overnet for communication, which is based on Kademlia theories.

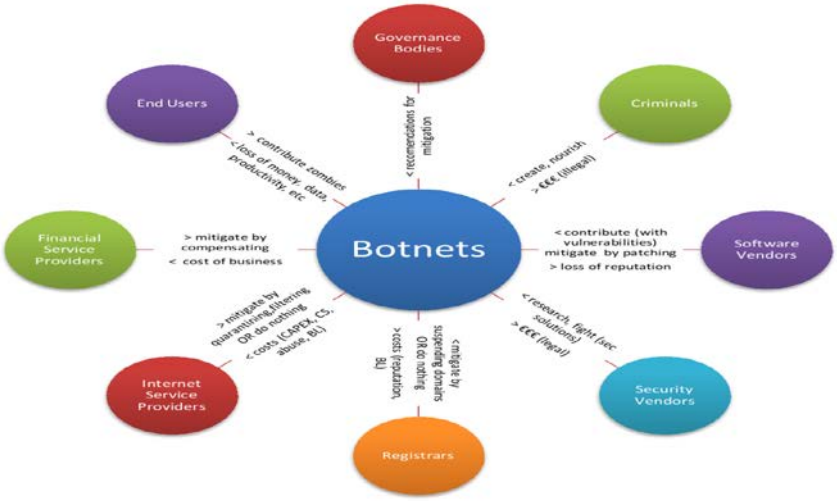


Figure (1) Simplified diagram of relations of online actorst the botnet problem.

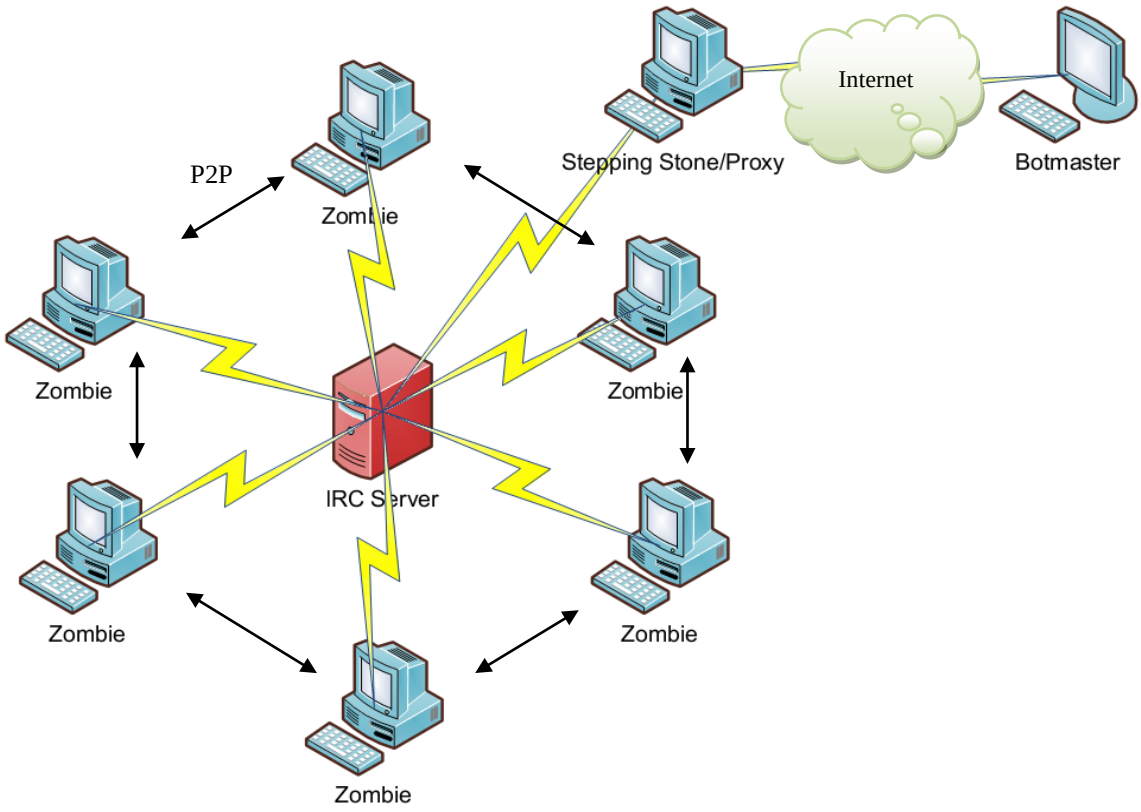


Figure (2) P2P and IRC Botnet Structure [3].

Waledac utilizes HTTP communication and a fast-flux based DNS network exclusively in its communication channels. SVM being used as the sole classifier in this paper mainly because SVMs are known to perform better than other classifiers in many critical respects, such as SVMs scale much better and SVMs give higher classification accuracy [8].

In this paper, we propose a proactive approach for P2P botnet detection i.e. it can detect botnets prior to they are being used for any malicious activity. We have used support vector machines (SVM) for classification of large volume of control traffic generated by a P2P botnet from that of normal web traffic. We have carried out the investigation based on assumption that for a carefully selected set of flow features, P2P botnet control traffic can be distinguished from legitimate P2P and other normal web traffic.

The actual feature set used to build a classifier is significantly important in determining its performance. Therefore, we propose a feature selection technique and provide insights into which features are most important for effective classification. Furthermore, we investigate the performances of SVM kernels, namely RBF kernel and also provide insights in to the best combination of parameter values in terms of classification accuracy. Rest of the paper is organized as follows: Section II provides a brief overview of related works. In Section III discuss the network Behavior examination Tools. In section IV, we discuss the architectural overview of our proposed detection approach and the approach we have followed for dataset preparation. Section V describes the feature selection technique we have formulated. In last section carries an analysis of the result and finally we conclude.

RELATED WORK

The research carried out till date for botnet detection and mitigation is simply inadequate, compared to huge threat that the botnets have evolved to over the years. Most of the botnet detection approaches are based on the anomalies being observed in network traffic, unusual system behavior etc. Botnet detection based on anomalies may not be useful always for several reasons. First, anomalies may not be always prominent to indicate a botnet attack. Second, it requires continuous monitoring of the network. Third, traffic belonging to botnets using HTTP protocol hides under the cover of normal web traffic and thus gets allowed everywhere. Because of these reasons, many researchers have used Machine Learning based approaches for botnet detection. Several techniques have been developed for classification and detection of botnet traffic. In paper [9], Livadas et. al. provided a two stage proactive approach to distinguish C&C traffic from normal traffic. First, it distinguishes IRC based chat flows from that of non-chat flows and then botnet infected IRC flows from that of real IRC flows. Their technique uses network flow classification based on ten flow characteristics. Among the machine learning tools, J48, naïve Bayes and Bayesian Network classifiers are used. While these are effective technique for identifying botnet C&C traffic, they are specific to IRC based C&C mechanism. Paper [8] also focuses on a similar proactive measure for detecting P2P botnet using five machine learning techniques, namely, Support vector Machine (SVM), Artificial Neural Network (ANN), Nearest Neighbors Classifier (NNC), Gaussian-Based Classifier (GBC), and Naives Bayes Classifier (NBC). In this model they have used information about payload size, number of packets, duplicated packets length, and concurrent active ports to build the features set. While they have used both network flow level features and features that represent host communication patterns, our approach is solely based on flow

features, as dependence on host behavior may be misleading given the complex communication pattern of P2P botnets. Analysis of their result shows that SVM produces best performance in terms of detection rate of Botnet C&C flows. In the paper [10] Wen-Hwa et. al. applied research on the original dissimilarity of P2P botnet differing from normal Internet behaviors as parameters of data mining, which were then clustered and distinguished to obtain reliable results with acceptable accuracy. By observing C&C communication pattern that characterize P2P botnets based on statistical fingerprints of the P2P communications, paper [11] proposes a botnet detection approach for detecting stealthy P2P botnets. Even though it can detect bots that are not involved in any malicious activity, it requires constant judging of activities of nodes in a monitored network. In [12], Thuraisingham et. al. have proposed a stream data classification algorithm for detection of P2P botnet. Based on two important properties of botnet traffic i.e. infinite length and concept drift, they have proposed a multi-chunk, multi-level ensemble classifier to classify concept-drifting stream data. In paper [13]Masud et. al. have proposed a flow based approach to classify C&C and normal flow using data mining to learn temporal correlation between an incoming packet and one of the following logged events: an outgoing packet, a new outgoing connection and an application startup. Any incoming packet correlated with one of these logged events is considered a possible botnet command packet. Classification was done using Support Vector Machine (SVM), Bayes Net, Decision tree (J48), Naïve Bayes and Boosted decision tree (J48).

NETWORK BEHAVIOUR EXAMINATION TOOLS

The first part of the study included understanding botnet behavior and predicting its life stages. Among the topics covered under the research were ICMP, HTTP, TCP and UDP protocols, botnet topologies, botnet behavior and lifecycle, prediction of botnet actions and general traffic characteristics. The attacker is using IP spoofing to include a random source IP address in the header and hence is trying to make identification of the source of attack impossible. The source IPs might not belong to the attacker's network. The goal was to define the data flow and state transitions for each phase of a botnet's operation.

A typical botnet has three major lifecycle stages:

1. Infection Stage
2. Recruitment Stage
3. Attack Stage Each stage is controlled by the botmaster through the C&C system.

During the infection stage, vulnerability in a potential bot is identified and used to infect it. In the recruitment stage, each infected system recruits (infects) other systems forming a network of bots. Once the botnet is established it may be used to execute other attacks. Though the pattern of attack may be different for different botnets, all botnets exhibit these three basic phases in their life cycle.

Some general characteristics of network traffic after a host infection:

1. The communication pattern between hosts is abnormal: For example: a) TCP SYN packet floods without any corresponding ACK packets b) ICMP echo request/reply packet floods.
2. The number of incoming connections is very large: For example: a) large amount of ICMP packets sent to a broadcast address with the IP address of the victims the source address. This make all the hosts in the network reply to victim

with ICMP reply packets. This is a smurf attack[14]. b) A rise in the number of incoming C&C requests as bots connect to IRC server after infection.

3. An unusual number of outgoing packets: For example: bots carrying out email spamming and denial of service attacks may show this characteristic.

4. DNS queries to unusual servers and establishing other communication with the IP address returned.

5. Hosts trying to connect using unusual ports or protocols. For example, if the network in question is never expected to use the IRC protocol and IRC transactions are observed.

6. Virus or worm signatures observed in packets entering or leaving a network boundary. Using the approach we developed, one could identify additional characteristics similar to the ones listed above.

This section provides an overview of some of these technologies, as well as references for further investigation. We cover these in an order that approximates their level of abstraction from the traffic: from the packet traces on one extreme to high-level summaries and visualizations on the other.

A. **WireShark**

WireShark [2] is a basic open-source packet sniffer and analyzer. It sniffs the network traffic visible from an interface that has been placed in promiscuous mode, records it to disk, and displays the packet traces in a linear format to the user. It has many built-in protocol parsers to add some semantic information to the data displayed, but does little else. Figure 3, shows Wireshark userinterface.

B. **Network Analysis visualization**

Network Analysis Visualization (NAV) is a project of the University of British Columbia's Department of Computer Science with the aim of gaining a higher level of understanding of network events than that provided by linear views of packet traces [18].

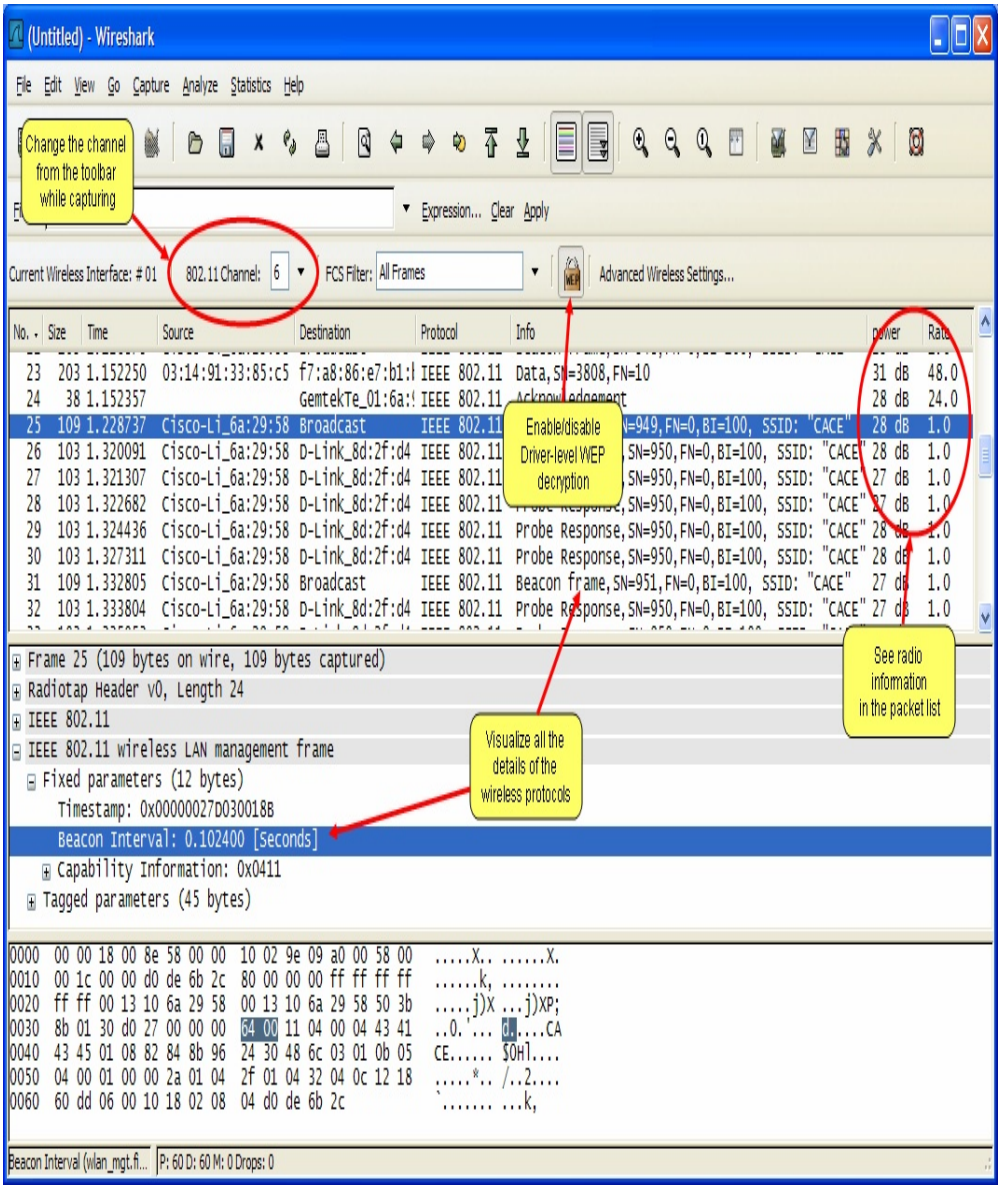
NAV provides a bipartite visual overview of traffic at the border of the network to help administrators understand local hosts' interactions with remote ones. It is therefore particularly well-suited to visualizing otherwise benign phenomena that gain the potential for mischief when they arc correlated across many hosts, such as (once again) portscan behavior. Figure 4, shows overviewofNAVInterface.

C. **Cisco NetFlow**

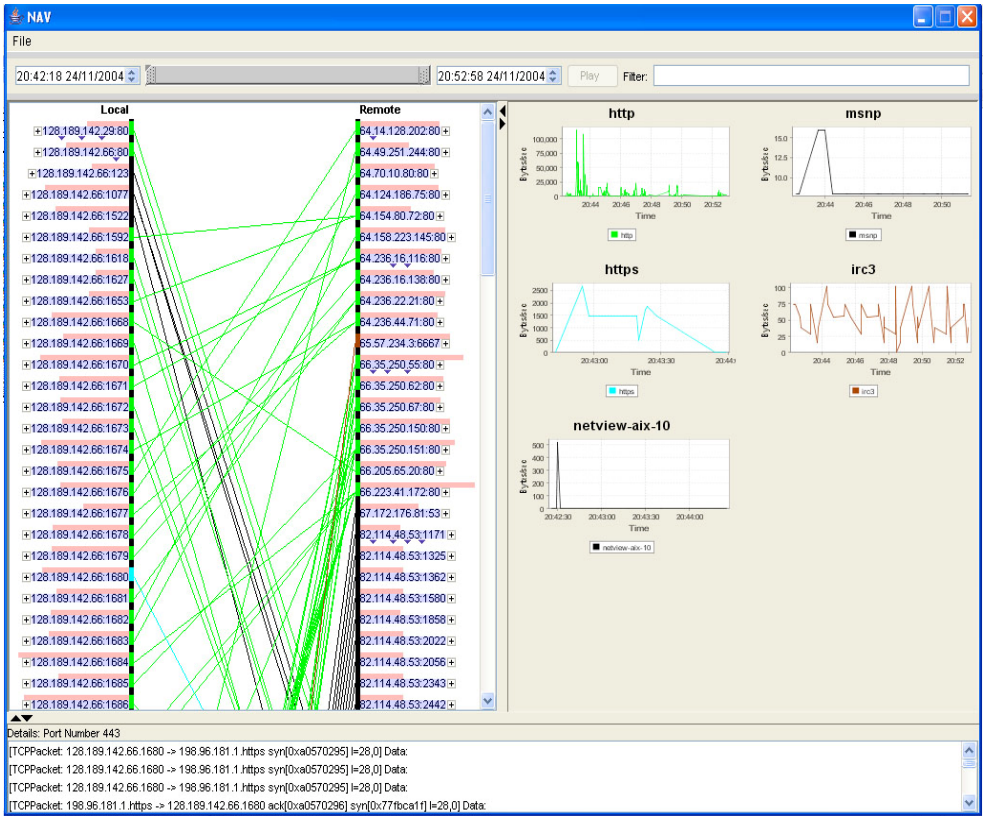
Cisco NetFlow is a Cisco developed flow technology that allows bandwidth monitoring of a network. NetFlow Analyzer is a software that uses Cisco NetFlow to monitor bandwidth and runs in windows and linux. In the Cisco Systems NetFlow v9 specication[15], a compliant exporter may export a combination of source address, destination address, source port number, destination port number, the number of packets exchanged in each direction, the number of bytes exchanged in each direction, a record of TCP ags, and more. Once NetFlow records and pcap files were available, the output data description of network characteristics to understand the behavior observed. Figure 5, shows overviewofNetFlowInterface.

A wide variety of tools and technologies have been developed to help understand complex behaviors in networks. Once the behavior of a botnet was understood, our goal was to generate botnet-like traffic, which we could attempt to analyze, thereby paving the way to building detection algorithms. This data would be used to study the particular characteristics of botnet behaviors, which once identified, could be used to develop and test detection algorithms. We could either design a simulator to generate NetFlow records traffic or find a generator from the net which could be reused for traffic generation. Since the objective of the study was to analyze botnets

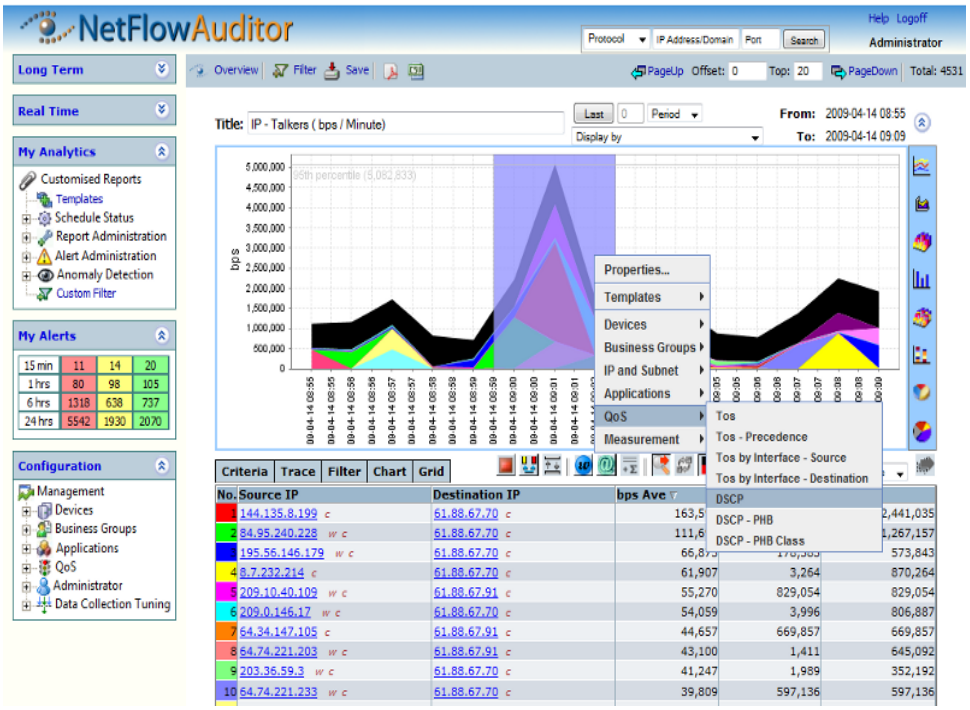
and find methods to detect and mitigate them, the team decided to look for an existing tool that would generate synthetic simulated NetFlow records traffic in large quantities. The team evaluated several off-the-shelf NetFlow generator tools and selected the NetFlow Traffic Generator(NetFlow Packet Generator) from Virtual Console [16] for simulating NetFlow traffic. It was also found that one could use freely available pcap files, with NetFlow records [17-18]. These data are categorized as good (normal) or bad (malicious), which helped facilitate our study. We used WireShark® (see Figure 3) to inspect data from live network traffic, and downloaded from publicly available data sources (pcap files). WireShark has the ability to track real-time network traffic and its characteristics. Alternately we could download pcap files and open them with WireShark to analyze.



Figure(3)Wireshark User Interface.



Figure(4) Overview of NAV Interface



Figure(5) Overview of Net Flow Interface

ARCHITECTURAL OVERVIEW AND DATASET PREPARATION

The architectural overview of our experimental set up for P2P botnet detection is shown in Figure 6. We acquired the benign data randomly from windows machines using WireShark. We kept WireShark running for long durations so that enough of data are captured to represent the flows.

We created a Perl script to extract flows from the raw data captured via WireShark. The benign web traffic we used includes flows that contain general network traffic such as HTTP, FTP, SMTP etc. and the normal P2P flows that contain legitimate P2P traffic such as Bittorrent, Skype, and e-Donkey.

While preparing the dataset, we have discarded flows that are unlikely to contribute significantly in the process of classification as follows.

(i) Flows having single packet, as it does not carry any meaningful information.

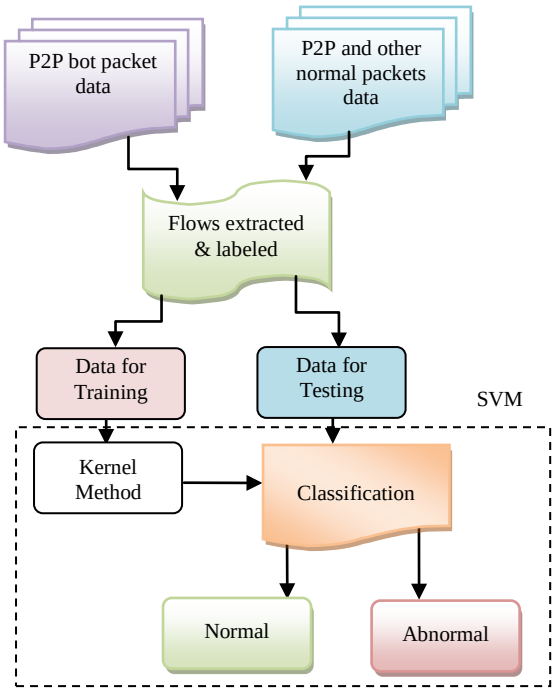
(ii) Flows that involves local broadcast activities in the network.

(iii) The remaining of the flows (including both normal and botnet data) are labeled and combined to form a single dataset.

(iv) The dataset prepared for conducting the experiments has a total of 20000 flows of which 10% are normal flows. The P2P bot keeps changing the communication ports frequently, making the number of bot flows much larger compared to normal flows.

FEATURE SELECTION

Feature selection is an important issue that affects the accuracy of detection. The question of identifying useless, less significant and truly useful features is relevant because the elimination of useless features (or audit trail reduction) enhances the accuracy of detection while speeding up the computation, thus improving the overall performance of the detection mechanism. In cases where there are no useless features, concentrating on the most important ones may well improve the efficiency of the detection mechanism, SVM Classifier without affecting the accuracy of detection in statistically significant ways. A model selection of SVM [8] being carried out using radial basis function (RBF) kernel [19]. It is given by $K(x_i, x_j) = exp(-\gamma||x_i - x_j||2), \gamma > 0$. We considered RBF Kernel, mainly for two reasons: (a) the number of hyper-parameters, which influences the complexity of model selection. The polynomial kernel has more hyper-parameters than RBF kernel; (b) the RBF kernel has fewer numerical difficulties. There are two parameters for a RBF kernel: C and γ (gamma). Since it is not known beforehand, which pair of value for C and γ would produce best result, we performed a kind of model selection (parameter search).



Figure(6) FLOW BASED P2P BOT DETECTION ARCHITECTURE USING (SVM)

We have considered the flow features based on following observations:

(i) Flows belonging to legitimate P2P networks (e.g. file sharing networks) normally carries packet to the size of MTU (Maximum Transmission Unit). Whereas, bots are malicious programs and are valuable asset for botmaster. Therefore, to avoid getting detected they try to keep the size of packets as small as possible, almost never reaching the size of MTU.

(ii) For the same reason as stated in (i), the botmaster keeps the ratio of largest packets in a flow to as low as possible which is just the opposite in case of legitimate P2P networks.

(iii) Bots are malicious programs that keep changing communication ports, which results in less bytes being transferred with each flow. Whereas, most of the legitimate P2P networks are file sharing networks that results in large amount of bytes being transferred with each flow.

(iv) Since bot flows are generated by programs, variances with respect to different flow parameters like packet sizes and inter arrival time of packets are expected to be low compared to legitimate P2P flows. Data collected for plotting the graphs are an average of per thousand flows for bot and normal data taken separately. A result of our analysis shows that the features are ideal for classification.

We have also used a simple performance-based input ranking methodology: One input feature is deleted from the data at a time; the resultant data set is then used for the training and testing of the classifier. Then the classifiers performance is compared to that of the original classifier (based on all features). Finally, the importance of the feature is ranked according to difference in performance produced by deletion of a specific feature. The more is the difference the higher is the rank of the deleted feature.

The procedure is summarized as follows:

1. Compose the training set and the testing set;
2. Use the complete data set to train the classifier and evaluate the performance of the classifier using the test set. repeat step 3 through step 6 until no feature is left out
3. Select a feature to delete from the (training and testing) data; If the feature is not selected earlier then do the following
4. Delete the feature and use the resultant data set to train the classifier;
5. Analyze the performance of the classifier using the test set.
6. Rank the importance of the feature by comparing its test result to the result obtained in step2.

EXPERIMENTATION AND MODEL SELECTION

Once the enormous volume of TCP and UDP traffic generated by the P2P bot are segmented into flows as described in section III and feature selection carried out using the procedure described in section IV, we performed a model selection [19] using Support Vector Machine. We used Sequential Minimal Optimization (SMO) algorithm in WEKA machine learning environment to carry out the classification. In order to obtain best (C, γ), we performed a "grid-search" on C and gamma using cross validation. For this purpose we tried with an exponentially growing sequences of C and γ, like $C = 2^5, 2^7, 2^9, \dots, 2^{13}$ and $\gamma = 2^{-5}, 2^{-3}, 2^{-1}, \dots, 2^1$. We use n-fold cross validation in our experiments. In n-fold cross validation, we first divide the training set into n subsets of equal size. Sequentially one subset is tested using classifier trained on remaining n-1 subsets. Thus, each instance of the whole training set is predicted once so the cross-validation accuracy is the percentage of data which are correctly classified. In general, the value of n does not influence the cross validation accuracy much if it is small enough compared to total number of samples. In our experiments we set n=10. Following steps were performed to do the classification:

1. SVM require that each data instance is presented as a vector of real numbers. Consequently, all data instances are presented as vectors of real numbers.
2. Each attribute is linearly scaled to the range [-1, +1]. This is done in order to avoid numerical problems that may arise for large attribute values, because the kernel values usually depend on the inner products of feature vectors.
3. In this step we performed model selection i.e. the Kernel is chosen along with the value for penalty parameter C and Kernel parameter γ.

Table I shows the percentage of accuracy for exponentially increasing sequences of penalty parameter C and kernel parameter γ.

TABLE(1): PERCENTAGE OF ACCURACY FOR DIFFERENT COMBINATION OF C AND γ.

	Percentage of accuracy
$C = 2^5, \gamma = 2^{-5}$	94.022
$C = 2^7, \gamma = 2^{-3}$	95.432
$C = 2^9, \gamma = 2^{-1}$	94.844
$C = 2^{11}, \gamma = 2$	99.262
$C = 2^{13}, \gamma = 2^1$	96.906

CONCLUSIONS

This study was a part of ongoing research on the behavior of botnets to find new ways to detect and mitigate malicious activities. Since we were the team working on the initial iteration of the project, our scope was limited to study of the botnet lifecycle and to determine a methodology to analyze the behavior of botnets as observed in data traffic captures. A botnet detection based on flow features of P2P bot has been introduced. The flow diagram for our process is shown in Figure 6. By using the approach we developed as a framework, this effort to study the observable traffic characteristics of malicious botnets. We hope that the results of these efforts will be algorithms to detect botnet traffic and alert network operators. The experimental result shows that our optimized method yields approximately between (94.02% ~ 99.26%) accuracy for unbiased training.

REFERENCES

[1] P. Bacher, *et al.*, "Know your Enemy : Tracking Botnets using Honeynets to learn more about bots," Honeynet Project, pp., 2005.

[2] Unrealircd. (2012). *Customized IRC Solutions, Unreal 3.2.9 released*. Available: http://www.unrealircd.com/, Last accessed: 12 June, 2012.

[3] D. Dittrich and S. Dietrich, "Command and control structures in malware: From Handler/Agent to P2P," *USENIX*, 32(6), pp. 8-17, December 2007 2007.

[4] M. Xiaobo, *et al.*, "A Novel IRC Botnet Detection Method Based on Packet Size Sequence," in *Communications (ICC), 2010 IEEE International Conference on*, 2010, pp. 1-5.

[5] K. Jian and Z. Jun-Yao, "Application Entropy Theory to Detect New Peer-to-Peer Botnet with Multi-chart CUSUM," in *Electronic Commerce and Security, 2009. ISECS '09. Second International Symposium on*, 1, 2009, pp. 470-474.

[6] E. Florio and M. Ciubotariu, "Peerbot: Catch me if you can, Symantec Security Response," pp., April 2007.

[7] G. Sinclair, *et al.*, "The waledac protocol: The how and why," in *Malicious and Unwanted Software (MALWARE), 2009 4th International Conference on*, 2009, pp. 69-77.

[8] S. Saad, *et al.*, "Detecting P2P botnets through network behavior analysis and machine learning," in *Privacy, Security and Trust (PST), 2011 Ninth Annual International Conference on*, 2011, pp. 174-180.

[9] C. Livadas, *et al.*, "Usilng Machine Learning Technliques to Identify Botnet Traffic," in *Local Computer Networks, Proceedings 2006 31st IEEE Conference on*, 2006, pp. 967-974.

[10] L. Wen-Hwa and C. Chia-Ching, "Peer to Peer Botnet Detection Using Data Mining Scheme," in *Internet Technology and Applications, 2010 International Conference on*, 2010, pp. 1-4.

[11] Z. Junjie, *et al.*, "Detecting stealthy P2P botnets using statistical traffic fingerprints," in *Dependable Systems & Networks (DSN), 2011 IEEE/IFIP 41st International Conference on*, 2011, pp. 121-132.

[12] B. Thuraisingham, "Data mining for security applications: Mining concept-drifting data streams to detect peer to peer botnet traffic," in *Intelligence and Security Informatics, 2008. ISI 2008. IEEE International Conference on*, 2008, pp. xxix-xxx.

[13] M. M. Masud, *et al.*, "Flow Based Identification of Botnets Traffic by Mining Multiple Log Files," in *In Proc. Distributed Framework and Applications*, 2008.

[14] (1998). *CERT® Advisory CA-1998-01 Smurf IP Denial-of-Service Attacks*. Available: http://www.cert.org/advisories/CA-1998-01.html Last accessed.

[15] B. Claise. (2004). *Cisco systems netFlow services export version 9. RFC 3954 (Informational)*. Available: http://www.ietf.org/rfc/rfc3954.txt, Last accessed: 18 Jan. 2012.

[16] N. P. Generator. Available: http://www.vconsole.com/client/?page=page&id=21, Last accessed.

[17] G. Combs, *et al.* (2012). *Wireshark is the world's foremost network protocol analyzer*. Available: http://www.wireshark.org/, Last accessed: 10-02-2012.

[18] sourceforge.net. (19 February 2012). *Publicly available PCAP files*. Available: http://sourceforge.net/apps/mediawiki/networkminer/index.php?title=Publicly_available_PCAP_files, Last accessed: 24-04-2012.

[19] C.-W. Hsu, *et al.*, "A practical guide to support vector classification," Department of Computer Science, National Taiwan University, Technical report, pp. 1-12, July 2003.