ISSN:0975-9646

BOTNET Detection Approach by DNS Behavior and Clustering Analysis

Vartika Srivastava, Ashish Sharma
Dept of Computer science and Information security,
JIIT Noida, India

Abstract -Botnets are one of the most serious threats to internet security. A botnet is a network of computers on internet which are under the influence of a malware code, oblivious to the owner of that computer and sends out transmissions (virus or spam) to other computers on internet. Botnet can be utilized for DoS attacks, phishing, spamming and many other fraudulent activities. Therefore, it is important to detect botnets. In this paper we will describe the strategy for botnet detection by detecting the fast flux characteristics of a botnet. Through fast flux bot-master DNS uses different IP addresses so that no one would be able to detect the actual physical location of the server of the botnet. Thus, in our approach we are using K-means clustering to the DNS data to detect the fast flux.

Categories and Subject Descriptors
K.6.5.a [Management of Computing and Information Systems]: Security and protection - authentication.

General Terms
DNS(Domain Name Services), DoS(Denial of Service), DDoS(Distributed Denial of Services).

Keywords
Clustering tool: Weka tool, K-means clustering.

1. INTRODUCTION

Botnet is a collection of software agents or robots that run autonomously and automatically. The term is most commonly associated with malicious software's, but it can also refer to a network of computers using distributed computing software. Also it can be said that a zombie army or number of internet computers that, although their owners are unaware of it, have been set up to forward transmission to other computers on the internet. Computers that are cooped to serve in a zombie army are often those whose owners fail to provide effective firewalls and other safeguards. A zombie or bot is often created through an Internet port that has been left open and through which a small Trojan horse program can be left for future activation. The computers that form a botnet can be programmed to redirect transmissions to specific computers such as a website that can be closed down by having to handle too much traffic – a DDoS attack, or in the case of spam distribution, to many computers. The motivation for a zombie master sending spam is in the money to be made. Both of them rely on unprotected computers that can be turned into zombie. They are considered to be most vulnerable security threats on the Internet due to anti-X algorithm that makes them highly dynamic and adaptive and also decrease the success rate of detection. Here to detect botnets we have monitored the network flow and done a clustering analysis to identify any malicious activities. We have also monitored the traffic flow at each node in the network and look at the IP requests it is generating. Looking at the kind of IP requests and using an effective clustering algorithm, we categorized all the IP requests into different groups. For example, the most popular IP requests can be grouped into one category while if there is a sudden burst of new IP requests which are targeting a new IP address and making large requests then they will be marked 'suspicious' and grouped differently. Same had been done with the other nodes also. This data was then compared to the data collected at the neighbouring nodes. If there was a strong correlation and overlap between the 'suspicious' data of different nodes then there was a high probability of existence of a botnet. To prevent harm, we can either stop the node from making that IP request or warn the user against it.

1.1 Lifecycle of a botnet

The life-cycle of a botnetstarts with a *Bot-Herder*. A *Bot Herder* is a cracker who uses automated techniques to scan specific network ranges and find vulnerabilities in system.

1. Bot-Herder configures initial bot parameters such as infection vectors, payload, stealth, C&C (command and Control), details
2. Register a DDNS (Dynamics Domain Name Service).
3. Register a static IP.
4. Bot-Herder launches or seeds new bots.
5. Bots spread.
6. Causes an increase of DDoS(Distributed Denial of Service) being sent to the victim.
7. Losing bots to rival botnets.

2. BACKGROUND

The existence of botnets was recognized several years ago, but the studies for defending botnets are still in an early stage. Some security companies and institutions have analysed the botnet traffic, the method of propagation and furthermore proposed the botnet detection and response mechanisms. However, their defence mechanisms are focused on the symptoms of abnormal network traffic and bot binary detections by matching with the signatures of known bot codes. Even though these are useful for many cases, they have inevitable limitations such that they are unable to detect new or modified bots.

There have been a few researches on the methodological analysis about the bot and botnet such as their behaviours, statistics, and traffic measurements. Jones [1] provided botnet background and recommendations so that network and system security administrators can recognize and

defend against botnet activity. Cooke *et al.* [2] outlined the origins and structure of bots and botnets, data from the operator community and study the effectiveness of detecting botnets by directly monitoring IRC communication or other command and control activity and show a more comprehensive approach is required. Barford *et al.* presented a perspective based on an in-depth analysis of bot software source code and reveals the complexity of botnet software, discusses implications for defence strategies based on the analysis [3]. Rajab *et al.* [4] constructed a multifaceted infrastructure to capture and concurrently track multiple botnets in the wild, and achieveda comprehensive analysis of measurements reflecting several important structural and behavioural aspects of botnets. They studied the botnet behaviour, botnet prevalence on the Internet, and modelling the botnet life cycle.

Recently, a few attempts have been made to cope with botnet problems and most of them have come to focus on detection of botnet. Bots are sending DNS queries in order to access the C&C channel server. If we could know the name of domain name of C&C channel server then we can blacklisting the domain name for sinkhole techniques to capture the botnet traffic and measure the botnet. Dagon *et al.* [5] identified key metrics for measuring the utility of a botnet, and describe various topological structures botnet may use to coordinate attacks. And using the performance metrics, they consider the ability of different response techniques to degrade or disrupt botnets. Their study used DNS redirection to monitor botnets. However our approach does not need any DNS redirection and communication with any component of botnet. Dagon also present botnet Detection and response approach [6] with analysing peculiarity of botnet rallying DNS traffic (particularly, measuring canonical DNS request rate and DNS density comparison). However the detection technique could easily be evaded when bot-masters know the mechanism and poisoned by using faked DNS queries. Kristoff [7] also suggested a similar approach, but the mechanism has the same weakness.

Binkley [8] proposed an anomaly-based algorithm for detecting IRC-based botnet meshes. The algorithm combines an IRC mesh detection component with a TCP scan detection heuristic called the TCP work weight. They can detect IRC channel with high work weight host but some of them could not be a member of botnet (false positive), additional analysis for many borderline cases as they mentioned in the paper. Ramachandran [9] developed techniques and heuristics for detecting DNSBL reconnaissance activity, whereby bot-masters perform lookups against the DNSBL to determine whether their spamming bots have been blacklisted. This approach of botnet detection is derived from novel idea that detect DNSBL reconnaissance activity of bot-master but also have false positives and some defects that is referred in their paper. Botnets are constructed and managed in several stages such as bot infection, C&C server rallying, and other types of malicious activities. Defense against botnet attacks seems to be a very complicated task. Only a few of works have been done in this area, but we need further improvements for the purpose of practical use. Moreover, previous works are difficult to be used for finding all types of botnet because the botnet have complex behaviour patterns.

3. SYSTEM ARCHITECTURE

Several challenges involved in solving this problem, as well as several unique opportunities botnets provide that will leverage in our design. We then describe our overall architecture and system design

.

3.1 Design Challenges

While designing the low-level framework and implementing the technique we described above, we faced the following challenges –

1. *Selection of Network Monitoring Tool*: To collect all the data we require to analyse and detect botnets, we used an appropriate Network Monitoring Tool. There exist many of them like pcap, winpcap, wireshark etc.

2. *Attributes used for clustering*: To ensure that we get relevant and strong correlations between the dataset, we need to well define the attributes based on which we classified the data. We have developed strong heuristics to effectively pin-point the group of 'malicious' internet traffic.

3. *Clustering Algorithm*: As already mentioned, there exist many clustering algorithms for grouping the data. Among these, we picked up the algorithm (coupled with a strong set of classification characteristics) which works best to group out the 'high-level-threat' traffic. To choose an effective algorithm, we compared and test a couple of algorithms and choose the one which gives the best results.

4. *False Positives*: If we do not use good heuristics and clustering algorithms, then it would had been resulted in false positives in favour of Bot-infection. Hence, we face the challenge of effectively choosing the above to ensure a high recall value and accuracy.

5. *Graphical User Interface*: We planned to implement a GUI which enables the user to easily monitor the network flow and block a bot from propagating.

3.2 Subsystem Decomposition

To detect Botnet, we started with monitoring the Internet traffic flow. Thus, analysed and clustered the data to compare it with the neighbouring nodes and detect bot-infection. The steps followed were as -

1. Network Traffic Monitoring
2. Clustering of Network Traffic Data
3. Comparison of clustered data with neighbouring nodes

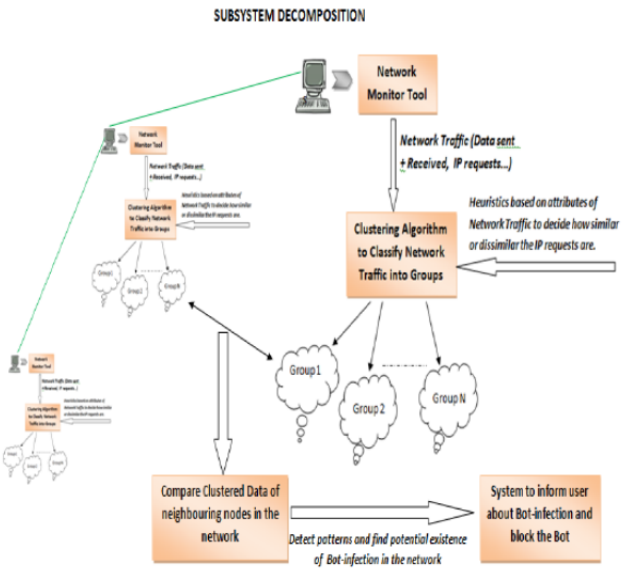


Figure 1 : Flowchart of the System

4. METHODOLOGY

As we are using Weka tool for clustering purpose the various steps involved in doing so are as follows:

4.1 Collect the sample data for network analysis

We are using Wireshark [10] as network analysis tool for capturing the packets. DNS data is captured on port 53 using wireshark. Wireshark produces the captured data file in .cap format, but since it is impossible to use .cap file for clustering purpose we need to convert it in a defined format which can be used as a input for clustering in our Weka tool (Clustering tool).

4.2 Conversion of .cap files to Specific format for Clustering analysis:

By so much search we are getting this fact that the most suitable format for clustering purpose is .csv format which is like a database format and easily read by Microsoft excel and other clustering tools. So for this conversion we are developing a special GUI tool in C#.Net which we named as Bot-detector tool[11] .The Bot-detector has one input file option in which we input only .cap file and after providing the input by use of another tool which is actually a command prompt tool named as Logparser[12] we give the following command-

logparser -i:NETMON -o:CSV "select Frame, DateTime , FrameBytes ,SrcMAC , SrcIP , SrcPort , ,DstMAC, DstIP, DstPort,IPVersion,TTL,TCPFlags,Seq,Ack,WindowSize,PayloadBytes,Connection INTO NetMonOutput.csv from sample1.cap".

4.3 Heuristics for clustering-

1. Session time between two IP addresses

In session time we consider the time for which two machines communicate with each other. This is usually measured in milliseconds. Clustering can be done on the basis of session time. Source and destination IP addresses communicating for same interval of time can be combined to form a cluster. Botnets have the property that the compromised machines communicate with each other for same interval of time. Thus IP addresses of compromised machines would get clustered and detected. Session time can be obtained by capturing packets on a machine using wireshark.

2. Domain name and corresponding IP addresses

Botnets have a unique property of fast flux. This is basically an attack technique that involves rapidly changing the bindings of IP addresses to domain names. It is typically employed to prevent detection of hosts operating illegal or unauthorized services. The detection of Fast Flux domains reveals indicates **malicious intent**. We are using this to detect suspected machines. If domain name corresponds to a large set of different IP addresses, this means machine is bot infected.

We are using 2nd heuristic to detect botnet.

4.4 Extracting domain names and ip addresses from domain name data

1. Initially, .cap file of DNS data is converted into .csv format.
2. A program is used to extract domain names and IP addresses from info column of Wireshark captured data file.
3. Next step is to convert IP address into a number.
4. Exact domain name is extracted from complete domain name by a program.
5. Domain name is converted into number by converting string into char and then assigning ASCII values and summing up to form a number.

4.5 Clustering of network traffic data

In previous step, we got network traffic data. Next step is to do clustering. Different types of clustering are available. Thus first step is to decide on what type of clustering to be used. Different steps as per plan document are discussed as.

4.5.1 Literature reading on clustering algorithms -

Our goal is to develop a botnet detection system. Clustering algorithms have been widely applied for this problem. Basically, clustering algorithms are partitioned into two categories-

1. Hierarchical clustering
2. Partitioning clustering

Hierarchical clustering method, as name clearly suggests, produces a set of clusters, in which pair of objects or clusters are progressively nested in larger cluster until one remains. In non-hierarchical method, n clusters are made for m data points, these clusters may or may not overlap. In portioning clustering these clusters do not overlap. We are employing K Means clustering for botnet detection. K-means is one of the simplest unsupervised learning algorithms that solve the well-known clustering problem. Input to K Means clustering is data file and number of clusters K to be formed. The main idea behind K Means clustering is to define k centroids, one for each cluster. It then Assign each object to the group that has the closest centroid. When all objects have been assigned, the positions of the K centroids are recalculated. Above steps

are repeated until the centroids no longer move. It basically calculates the distance (or difference) between two objects and perform clustering on basis of these distances.

4.5.2 Finding a tool

Number of tools are available to do clustering, out of which, we used WEKA tool, one reason being its popularity and other being usability.

WEKA tool

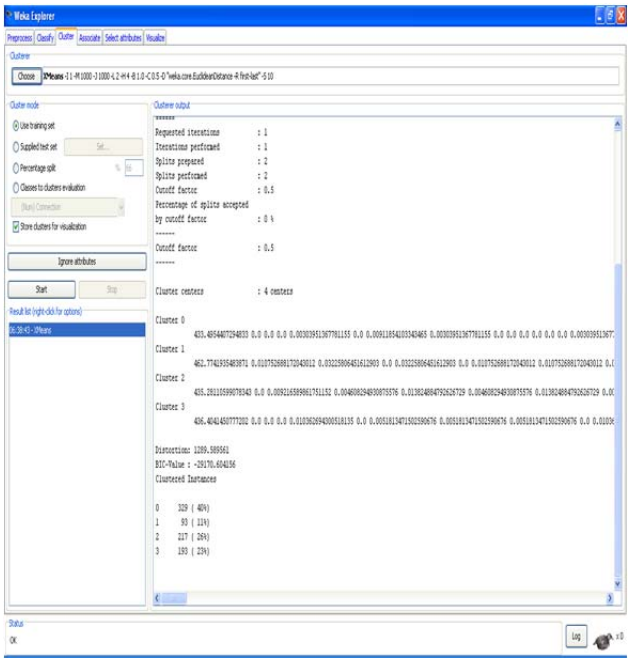
Weka is an abbreviation for Waikato Environment for Knowledge Analysis. It is a machine learning software written in java and developed at University of Waikato, New Zealand. It contains a collection of machine learning algorithms that can be used for real world problems. It contains tools for

1. Pre-processing
2. Classification
3. Clustering
4. Regression
5. Association
6. Visualization

These algorithms can be applied directly to a dataset or called from own java code. Dataset is imported and clustered using simple GUI called WEKA GUI CHOSSER, using Explorer or Simple CLI. Snapshot of data after loading is shown below, we can also perform both supervised and unsupervised filtering on data, which can be easily customized

4.5.3 Snapshot of clustering:

Snapshot of data after clustering is shown below. Weka supports 11 different types of clustering. We can customize clustering by clicking on type of clustering specified as if we can change number of iterations, max-clusters, seed etc. we are doing clustering on the basis of domain names.

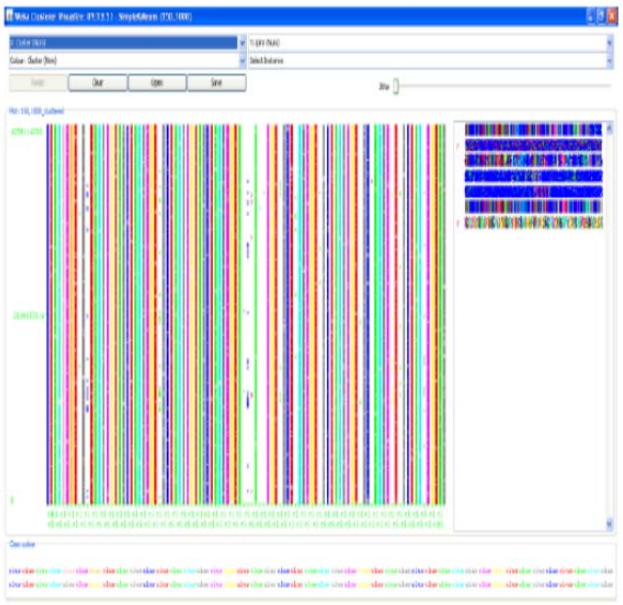


4.5.4 Insert infected data

We inserted infected data which resembles the botnet DNS data. Actually botnets are using fast flux for their DNS queries by which they can hide their malicious activities. So for the botnet like data we manually include different IP addresses for a particular DNS name. We implemented the analysis of results then on the basis of no. of DNS instances and their corresponding IP addresses per DNS names.

4.5.5 Detecting botnet by fast flux

We are detecting botnet by clustering our infected data on the basis of domain names. We are visualizing IP addresses of clusters of domain names, to identify botnet. If a domain name cluster covers a whole range of IP addresses available, then as per fast flux technique, it is suspected.



In this snapshot, vertical lines represented bot infected traffic.

5. RESULT

This section show whole analysis of Botnet Detection system. Different tables are constructed for different number of clusters and table contains number of DNS instances and their corresponding IP addresses. Each table shows percentage of botnet traffic detected, false positives and false negatives in the system.

for k = 50 cluster

S.NO	DNS INSTANCES	IP INSTANCES PER DNS	DETECTION RATE (%)	FALSE POSITIVE RATE (%)	FALSE NEGATIVE RATE (%)
1.	10	100	20	2	80
2.	10	500	90	2	10
3.	10	1000	90	2	10
4.	50	100	20	2	80
5.	50	500	59	2	41
6.	50	1000	66	2	34
7.	100	100	26	2	74
8.	100	500	37	2	63
9.	100	1000	42	2	58
10.	150	100	17.33	2	82.67
11.	150	500	27.33	2	72.67
12.	150	1000	30	2	70

For k =200 clusters

S.NO	DNS INSTANCES	IP INSTANCES PER DNS	DETECTION RATE (%)	FALSE POSITIVE RATE (%)	FALSE NEGATIVE RATE (%)
1.	10	100	90	0.5	10
2.	10	500	100	0	0
3.	10	1000	100	0.5	0
4.	50	100	82	0.5	18
5.	50	500	100	0.5	0
6.	50	1000	100	0	0
7.	100	100	89	0.5	11
8.	100	500	100	0.5	0
9.	100	1000	99	0.5	1
10.	150	100	69.33	0.5	31.67
11.	150	500	88	0.5	12
12.	150	1000	98.66	0.5	1.37

6. Conclusion

Botnets are a serious threat to network security. The aim of this paper is to detect botnets effectively at an early stage and deter malicious transmissions. Thus, to make a network of computers secure by stemming the bot-infection and hence preventing the spread of other fraudulent activities like spamming, phishing and DoS attacks. When we share data among nodes to compare the clustered data, we actually compromise upon the privacy of the users by sneaking into the network traffic flow of that node.

7. Future Work

In future, we want to extend our technique to work for mobile phones as well. With Wi-Fi and GPRS services now available on mobile phones, they have become prone to spam attacks and phishing. Botnet attacks on cell phones have not been spotted yet, but due to high similarity between internet and cellular networks, there exists a potential threat of such attacks in near future. To protect the privacy of users, i.e., to protect the traffic flow information of each node when this data will be shared to make comparisons of the clustered data, we can use homomorphic encryption on the clustered data.

Acknowledgment

I wish to acknowledge my mentor Ms. Vartika and all the contributors who contributed in developing and maintaining this work.

References

[1] J. Jones, “Botnets: Detection and mitigation,” Feb 2003. FEDCIRC.
[2] E. Cooke, F. Jahanian, and D. McPherson, “The zombie roundup: Understanding, detecting, and disturbing botnets,” in *The 1st Workshop on Steps to Reducing Unwanted Traffic on the Internet (SRUTI ’05)*, July 2005.
[3] P. Barford and V. Yegneswaran, “An inside look at botnets,” 2006. Special Workshop on Malware Detection, Advances in Information Security, Springer Verlag.
[4] M. A. Rajab, J. Zarfoss, F. Monrose, and A. Terzis, “A multifaceted approach to understanding the botnet phenomenon,” in *Internet Measurements Conference (IMC ’06)*, Oct 2006.
[5] D. Dagon, C. Zou, and W. Lee, “Modeling botnet propagation using time zones,” in *NDSS 2006*, Feb 2006.
[6] D. Dagon, “Botnet detection and response,” in *OARC Workshop, 2005*, 2005.
[7] J. Kristoff, “Botnets,” Oct 2004. 32nd Meeting of the North American Network Operators Group.
[8] J. R. Binkley and S. Singh, “An algorithm for anomaly-based botnet detection,” in *The 2nd Workshop on Steps to Reducing Unwanted Traffic on the Internet (SRUTI ’06)*, 2006.
[9] A. Ramachandran, N. Feamster, and D. Dagon, “Revealing botnet membership using dnsbl counter-intelligence,” in *The 2nd Workshop on Steps to Reducing Unwanted Traffic on the Internet (SRUTI ’06)*, 2006.
[10] **1** www.**wireshark**.org *contains all relevant information about wireshark.*
[11] Bot-Detector is our own GUI tool developed in C#.net used for converting .Cap format into .csv format for performing filterization and clustering purpose.
[12] LogParser:*technet.microsoft.com/enus/scriptcenter/dd919274.aspx is the BackEnd tool for our BotDetector tool and its main purpose is change it into .csv format*