Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 132 (2018) 1668–1677

Procedia Computer Science

www.elsevier.com/locate/procedia

International Conference on Computational Intelligence and Data Science (ICCIDS 2018)

Botnet Detection via mining of network traffic flow

Lakshya Mathur[b], Mayank Raheja[b], Prachi Ahlawat[a]

[a]The NorthCap University, Gurugram 1220017, India

[b] School of Engg & Technology, The NorthCap University, Gurugram 1220017, India

Abstract

During the past decade, botnet has emerged as a very serious threat to cyber security by proving it's capability of compromising billions of computers and making them does the illegal work.There are a number of existing ways by which botnet can be detected.A comprehensive overview of the existing techniques is also stated in this paper. Due to the involvement of huge amount of data, detection of botnet using machine learning algorithms is in huge trend. In this paper, we have used machine learning to train classifiers by a specific network flow dataset. Thereafter, the trained classifiers were applied on the collected data in order to evaluate the results. Analysis of network flow data is usedas a method of detection because it doesn't depend upon the packet content hence giving immunity towards the latest form of encryption and obfuscation used by attackers in order to hide their bots. Results are clearly showing that the proposed method is capable of differentiating the normal traffic and the bot traffic with a high accuracy and low false positive rate. In addition to this, almost every type of botnet can be detected using the proposed model.

Peer-review under responsibility of the scientific committee of the International Conference on Computational Intelligence and Data Science (ICCIDS 2018).

*Keywords:*botnet detection;zombie;machine learning;network flow;classifier;logistic regression;diversity;

1. Introduction

A botnet is a collection of compromised interconnected devices spread geographically over a wide variety of networks and devices. Botnet basically consists of a botmaster (Cybercriminal)to control and give commands to the

Peer-review under responsibility of the scientific committee of the International Conference on Computational Intelligence and Data Science (ICCIDS 2018).
10.1016/j.procs.2018.05.137

bots. Bots are the compromised computers which are connected to each other by any network protocol channel. Cybercriminals use botnet to perform illegal activities such as DDoS attack (Distributed denial of service attack), identity theft, stealing data, spamming etc [1]. Botnet ismalicious program (commonly trojan) that breaches the security of the devices and transfers the controls from user to botmaster through a network channel established under different network protocols like HTTP, IRC etc. Initially, botnets were designed to establish IRC i.e. Internet Relay Chat but after some time criminals compromised the security of the IRC network and used botnets to do illegal activities. Also, IRC botnet werevery common in the past but,in recent times botnets are designed in a way that all communication is done via P2P network of devices which makes it harder to pinpoint the source of the botnet.

Botnet network can be C&C(Command and Control) and P2P(Peer to Peer) as well. The structure can be both centralized and decentralized. Before 2000, Command and Control botnets were dominating the cybercrime world and moreover there was very little awareness about the same. In this kind of botnet, a command and control client server is established and all the bots are connected to that server. The server remotely controls the bots for giving commands and queries and those commands are executed and replied by the zombies (bots)[2]. At that time this was difficult to detect as the commands mix them in the normal HTTP traffic and hence it was difficult to differentiate between normal HTTP and bot HTTP traffic. Around 2000, P2P (peer to peer) botnet came into picture. It is basically a decentralized structure in which every bot behaves as a client as well as a server. There is a list of neighbors with every bot and the bot(behaving as a server) circulates command to every bot in its neighbor list. Botmaster controls P2P botnet by digital signing. Most of the systems using firewall or proxy does not accept an incoming connection to be made but readily accepts the outgoing connection. Those bots who are not capable of accepting becomes the server [3]. It is really difficult to take down this botnet as it proposes very high resiliency, even if a portion of P2P botnet is taken down, the rest part works same and before and hence it is important to take down every bot in this structure [4].

According to Justin Leonard, 2009[5], the life cycle of botnet has four different stages: formation, command and control, attack and post attack stage. In the formation stage, the cybercriminal tries to become a bot master and finds the systems with unpatched vulnerabilities and acquires them to create a bot structure. After that, second stage begins where the botmaster may send commands to control bots. In attack stage, the botmaster performs the attack and his desire of doing illegal work is accomplished. The post attack phase includes performing maintenance on the networksuch as replacing detected bots with new systems and ensuring no digital footprints were left behind. In thisphase, learning from the past attack, the functionality of bot structure is also improved.

Although researchers have proposed a number of botnet detection models in recent past but most of them were not able to detect the recent generation of botnets with both spatial (accuracy, memory) and temporal efficiency.

In our detection model, in order to minimize the destructive impact of the botnet,the proposed model aims to detect botnet in its first or second stage by examining the packet header instead of the packet contents to save both time and resources. The model aims to examine the network flow data in its TCP/UDP leveland uses both existing and self-generated network flow data. Further, through feature selection, the most relevant subset of features is extracted for analysis to improve accuracy and minimize time and resources. Machine learning classification techniques are then used to classify the data and get the results in a dichotomous form (botnet traffic or normal traffic). As we are only examining the packet header and not the data inside the packet hence encryption/obfuscation proves no difficulty to the model.

2. Related Work

In the current tech oriented world, as the threat of botnets is rising, there has been a great emphasis on research for botnet detection techniques over the years. There have been many types of approaches to detect botnets in both a single host and a whole network. Some approaches focus on mining system log files to detect anomalous behavior in system.

Some techniques focus of analyzing content of the packets to check if the packet is malicious or benign. These techniques in which we analyze payload are usually very resource intensive as large amount of resources and time are required to parse packet content for every packet. This technique also falls short when the packet is encrypted thus rendering the contents useless without decryption. One such technique was adopted in 2010 in a paper by Braun et al [6]., where their system chooses a certain number of packets at the onset of a TCP stream and analyzes their contents to see if they match a botnet signature. While this technique has its merits of streamlining another detection system's process, it still requires a large amount of memory to parse the packets and also would fail to work if the packet content is encrypted which is quite common these days.

In 2011, Saad et al. [7] explored a technique to detect botnets through network behavior analysis and machine learning in which they attempted to detect botnets in their command and control phase (C&C) before they launch their attack. Their approach specialized to detect P2P bots. They parsed and analyzed information about payload size, number of packets, duplicate packets length and concurrent active ports to build a feature set. Authors were able to achieve accuracy of above 90% and false positive rate less than 7% using artificial neural network and nearest neighbor classifiers. A drawback in their approach lies in the fact that the network traffic stream will change over time varying from phase to phase which will create a concept-drift problem. This creates problems during the training phase as detection model will not be able to detect the most recent types of botnet.

In 2008, a model was proposed (Gu et al.) [8], which was anomaly based detection technique. It was capable of identifying C&C servers with a great accuracy. Authors deduced the temporal and spatial link of same botnets such as synchronized communication, DDOS attacks, etc. It was a good attempt but the major drawback is that it worked well only in the case of centralized botnet. As it relied on spatial and temporal ties, it didn't account for the fact that these relations change from one centralized botnet to another which could prove to be a major setback.

In 2012, D Zhao et al. [9] proposed P2P detection system where the model aimed to detect botnets in the C&C phase and attack phase. They chose a small interval of time to analyze packets from. Authors used REP Tree classifier and bayesian network classifier to analyze and classify data up to 300s. The model was able to achieve an overall accuracy of over 90% while keeping the false positive rate below 5%. There system was vulnerable to the fact that if the botnet changes its behavior erratically, the model will not be able to account for the random change. It also required previous data for it to be trained. Our model addresses the above problems by not limiting the time frame and using different set of parameters to judge the flow other than temporal correlation.

In 2015, by Udaya et al. [10] there was a quite different approach to detect botnets where their model uses a generic template of Ip flows to detect a particular range of botnets. IPFIX enables them to use over 50 attributes to build their template. Now the advantage to using a generic template is the low overhead to even amid range or low range router which can be financially feasible. But the inability to detect any other botnet other than the programmed template makes it difficult to use for practical and ever growing needs.

In 2006, Livadas et al. [11] took an interesting approach to divide the model into two stages. Firstly to differentiate between IRC and non-IRC traffic, second stage to classify malicious and benign data among the results of first stage. They achieved false negative rates of 2.49% and false positive rates of less than 15%. This system reduced the detection time considerably by eliminating non-IRC traffic in the initial stage; leaving limited data to be analyzed further this saving resources. A drawback of their system was that they were specific to IRC botnets only and their false positive rates were quite high. Our model is able to achieve better false positive rates while maintaining the ability to detect a wider range of recent botnets.

In 2016 Wei et al. [12] proposed an unsupervised machine learning approach using clustering instead of classification like most methods. This approach was similar to ours in the way that its scope was not limited to a type of botnet and it was adaptable to new botnets. They worked on similarity analysis between malicious and benign data. But their result gave only a heuristic result for only a particular host. We decided to overcome this by using classification and to detect a botnet in an entire network and future plans to pinpoint the host as well.

There was a very interesting model in 2013 by Guntuku et al. [13], where their model was one which worked in real-time to detect P2P botnets in a system. They were able to achieve accuracy of greater than 99% for trained data and almost 99% for novel botnet traffic data. This worked in real time using a Bayesian regularized neural network which could effectively accommodate for new botnets that the system was not trained for. The only setback was the massive network bandwidth it had to parse and analyze along with the system and network resources it consumes. A

more effective way to overcome this hurdle would be distributed computing using Map-Reduce Framework in Hadoop or any other distributed environment.

In 2015, Dave et al. [14] proposed a botnet detection framework but for only P2P botnets. As P2P botnets are decentralised and distributed, so it becomes very difficult to take down such kind of network structure. Author presented a two-tier detection framework for detection of botnet in waiting stage itself with 99% accuracy by monitoring the behaviour of the network on two terms i.e. long-living peers and search request intensity. The disadvantage of this framework is that it can only detect P2P botnets and none else.

In 2017, Bennison et al. [15] proposed a detection framework for P2P botnets using graph analysis techniques or machine learning techniques on communication graph. This research was not much feasible as the ISP contains the universal communication graph which in reality is not very likely to be collected. Using partial information, the author analysed the detection algorithms and gave the result.

In the same year 2017, Chen et al. [16] proposed a detection method for botnets in high speed network environment. In this PF_RING was used to solve the problem to high packet drop rate and for the extraction of required fields from the traffic data. Random forest algorithm was used by the author on the CTU dataset. They obtained high accuracy but the unimpressive part of this paper is the use of only offline public dataset and no other online or self-generated data.

The prevailing techniques lack certain qualities such as adaptability, novelty detection and non-specific approach. Our method addresses each of these issues and also leaves scope for additional improvement for any future need.

Adaptability in any technique is a concern because an attacker will never use the same type of botnet to attack consecutively. Every other botnet we encounter will be different in the aspect of technology and complexity. A detection system needs to be adaptable to such changes automatically without any requirement for change in its core modules. This is achieved by using machine learning techniques mentioned in our experiments. As we choose to segregate any outlying data other than normal network traffic, it hardly leaves any scope for lack in adaptability as it will flag any data it deems not in the normal spectrum.

A number of existing techniques target a specific kind of botnet (Telnet, IRC, P2P, Domains, etc.) which limits its scope to that type only. Hence it will not be able to detect any other type of botnet which limits its usage below the line of practical usage. We address this problem as not differentiating between botnet types mainly but focusing on differentiating normal network traffic from the botnet traffic regardless of its type. In the beginning it was a challenge as having such a broad target leads to higher false positive rate. But consecutive training rectified the problem.

We also incorporate novelty detection till a certain extent in our approach as well; as a tool when compiled will already be trained with data that it when tested on another system, it will not need to train itself again to work, unless there is a huge change in botnet behavior over a large period of time.

3. Proposed Model

The model we propose offers qualities such as diversity of data, generalized detection of any kind of botnet, results in timely fashion and an acceptable level of accuracy fit for any purpose.This section will portray how each step of the experiment works and how it has been chosen to reflect the aforementioned qualities.

3.1. Data Collection

To ensure that our system will be able to deal with any type of botnet, we need to train it with network flow data from a diverse variety of sources which includes different types of botnets tested in different types of environment. In that pursuit, a virtual environment was set up with the help of *VMWare* software running operating systems such as Windows and Linux. In order to collect network data on linux machines, we set up Nfdump on the system. *Nfdump* with the help of its daemon *nfcapd* captures netflow data and stores into files. One nfcapd instance associated itself with one network flow and records data over time.

The advantage of using nfcapd in a linux environment is that it records a large number of netflow features which prove to be quite advantageous in further analysis. It also runs quietly in the background using minimal processing power and memory, hence an optimal choice as a tool to collect data. These features are further explained in

subsequent section.
In windows environment, *Wireshark* was used to capture network traffic data as the software is freely available. Other sources were also used to obtain data:

- CTU-13 Dataset[17]:
  - This is a labelled dataset with normal, botnet and background traffic.
  - It consists of 13 scenarios or 14 different captures of different botnet samples.
  - Captured files were in pcap format.
- ISOT Dataset[18]
  - This dataset combines both malicious and non-malicious data in pcap format.

The obtained data was converted to csv (comma separated values) format because csv format is recognized and compatible with most softwares and tools. It's a simple format to be parsed in a script as each a simple comma separates two records

All attributes captured in the netflow files are not always recorded in every scenario in every operating systems. So we eliminated a few of the attributes to maintain validity and non-redundancy in the data. The attributes were eliminated because all attributes are not applicable to every environment and monitoring tool, hence they get a null value assigned. So they are not of any importance to an experiment with the null values.
Converting the data from pcap/pcapng format to csv sometimes leads to a few errors, omissions, garbage values in the starting or end of the file. But these errors can manually be fixed by editing the files in any csv supporting software such as MS Excel or notepad.

The data recorded using virtual machines and the data downloaded from external sources was too large to be processed using normal desktop or laptop machines. Hence to overcome that hurdle, a small but sizable chunk of data was randomly selected which could be processed with the available machines in a practical amount of time. The selection of data was entirely random to ensure that the results of the analysis remain unbiased by any editing or selective choosing.

3.2. Machine Learning Algorithms

We explored different machine learning classification techniques, which could be used to distinguish botnet and benign traffic efficiently. We have selected the most effective classification techniques namely.

- Randomized Filtered Classifier:
  - A decision of a committee is always considered to be better than that of an individual. This ideology is incorporated in this classifier. A simple variant of the FilteredClassifier that implements the Randomizable interface, useful for building ensemble classifiers using the RandomCommittee Meta learner. It requires that either the filter or the base learner implement the Randomizable interface.
- Logistic regression[19]:
  - Logistic Regression predicts the outcome of a situation in the form of a dichotomous variable i.e. it can have two values, 0 or 1.
  - It tries to estimate the probability of a binary outcome according to a few independent variables also known as predictor variables.
- Random Committee:
  - A number of base classifiers are built using different seed values selected at random. The final prediction is a simple average of all the predictions made by the base classifiers.
  - This improves the result as it combines the result of classifiers used with various seed values and reduces the possibility for error.
- Random Subspace:
  - This method constructs a decision tree based classifier that maintains highest accuracy on training data and improves on generalization accuracy as it grows in complexity. The classifier consists of multiple trees constructed systematically by pseudorandomly selecting subsets of components of the feature vector, that is, trees constructed in randomly chosen subspaces.

- Multi Class Classifier:
    - This classifier classifies records into 3 or more classes. A metaclassifier for handling multi-class datasets with 2-class classifiers. This classifier is also capable of applying error correcting output codes for increased accuracy.

3.3. Feature Extraction

There were a number of features recorded using nfdump:

The following are the list and description of the attributes that were used are:

Ts (Flow start time), Te (Flow end time), Td (Flow duration), Sa (Source IP address), Da (Destination IP address), Sp (Source port), Dp (Destination port), Pr (Protocol), Ra/flg (Flg Flags), Ipkt (Input Packets), ibyt (Input Bytes).

Feature selection is an important part of the experiment. It has been proven that not all attributes are equally contributing to the result. Some are more important and relevant to the learning and analysis than others. Some features just cause redundancy causing the accuracy to fall. Feature Selection or attribute selection is an operation by which itself-actively searches for the best subset of attributes in a dataset. The notion of "best" is specific to the problem one is trying to solve, but typically means highest accuracy and low false positive rate.

There are two types of feature selection techniques: Wrapper and Filter.

Wrapper technique looks for an optimal subset of features fitting to a particular algorithm; it also detects interactions between variables unlike the filter approach. But this technique finds a subset limited to the domain and the algorithm.

In our model, we use the filter technique, which is a rather straightforward approach which measures which subset of the features has the highest predictive power. It's not dependent on any algorithm and can be used in any domain, which is the main reason for choosing this method.

We use CfsSubsetEval as a filter method to choose the most relevant features to the experiment.

4. Results and Discussion

To test the aforementioned machine learning algorithms, a series of experiments were conducted using *WEKA*, written in Java, is a suite of machine learning algorithms accessible by either a Java code or a GUI.

The dataset for the experiments was created as follows; a random portion of data from all sources was combined together. To maintain authenticity and unbiased results even when working on a smaller dataset this selection was made on a purely random basis.

A small dataset was required for analysis because a large dataset would require amounts of processing power unavailable in a personal computing system.

Combining different datasets also provides diversity of data as we compile traffic flow of different types of botnets from different type of environments.

We used a number of factors to judge the efficiency of our approach and determine the results of the experiment. We judged an algorithm on the basis of *Accuracy*, which is the percentage of correct results predicted by the classifier, *False Positive Rate*, which denoted the percentage of results the classifier flagged as botnet traffic but which were actually normal traffic and *Time Taken*, which is a measure of time to build and evaluate the training set for an algorithm.

Accuracy can be calculated as: $ACC = \frac{TP+TN}{\mathrm{TP + TN + FP + FN}}$, (1)

Where **TP**(True Positive) means number of botnet packets labelled as botnet,
FP(False Positive) meaning number of normal packets labelled as botnet,
TN(True Negative) meaning number of normal packets labelled as normal traffic and,
FN(False Negative) meaning number of botnet packets labelled as normal traffic.

False Positive Rate can be calculated as $FPR = \frac{FP}{FP+TN}$ (2)

And Time taken is represented in seconds (s).

In order for the algorithms to calculate the above result the data needed to be in nominal form instead of binary or numerical. Weka pre-processing stage was used to do this.
An *unsupervised attribute* filter called *numerictonominal* was used to convert the numeric attributes to nominal values.
Next, we used the *attribute selection* tab of WEKA to perform feature selection.
CfsSubsetEval attribute evaluator was used in conjunction with a best first search method which resulted in the following subset: **td**(Flow duration), **da**(destination address), **pr**(protocol).

We used 5 classifiers to build training sets and analyse our data to make predictions and got the following results:

Table 1: Performance of algorithms according to different metrics

ALGORITHM	ACCURACY	FPR	TIME TAKEN(s)
Logistic Regression	0.984	0.004	0.03
Random SubSpace	0.975	0.111	0.03
Randomizable Filtered Classifier	0.977	0.042	0.05
MultiClass Classifier	0.984	0.004	0.03
Random Committee	0.953	0.219	0.00001

Observing Table 1, we can clearly see that Logistic Regression and MultiClass Classifier were able to achieve the highest Accuracy of 0.984 and lowest false positive rate of 0.004 in 0.03s of time. However Random Committee's time taken to build its model is significantly lower than any other algorithm but it falls short in terms of accuracy and false positive rate and hence not a practical solution.
Random Subspace and Randomizable Filtered Classifier also performed well but they have higher false positive rateand slightly lower accuracy than others.

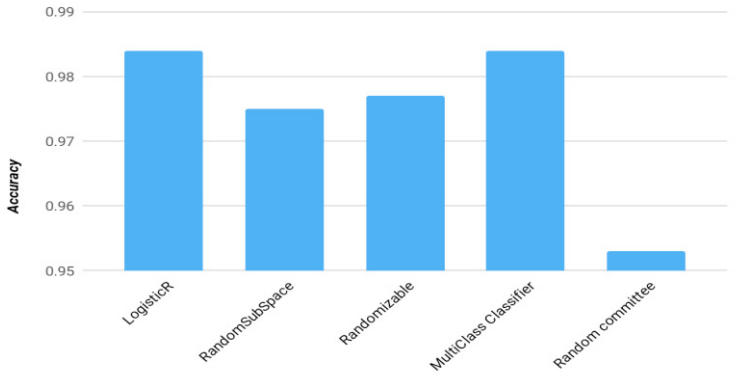

Figure 1 Accuracy of all algorithms

Figure 1 shows that out of all 5 algorithms Logistic regression and Multi Class Classifier were better than all others achieving an accuracy of 0.984. They were able to correctly predict 98.4% of the data.

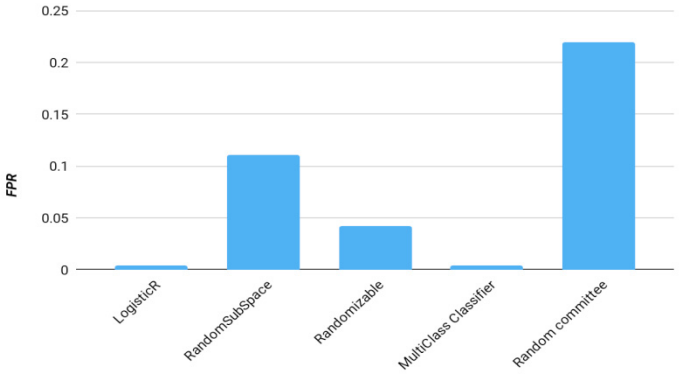

Figure 2.False Positive Rate of all algorithms

Figure 2 shows that once again Logistic Regression and MultiClass Classifier achieve the best false positive rates of 0.004 meaning only 0.4% of the data was falsely labelled as normal traffic which can be accepted in a practical environment.

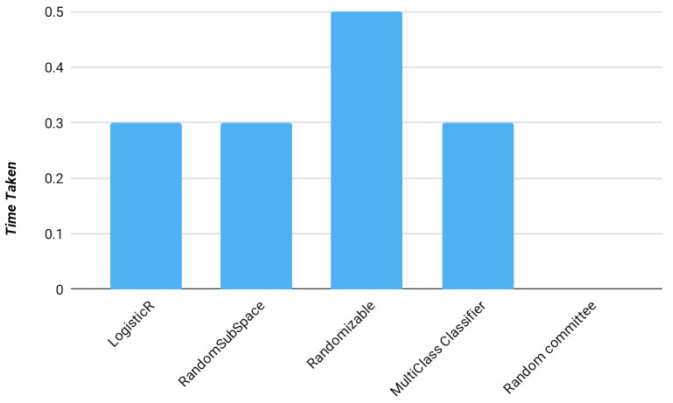

Figure 3.Time taken to build training set for all algorithms

Figure 3 depicts that Random Committee outperforms all other algorithms, in terms of time taken to build the training model, by a significant amount. It took less than 0.00001 seconds to build its training set which is a great trait for a practical solution.

Observing all factors such as accuracy, false positive rate and time taken to build a model we can see that while Random Committee taken record time to finish the task but it cannot compete with other algorithms in terms of accuracy and false positive rate and hence has to be disregarded.

Logistic Regression and Multi Class Classifier offer high accuracy of 0.984 as we can see from Table 1 with a false positive rate of 0.004 and practically doable time of 0.03 seconds.

Now Multi Class classifier is more useful when data needs to be classified into three or more classes while Logistic

Classifier statistically deals with independent variables which determine an outcome which is binary in nature i.e classifiable into two classes.
Now we are dealing with mutually independent variables td(duration), da(destination address) and pr(protocol) which determine a dichotomous variable botnet which can either be 0 or 1.
Hence Logistic Regression fits into our botnet detection scenario better than Multi Class Classifier.

5. Conclusion

This paper analyzed and experimented with five different classification techniques to find out two most suitable techniques for detection of botnet. Thereafter, out of two classification techniques i.e. Logistic Regression and Multi Class classifier,we recommend Logistic Regression for practical usage as it performs the tasks in favourable amount of time with high accuracy. While random Committee takes a fraction of the time taken by others, its false positive rate and accuracy are worse as compared to others by a significant amount.
MultiClass classifier also shows the same results as logistic regression but considering the fundamental working of both algorithms, Logistic Regression Classifier is better suited for botnet detection purposed as explained in the previous chapter.
One of the drawbacks of this approach is that normal network usage evolves exponentially in the current cyber landscape and after a certain point in time the method might not be able to differentiate malicious or benign data. A scope of improvement is present which will solve the high false positive rate in Randomized Filtered Classification algorithm and also incorporate novelty detection to higher levels. This can be achieved by applying a neural network with deep learning optimization to account for evolving botnets.
For our future work, we aim to add features to classify result as a particular type of botnet, incorporate a modified trace to pin point source (IP) of botmaster. There is also scope to identify, if a botnet exists, what kind of services it has access to, so it becomes easier to wipe it from the system. More recent plans are to automate the entire system for ease of understanding and access.

References

[1] Botnet. [Internet] 2018 [updated 2018 Mar 5] Available from: https://en.wikipedia.org/wiki/Botnet

[2] P. R. Marupally and V. Paruchuri(2010), Comparative Analysis and Evaluation of Botnet Command and Control Models, 24th IEEE International Conference on Advanced Information Networking and Applications, Perth, WA, 2010, pp. 82-89.

[3] Wang et al (2007), An Advanced Hybrid Peer-to-Peer Botnet, University of Central Florida, Orlando, FL.

[4] G. Kirubavathi∗, R. Anitha(2016),Botnet detection via mining of traffic flow characteristics by Department of Applied Mathematics and Computational Sciences, PSG College of Technology, Coimbatore, India.

[5] Justin Leonard, Shouhuai Xu, Ravi Sandhu(2009), "A Framework for Understanding Botnets", InProceedings, rd International Workshop on Advances in Information Security (WAIS at ARES), March 16, 2009.

[6] L. Braun, G. Münz and G. Carle (2010), "Packet sampling for worm and botnet detection in TCP connections," *2010 IEEE Network Operations and Management Symposium - NOMS 2010*, Osaka,

[7] S. Saad *et al*.(2011), "Detecting P2P botnets through network behavior analysis and machine learning," *2011 Ninth Annual International Conference on Privacy, Security and Trust*, Montreal, QC,

[8] Gu, G., Zhang, J., & Lee, W. (2008). BotSniffer: Detecting Botnet Command and Control Channels in Network Traffic. *Proceedings of the 15th Annual Network and Distributed System Security Symposium*.

[9] Zhao D., Traore I., Ghorbani A., Sayed B., Saad S., Lu W. (2012) Peer to Peer Botnet Detection Based on Flow Intervals. In: Gritzalis D., Furnell S., Theoharidou M. (eds) Information Security and Privacy Research. SEC 2012. IFIP Advances in Information and Communication Technology, vol 376. Springer, Berlin, Heidelberg

[10] Wijesinghe U, Tupakula U, Varadharajan V(2015). An enhanced model for network flow based botnet detection. In Parry D, editor, Proceedings of the 38th Australasian Computer Science Conference, ACSC 2015. Vol. 159. Sydney: Australian Computer Society.. p. 101-110. (Conferences in Research and Practice in Information Technology).

[11] Livadas, Carl, Robert Walsh, David Lapsley and W. Timothy Strayer(2006). Using Machine Learning Techniques to Identify Botnet Traffic.

[12] Wu, Wei & Alvarez, Jaime & Liu, Chengcheng & Sun, Hung-Min. (2016). Bot detection using unsupervised machine learning. Microsystem Technologies. . 10.1007/s00542-016-3237-0.

[13] Guntuku, Sharath Chandra & Narang, Pratik & Hota, Chittaranjan. (2013). Real-time Peer-to-Peer Botnet Detection Framework based on

Bayesian Regularized Neural Network.

[14] Priyanka, Mayank Dave (2015), "PeerFox: Detecting parasite P2P botnets in their waiting stage", *Signal Processing Computing and Control (ISPCC) 2015 International Conference on*, pp. 350-355, 2015.

[15] Harshvardhan P. Joshi, Matthew Bennison , Rudra Dutta (2017), "Collaborative botnet detection with partial communication graph information", *Sarnoff Symposium, 2017 IEEE 38th*.

[16] Ruidong Chen, Weina Niu, Xiaosong Zhang, Zhongliu Zhuo, and Fengmao Lv (2017), "An Effective Conversation-Based Botnet Detection Method", *Mathematical Problems in Engineering*, Volume 2017

[17] Malware Capture Facility project. Retrieved from https://mcfp.weebly.com/the-ctu-13-dataset-a-labeled-dataset-with-botnet-normal-and-background-traffic.html

[18] Datasets. Retrieved from https://www.uvic.ca/engineering/ece/isot/datasets/

[19] Class Logistic. Retrieved from http://weka.sourceforge.net/doc.dev/weka/classifiers/functions/Logistic.html