ISSN 1870-4069

Botnet Detection using Clustering Algorithms

Francisco Villegas Alejandre, Nareli Cruz Cortés, Eleazar Aguirre Anaya

Instituto Politécnico Nacional,
Centro de Investigación en Computación, Mexico City,
Mexico

b140572@sagitario.cic.ipn.mx,{nareli,eaguirre}@cic.ipn.mx

Abstract. In this paper, some clustering techniques are analyzed to compare their ability to detect botnet traffic by selecting features that distinguish connections belonging to or not belonging to a botnet. By considering the history of network's connections, some clustering algorithms are used to derive a set of rules to decide which should be considered as a botnet. Our main contribution is to evaluate different clustering techniques to detect botnets based on their detection rate (true and false positives). The algorithms used are *K-medoids* and *K-means* clustering. Datasets used in this paper were extracted from the repositories ISOT and ISCX. Results on *K-medoids* were better for almost all the experiments than *K-means*.

Keywords: Malware detection, botnet, clustering algorithms.

1 Introduction

Botnets are a group of infected computers (bots) remotely controlled by a botmaster through a channel Command and Control (C&C). Botnets can be classified as centralized and decentralized. In centralized botnets, bots contact the server C&C to receive instructions. Meanwhile, in decentralized botnets, only one of bot receives the message from C&C server, then this bot is responsible for transmitting the message to other bots, and those bots to more bots, and so on.

Leonard et al. divided the life cycle of the botnet in four phases: training, C&C, attack, and post-attack [6]. During the training phase, the botmaster infects other machines through the Internet, these infected machines now become bots controlled by the botmaster and receive instructions from the botmaster during phase C&C. During the attack, bots perform malicious activities based on the instructions received. During the phase post-attack, some bots could be detected and removed, for this reason, the botmaster analyzes the botnet (occasionally) to detect bots still active.

In this research, an approach based on clustering algorithms to detect botnets at the phase C&C using connections between devices, is presented. Network connections are used to identify the behavior of botnets. A connection network is the traffic between two specific endpoints. The connections are used to organize packages as 5-tuple in the following manner: <source IP address, destination IP address, source port, destination port, protocol>.

This document is organized as follows: in Section 2 some related algorithms to detect botnets are mentioned, in Section 3 the detection process used in this work to detect botnets is mentioned, in Section 4 the different types of experiments carried out are explained, in Section 5 the results of experimentation are mentioned, in Section 6 the conclusions of the paper are mentioned.

2 Related Works on Algorithms for Detecting Botnets

Some works related to the algorithms based on connections into time intervals to detect botnets at the phase of C&C, are following:

K. Huseynov et al. [1] performed a comparison between the algorithms *K-means* and *Ant Colony System* to detect decentralized botnets. They used features based on the host, to propose a method able to detect the botnets quickly and accurately. Their results show that the algorithm *K-means* has a better performance than *Ant Colony System*. The algorithm *K-means* obtained 82.1% of detection rate with 2.4% false positives, and the Ant Colony System scored a very low detection rate with 67.8% and high rate of false positives, about 23.5%. The dataset used was ISOT.

S. Garg et al. [2] performed a comparison of three machine learning techniques commonly used for the detection of decentralized botnets: *C4.5*, *Nearest Neighbours*, and *Bayesian Network*. They used features based on connection intervals of time. Additionally, feature selection was executed. They showed the effectiveness of some of the features on the dataset ISOT, the algorithms with the best results were *C4.5* and *Nearest Neighbors*.

Saad et al. [3] performed a comparison among five machine learning techniques for decentralized botnets detection. The results of the experiments based on the dataset ISOT showed that it is possible to detect botnets with high precision during the phase C&C. The classification techniques were: *Support Vector Machine (SVM)*, *Naive Bayes*, *Gaussian Classifier*, *Artificial Neural Network*, and *Nearest Neighbours*, where *SVM* obtained the best detection rate, about 98%.

D. Zhao et al. [4] used the *RepTree* algorithm to detect botnets using the dataset ISOT. The results showed a detection rate of 98.1% for a reduced dataset and 98.3 % for the full set with time windows of 300 seconds (8.58 seconds for training and 29.4 seconds of testing). They analyzed the detection rate and false positives of botnets with various time windows, where the best time window was 300 seconds. Furthermore, they built a server to detect botnets in real time and tested with two centralized botnets: black energy and weasel, obtaining 100% of detection rate.

P. Narang et al. [5], instead of the traditional 5-tuple flow-based detection approach, used a 2-tuple conversation-based approach which is port-oblivious, protocol-oblivious, and it does not require Deep Packet Inspection. They named their detector PeerShark; it also classified different P2P applications like Emule and Utorrent, furthermore, they also detect Storm and Waledac p2p traffic with a detection rate of more than 95%. They executed tests with 3 different algorithms

Bayesian Network, *C4.5*, and *Adaboost with Rep trees* to detect the decentralized botnets and the best algorithm was the C4.5. The dataset used was ISOT.

3 Proposal

Our proposal consists of the definition of some clusters that defines the limits for a botnet detection strategy. The detection process receives as input a feature vector and a training set. The clustering algorithm receives that training set and the feature vector to generate clusters that define the behavior of the training set, after that, the clusters are transformed into a set of if-then-else rules, that can distinguish between botnets and non-botnets in new datasets (testing sets). Figure 1 refers to the detection process carried out in this research.

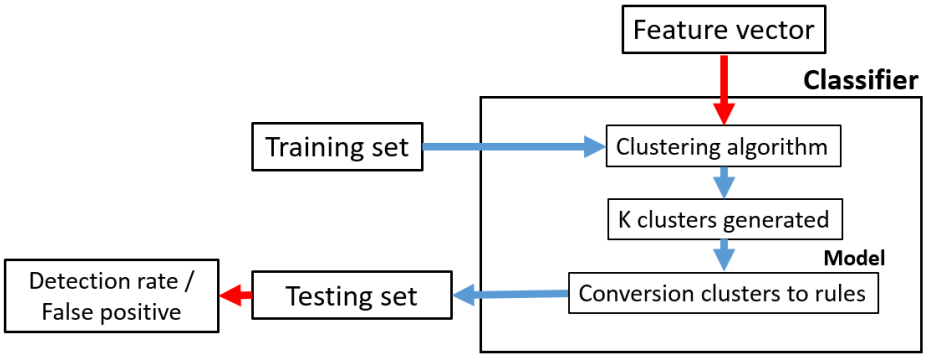


Fig. 1. The general idea of the proposed detection process.

3.1 Feature Vector

The initial 8-feature vector used in this work is shown in Table 1. The first column refers to the names given to the features and the second column is the description of the features.

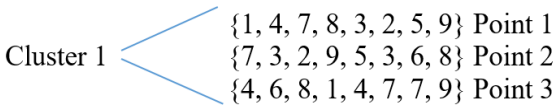
3.2 Conversion from Clusters to Rules

The clustering algorithm receives the training set and the feature vector to generate clusters that define the behavior of the training set. The training set contains data points representing botnet connections. Each data point contains 8 values representing the feature vector. The clustering algorithm forms groups of data points called clusters. These clusters are converted into rules represented as a vector, where each rule has 8 limits with the lowest value from every feature in all the data points and 8 limits with the highest value from every feature in all the data points.

Table 1. Feature vector using connections.

Name	Description
APL	Average payload length per connection
MDPS	Number of a different size of packets transferred to a total number of frames per connection
Payload	Total number of bytes per connection excluding the header.
TPT	Total bytes transmitted per connection
Duration	Duration of the connection
Flen	Large of the first packet of the connection
ToByte	Bytes from origin to destination
NPacketsAB	Number of packages from origin to destination

An example of the conversion from clusters to rules:



Rule 1 {1-7, 3-6, 2-8, 1-9, 3-5, 2-7, 5-7, 8-9} generated from cluster 1, where the cluster1 contains 3 data points with 8 features. To generate the rule1 from the cluster1, the lowest-highest value is taken from every feature in the data point.

3.3 Clasification

Once the rules have been generated, the classification is achieved by applying the rules forming a if-then-else model in a testing set containing botnet and non-botnet connections. The point classified as botnet or non-botnet depends on if it meets the conditions of at least a rule or not.

Being so, if the new point meets the conditions of a rule (where the new point should be between all the limits of the rule) is considered botnet, otherwise is considered a non-botnet.

The if-then-else model is composed as follows:
If any rule is accomplished
    A botnet is detected
Else
    A no botnet is detected
An example of this, where rule1 classify 3 points of a testing set:
Rule 1 {1-7, 3-6, 2-8, 1-9, 3-5, 2-7, 5-7, 8-9}
Testing set containing 3 points:
{2, 4, 7, 8, 3, 2, 5, 9} Point1. Considered as Botnet by rule 1

{0, 4, 7, 8, 3, 2, 5, 9} Point2. Considered as No Botnet by rule 1
{8, 7, 9, 9, 6, 1, 4, 9} Point3. Considered as No Botnet by rule 1

4 Experiments

We performed specific experiments, varying the number of clusters to obtain the number of clusters with highest detection rate and lowest false positives for every centralized and decentralized botnet. In our experiments, two datasets were used. hese datasets are:
· ISOT. Which contains decentralized botnets.
· ISCX. Which contains centralized botnets.

In the next subsections, first a description about the datasets used for the experiments are mentioned, then a description of the experiments carried out are mentioned.

4.1 Datasets

A description of the two datasets used for the experimentation are mentioned. These datasets contain representative data for centralized and decentralized botnets as well as normal traffic, focusing only on the connections of this traffic. From these datasets the training set, as well as the testing set are extracted. These two datasets are the following:

ISOT: This dataset was created by Information Security and Object Technology (ISOT) research laboratory at the University of Victoria [9]. Basically, it is a mixture of many existing datasets (malicious and non-malicious). Malicious traffic in the dataset ISOT were extracted from the French chapter of the Honeynet Project [10] and it includes three different decentralized botnets: Waledac, Storm, and Zeus. In total, the dataset of ISOT contains 14.1 GB of data in pcap format. The description of this dataset is shown in Table 2, in which the first column represents the type of botnet, the second represents the number of connections, and the third column represents the type of botnet.

Table 2. Description of dataset ISOT.

Botnet	Number of connections	Type
ISOT Storm	22,888	Decentralized
ISOT Waledac	34,442	Decentralized
ISOT NO Botnet	77,586	NO Botnet

ISCX: This dataset was created by Information Security Centre of Excellence ISCX (ISCX) research laboratory at the University of New Brunswick [7]. It

has been generated in a physical test environment using actual devices that generate traffic (SSH, HTTP, and SMTP). It contains the centralized botnets Neris and RBot. In total, the dataset ISCX contains 5.6GB of data in pcap format. The description of this dataset is shown in Table 3, in which the first column represents the type of botnet, the second represents the number of connections, and the third column represents the type of botnet.

Table 3. Description of dataset ISCX.

Botnet	Number of connections	Type
ISCX Neris	33,084	Centralized
ISCX RBot	34,217	Centralized
ISCX NO Botnet	76,175	NO Botnet

4.2 Design of Experiments

To validate our proposal we did four types of experiments, each experiment is repeated four times, varying the parameter number of clusters (rules) in each one (cluster values of 100, 200, 500, 1000). To get the number of the clusters with highest detection rate and lowest false positives in the phase of C&C for a specific botnet, the botnets evaluated were: the decentralized botnets Storm and Waledac are shown in Table 4a and 4b, and the centralized botnet Neris and RBot in Table 4c and 4d.

Furthermore, to perform a comparison of results with the related work, a general experiment called ISOT was performed. The main goal in this experiment is to obtain the detection rate for the decentralized botnets.

The ISOT experiment contains the decentralized botnet Storm and Waledac shown in Table 4e.

The first column of the tables corresponds to the type of botnet in the dataset; the second column to the class or label of data; and the third column the number of connections used for that botnet. The *K-medoids* algorithm was programmed in the Java language and the *K-means* is the one implemented in weka [8].

Table 4a. Experiment 1 to detect Storm.

Botnet	Class	Connections
Storm	Storm	22,888
NO Botnet	No Storm	22,888

Table 4b. Experiment 2 to detect Waledac.

Botnet	Class	Connections
Waledac	Waledac	34,442
NO Botnet	No Waledac	34,442

Table 4c. Experiment 3 to detect Neris.

Botnet	Class	Connections
Neris	Neris	33,084
NO Botnet	No Neris	33,084

Table 4d. Experiment 4 to detect RBot.

Botnet	Class	Connections
RBot	RBot	34,217
NO Botnet	No RBot	34,217

Table 4e. General experiment for ISOT.

Botnet	Class	Connections
Storm	Botnet	22,888
Waledac	Botnet	34,442
NO Botnet	No Botnet	77,586

5 Results

K-means and *K-medoids* algorithms were used for clustering. These two algorithms were chosen, influenced by state of art in clustering algorithms, due to their popularity. Some other clustering algorithms can be used. The state of art obtained better results.

Two values were used to evaluate the clustering algorithms: false positives, and detection rate.

The results of the specific experiments are illustrated in the Figures 2a, 3a, 4a, 5a for the detection rate obtained with that number of clusters (rules) for each botnet, and the figures 2b, 3b, 4b, 5b for the false positives according to the detection rate obtained with that number of rules for each botnets. The results of the general experiment ISOT are illustrated in the Figures 6a and 6b, for the detection rate and false positives respectively. These figures show detection rate and false positives for both clustering algorithms *K-means* and *K-medoids*, varying the number of rules parameter. Results are better when the detection rate is higher and the false positives are lower, an opposite correlation between detecion rate and false positives. To obtain the best correlation between detection rate and false positives, maintaining the detection rate as high as possible and the false positives as low as possible.

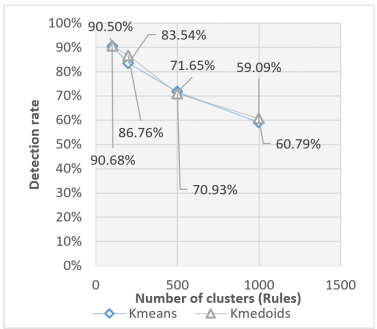


Fig. 2a. Detection rate for Storm.

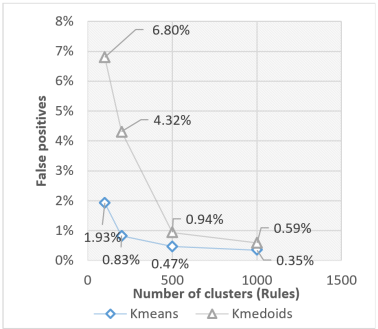


Fig. 2b. False positives for Storm.

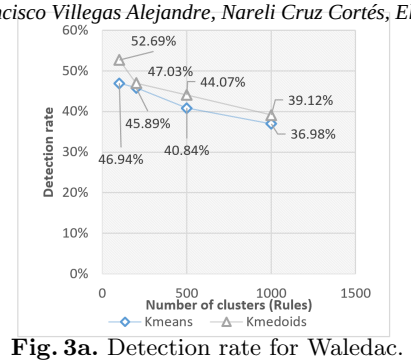


Fig. 3a. Detection rate for Waledac.

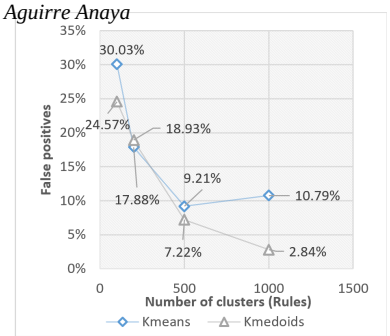


Fig. 3b. False positives for Waledac.

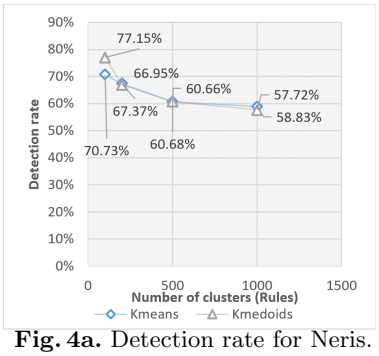


Fig. 4a. Detection rate for Neris.

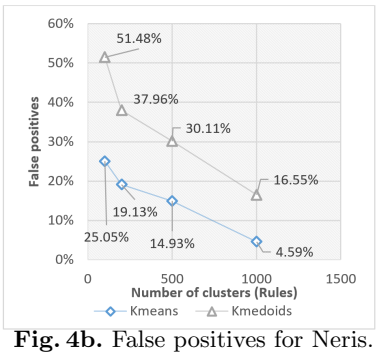


Fig. 4b. False positives for Neris.

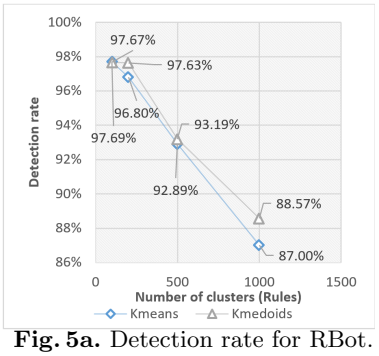


Fig. 5a. Detection rate for RBot.

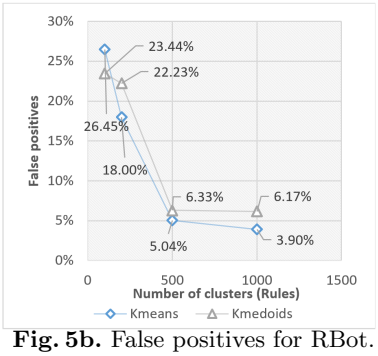


Fig. 5b. False positives for RBot.

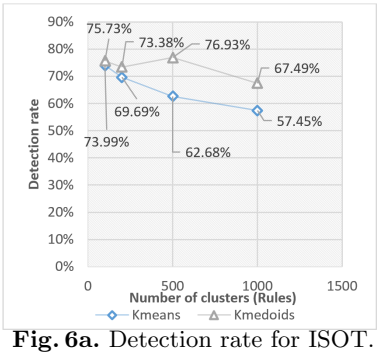


Fig. 6a. Detection rate for ISOT.

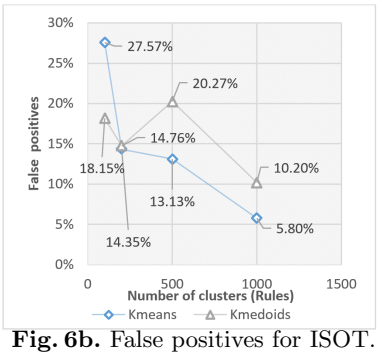


Fig. 6b. False positives for ISOT.

Comparison of Results: In Table 5, a comparison of results with the related work is shown. The table shows the reference to the related work in its first column; the second column the algorithms used; the third column is the dataset used; in the fourth and fifth column the detection rate and false positives.

Table 5. Comparison of results.

Reference	Algorithm	Dataset	Detection rate	False positives
K. Huseynov [1]	*K-Means*	ISOT	82.1%	2.4%
K. Huseynov [1]	*Ant Colony System*	ISOT	67.8%	23.5%
S. Saad [3]	*SVM*	ISOT	97.8%	5.1%
D. Zhao [4]	*RepTree*	ISOT	98.3%	0.01%
P. Narang [5]	*C4.5*	ISOT	98.7%	0.04%
This research	*K-means*	ISOT	69.99%	14.35%
	K-medoids	ISOT	73.37%	14.76%

The *K-means* algorithm and *K-medoids* show almost similar results. The number of clusters (rules) with high detection rate, but low false positives are 500 in average. The *K-means* algorithm got better results for the bot neris with a detection rate of 57.72% and false positives 4.59% with 1000 clusters, and for the bot storm with 90.68% detection rate and false positives of 1.93% with 100 clusters than the algorithm *K-medoids*. On the other hand, the algorithm *K-medoids* got better results for the bot rbot with a detection rate of 93.19% and false positives of 6.33% with 500 clusters. Furthermore, for the bot waledac with 44.07% and false positives 7.22% with 500 clusters than *K-means*, this is because they have a better correlation between detection rate and false positives.

The algorithm *K-medoids* obtained better results in the general experiment ISOT with 73.33% detection rate and 14.76% false positives. The algorithm *K-means* obtained 69.99% detection rate and 14.35% false positives. The clusters (rules) with better correlation in the general experiment were 200.

The comparative of results in Table 5 show that our proposal obtained worst results than K. Huseynov [1], S. Saad [3], D. Zhao [4], and P. Narang [5] with the algorithms *K-means*, *SVM*, *RepTree*, and *C4.5*. On the other hand, our proposal obtained better results than K. Huseynov [1] with the *Ant Colony System* algorithm.

Perhaps the comparisons are not so fortunate for the results of the proposed method and may be because they have used different basis of features, the next phase of experimentation would force to consider tests on the exact base of features that each method in the state of the art uses generating and confronting a pair of results to confirm or discard clustering as a strategy for the detection of botnets.

6 Conclusions

In this work, two clustering algorithms were evaluated for the formation of rules to detect botnets via network connections. In this study the detection rate and false positives of different types of botnets were evaluated, our proposed method is allowed to detect the botnets in the phase of C&C accurately for some botnets by generating rules.

Based on the results we can conclude that the *K-means* algorithm is better to detect Neris and Storm than *K-medoids*. The *K-medoids* algorithm is better for detecting Rbot, Waledac, and the general experiment with decentralized botnets (Storm and Waledac) than *K-means*. These algorithms detect the botnets accurately for Storm and RBot with detection rate over 90% and false positives under 7% with a very low number of clusters (rules) 500 in average. Our main objective was accomplished partially because the algorithm *K-medoids* has better results only for 2 botnets and for the general experiment. This means that the *K-medoids* can be used as an alternative to *K-means* algorithm only for these types of experiments. Therefore, evaluation of other clustering algorithms is needed to achieve this objective.

Our results can be improved with another technique of detection. Furthermore, the results can be improved with a method of feature selection, this is the purpose of our future work.

Acknowledgment. The authors would like to thank to the *Instituto Politécnico Nacional* (IPN), the *Centro de Investigación en Computación* (CIC), and the *Consejo Nacional de Ciencía y Tecnología* (CONACYT) for the support in this research.

References

1. Huseynov, K., Kim, K., Yoo, P.: Semi-supervised Botnet Detection Using Ant Colony System. In: 31th Symposium on Cryptography and Information Security, Kagoshima, Japan, Jan. 21-24 (2014)
2. Garg, S., Singh, A., Sarje, A., Peddoju, S.: Behaviour Analysis of Machine Learning Algorithms for detecting P2P Botnets. Department of Computer Science and Engineering, Indian Institute of Technology, Roorkee, India (2013)
3. Saad, S., Sayed, B., Felix, J., Traore, I., Zhao, D., Ghorbani, A., Ku, W., Hakimian, P.: Detecting P2P botnets through network behavior analysis and machine learning. In: Privacy, Security and Trust (PST), on Ninth Annual International Conference, July 19-21 (2011)
4. Zhao, D., Traore, I., Sayed, B., Lu, W., Saad, S., Ghorbani, A., Garan, D.: Botnet detection based on traffic behavior analysis and flow intervals. In: 27th IFIP International Information Security Conference (2013)
5. Narang, P., Ray, S., Hota, C., Venkatakrishnan, V.: PeerShark: Detecting Peer-to-Peer Botnets by Tracking Conversations. In: IEEE Security and Privacy Workshops (SPW), San Jose, CA, May 17-18 (2014)

6. Leonard, J., Xu, S., Sandhu, R.: A Framework for Understanding Botnets. In: Proceedings of the International Workshop on Advances in Information Security (WAIS at ARES), Fukuoka, Japan, Fukuoka Institute of Technology, March 16-19 (2009)
7. Shiravi, A., Shiravi, H., Tavallaee, M., Ghorbani, A.A.: Toward developing a systematic approach to generate benchmark datasets for intrusion detection. IComputers & Security, vol. 31, no. 3, pp. 357–374 (May 2012)
8. Ian, E., Witten, H., Trigg, L., Hall, M., Holmes, G., Cunningham, S.: Weka: Practical Machine Learning Tools and Techniques with Java Implementations (1999)
9. Information security and object technology (ISOT) research lab, ISOT Botnet dataset, University of Victoria, obtained from: http://www.uvic.ca/engineering/ece/isot/datasets/index.php
10. The Honeynet Project, French Chapter of Honeynet, obtained from: http://www.honeynet.org/chapters/france
11. Szab'o, G., Orincsay, D., Malomsoky, S., Szab'o, I.: On the validation of traffic classification algorithms. In: Proceedings of the 9th international conference on Passive and active network measurement (PAM'08), Berlin, Heidelberg, pp. 72–81, Springer-Verlag, (2008)
12. LBNL and ICSI, LBNL Enterprise Trace Repository, obtained from: http://www.icir.org/enterprise-tracing.