Netflow-Based Malware Detection and Data Visualisation System

Rafał Kozik(✉), Robert Młodzikowski, and Michał Choraś

Institute of Telecommunication and Computer Science,
UTP University of Science and Technology, Bydgoszcz, Poland
rafal.kozik@utp.edu.pl

Abstract. This paper presents a system for network traffic visualisation and anomalies detection by means of data mining and machine learning techniques. First, this work describes and analyses existing solutions in the field of network anomalies detection in order to identify adapted techniques in that area. Afterwards, the system architecture and the adapted tools and libraries are presented. Particularly, two different anomalies detection methods are proposed.

The key experiments and analysis focus on performance evaluation of the proposed algorithms. In particular, different setups are considered in order to evaluate such aspects as detection effectiveness and computational complexity.

The obtained results are promising and show that the proposed system can be considered as a useful tool for the network administrator.

Keywords: Cyber security · Anomaly detection · Botnet · NetFlow · Visualisation

1 Introduction

Nowadays, one of the cybersecurity challenges is to counter the malicious software [1]. Usually, malware samples are carefully crafted pieces of computer programs that aim at staying dormant while performing detailed surveillance of infected infrastructures and assets. Infected computers commonly connect together over the telecommunication network and form so-called botnet that can be easily centrally controlled by the cybercriminals for different malicious purposes such as DDoS attacks, SPAM distribution, sensitive data thefts, extortion attacks, etc.

In order to combat such cyber threat, one may use different solutions. However, commonly used anti-virus software may not be efficient enough to protect the network. An example is the case of the polish financial sector problem that happened in 2017 [2]. During that attack, the largest system hack in the country's history took place and several banks in Poland have been infected with malware. This particular malware was a new strain of malicious software which

K. Saeed et al. (Eds.): CISIM 2017, LNCS 10244, pp. 652–660, 2017.
DOI: 10.1007/978-3-319-59105-6_56

has never been seen before in live attacks and it had a zero detection rate on VirusTotal.

The advancements in machine learning and data mining techniques in the area of Big Data introduces new tools supporting the fight against the malware. Therefore, in this research, we analyse existing techniques for botnet detection. Moreover, we propose the system that adapts different pattern extraction techniques, classification and visualisation methods.

The main contribution of this work is a proposal of a tool enhancing the cyber security of local area network. The tool intends to support network administrator in network traffic analysis by providing visualisation, data mining and feature extraction capabilities. In the current version of the system we have provided (i) two different pattern extraction algorithms, (ii) a variety of data mining algorithms available via Weka [3] library, (iii) a visualisation module.

The paper is structured as follows. First, we provide an overview of existing solutions and methods for botnets detection. Next, we propose system architecture and different pattern extraction and classification methods for NetFlow analysis. The experiment section presents evaluation methodology and obtained results. This paper is concluded with final remarks and plans for the future work.

2 Related Work

Commonly the signatures (in form of reactive rules) of an attack for a software like Snort [4] are provided by experts from a cyber community. Typically, for deterministic attacks, it is fairly easy to develop patterns that will clearly identify the particular attack. It often happens when given malicious software (e.g. worm) uses the same protocol and algorithm to communicate trough network with command and control centre or other instance of such software. However, the task of developing new signatures becomes more complicated when it comes to polymorphic worms or viruses. Such software commonly modifies and obfuscates its code (without changing the internal algorithms) in order to be less predictive and hard to detect.

The development of an efficient and scalable method for malware detection is currently challenging also due to the general unavailability of raw network data. Therefore, this aspect while being related to users privacy and administrative and legal reasons causes additional difficulties for research and development [5,6].

Currently, the common alternative is so-called NetFlow [7] data that is often captured by ISPs for auditing and performance monitoring purposes. Since NetFlow samples do not contain any sensitive data they are widely available. However, the fact that this kind of samples is lacking raw content of network packets is the disadvantage.

In the literature, there are different approaches focusing on the analysis of NetFlow data. In [8,9] authors focused on computational paradigms (e.g. MapReduce) for NetFlow data analysis and malware detection. On the other hand, in [10,11] author proposed statistical techniques for feature extraction from groups of network flows.

The BClus [12] method uses behavioural approach for botnet detection. It aggregates NetFlows for specific IP addresses and clusters them according to statistical characteristics. The properties of the clusters are described and used for further botnet detection. Another approach is used in BotHunter [13] tool. It monitors the two-way communication flows between hosts within internal network and the Internet. BotHunter employs Snort intrusion detection system. It models an infection sequence as a composition of participants and a loosely ordered sequence of network information exchanges.

3 Proposed System Architecture

In the Fig. 1 the general overview of the system design is presented. The collected raw data is processed in order to extract the NetFlows. The NetFlow is a standardised format for describing bidirectional communication and contains such information as IP source and destination address, destination port, amount of bytes exchanged, etc. The extracted NetFlows are stored in the database for further processing, so that the data mining and feature extraction methods currently work in the batch processing mode. However, in the future, we plan to allow the system to analyse directly the streams of data containing the raw NetFlows.

The single NetFlow usually does not provide enough evidence to decide whenever the particular machine is infected or if the particular request has malicious symptoms. Therefore, it is quite common [12] that NetFlows are aggregated in

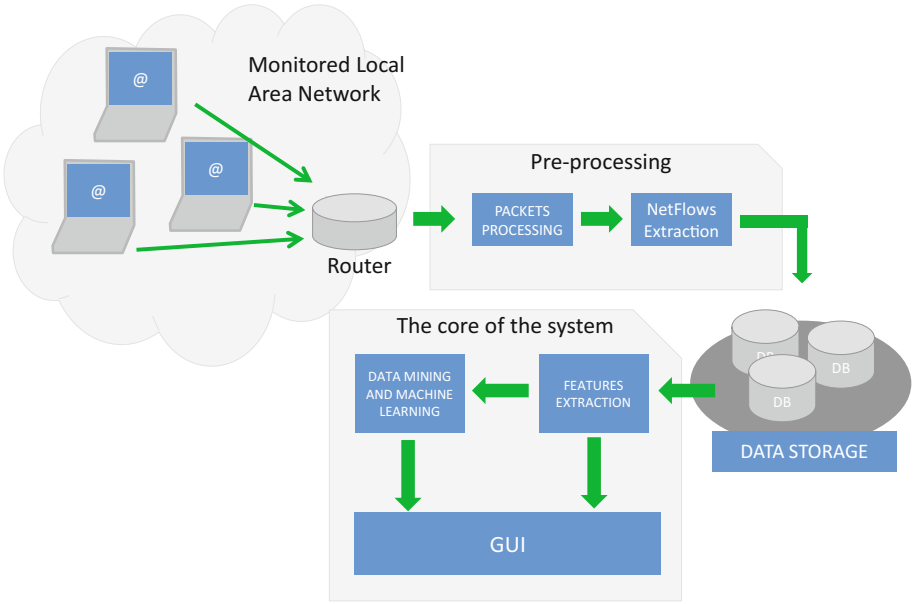

Fig. 1. System architecture diagram

so-called time windows so that more contextual data can be extracted and malicious behaviour recorded (e.g. port scanning, packet flooding effects, etc.). In order to do that different statistics can be extracted for each time window. In the current version of the proposed system, we have implemented two different methods for pattern extraction (the Feature Extraction block on the diagram). These methods have been described in the consecutive subsections. In general, these methods produce the feature vectors that are further used to learn different ML algorithms (the Data Mining and Machine Learning block on the diagram). The machine learning algorithms are available via the Weka [3] library.

The system is also facilitated with graphical user interface (indicated as GUI on the diagram) which allows the network administrator to visualise different statistical properties of the analysed traffic (e.g. amount of data generated by specific IP addresses or the most active ones) as well as classification results. Some aspects of the visualisation process have also been described in separate section.

In order to evaluate the effectiveness of different algorithms, we have used CTU-13 dataset. It contains different scenarios representing different infections and malware communication schemes with command and control. Therefore, in this paper, we do not consider the problem of the realistic testbed construction.

3.1 Method 1

The first feature extraction method aggregates NetFlows in a time window (in this approach we use 1-minute long time windows). One of the reasons behind the aggregation process is the context identification in order to capture relevant behaviours of different hosts. For each time window the following statistics are calculated:

- number of NetFlows
- total sum of transferred bytes
- average sum of transferred bytes per NetFlow
- number of unique destination IP addresses

One of the advantages of this approach is the fact that the number of features vectors is equal to the number of time windows. Therefore, for the short scenarios the size of the resulting dataset will be small and thus the machine learning process will be faster.

However, one of the obvious drawbacks is the fact that for this approach it is impossible to identify the IP address of the infected machine because the system will only signal that particular time window should be considered anomalous.

3.2 Method 2

The second feature extraction method, similar to the previous one, aggregates NetFlows in the time windows. However, for each time window, we additionally group the NetFlow by IP source addresses. For each group (time window, IP source address) we calculate the following statistics:

- number of flows
- sum of transferred bytes
- average sum of bytes per NetFlow
- average communication time between unique IPs
- number of unique IP addresses
- number of unique destination ports
- most frequently used protocol (e.g. TCP, UDP) by specific IP source address

In contrast to the previous method, the advantage of this approach it the fact that it allows the network administrator to identify the possibly infected IP addresses.

3.3 Machine-Learning Module

In our research, we have selected different machine learning algorithms available in the WEKA software package [3]. We have considered such algorithms as NaiveBayes, Logistic, MultilayerPerceptron, SimpleLogistic, IBk, ClassificationViaRegression, LogitBoost, RandomCommittee, RandomizableFilteredClassifier, JRip, PART, J48, RandomForest, RandomTree. During the experiments, we have used different configurations of the algorithms in order to obtain optimal results.

4 Data Visualisation

The visualisation module is dedicated for network administrator in order to facilitate the in-depth analysis of network traffic. Different figures allow for visual

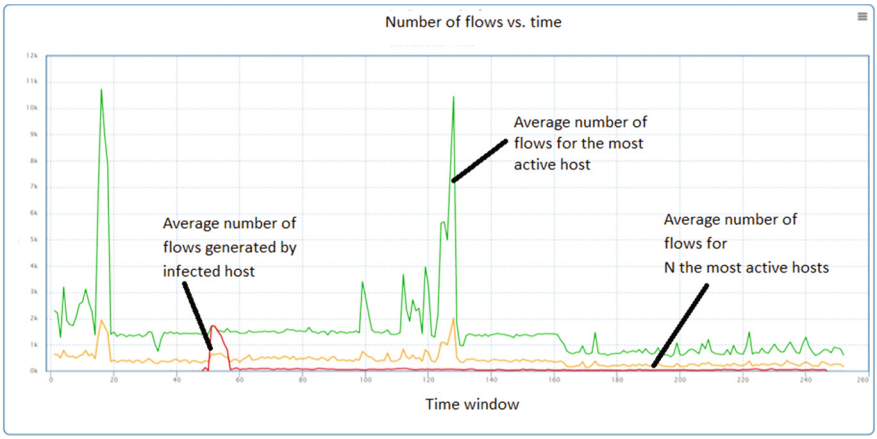

Fig. 2. An example of the proposed system visualisation capabilities for the selected scenario. The figure presents the amount of traffic generated on average by all hosts (green line), by the most active hosts (orange line), and the infected host (red line). (Color figure online)

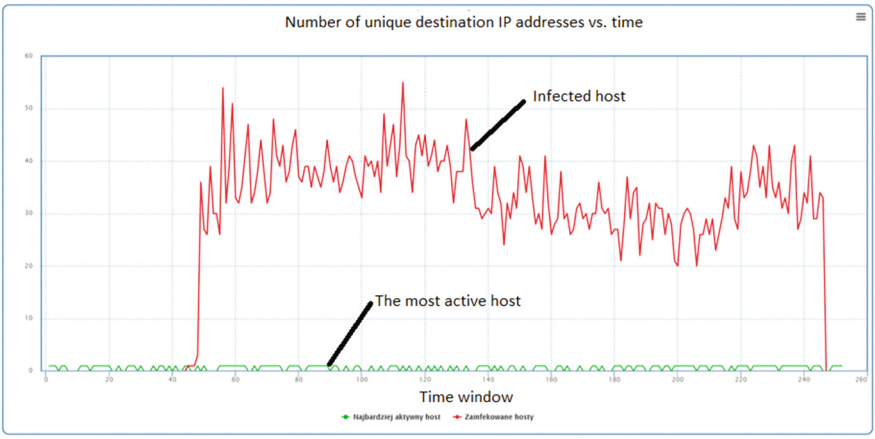

Fig. 3. The figure presents the number of established connection by different hosts for the analysed scenario. It can be noticed that the infected host establishes suspiciously high number of connections.

detection of possibly anomalous network situations (e.g. port scanning). The examples of GUI screenshots (for the same scenario) has been shown in Figs. 2 and 3. The system allows also the administrator to visualise:

- number of NetFlows for different time windows
- amount of bytes transferred for specific search criteria
- destination port utilisation
- results of classification process
- dependencies analysis between possibly infected hosts and other ones

5 Experiments

5.1 Validation Methodology

For the evaluation purposes, we have adapted stratified 10-fold cross-validation methodology. The method was used to assess the TPR - true positives, and FPR - false positives rates.

True Positives Ratio (TPR) is defined as the number samples (feature vectors) identified correctly as infected (True Positives - TP) divided by the number of all samples that are infected (True Positives + False Negatives).

$$TPR = \frac{TP}{TP + FN} \tag{1}$$

False Positives Ratio (FPR) is defined as the number of samples identified wrongly as infected (False Positives - FP) divided by the number of all clean samples (True Negatives + False Positives).

$$FPR = \frac{FP}{TN + FP} \tag{2}$$

The procedure for effectiveness evaluation is following:

- The dataset representing particular scenario in the CTU-13 dataset is divided into 10 parts.
- The 9 parts are used for training the algorithms while the remaining part is used for the evaluation purposes.
- The NetFlows are grouped in time windows.
- For two extraction methods, the feature vectors are extracted (according to the procedure described in Sect. 3).
- Different ML algorithms are trained on the training dataset and the results are obtained on the testing dataset.

The algorithms are learnt and evaluated 10 times and the obtained results are averaged.

5.2 Evaluation Dataset

For the evaluation purposes we have used CTU-13 dataset [12]. This dataset includes different scenarios, which represents different types of attack scenarios including a different type of botnets. Each of these scenarios contains collected traffic in form of NetFlows. As it is explained in [12], the data was collected for the realistic testbed. We have presented the results and the discussion for one of the most problematic scenarios. Each of the scenarios has been recorded in a separate file as NetFlow using CSV notation. Each of the row in a file has 15 attributes (columns):

- StartTime - Start time of the recorded NetFlow,
- Dur - Duration,
- Proto - IP protocol (e.g. UTP, TCP),
- SrcAddr - Source address,
- Sport - Source port,
- Dir - Direction of the recorded communication,
- DstAddr - Destination Address,
- Dport - Destination Port,
- State - Protocol state,
- sTos - Source type of service,
- dTos - Destination type of service,
- TotPkts - Total number of packets that have been exchanged between source and destination,
- TotBytes - Total bytes exchanged,
- SrcBytes - Number of bytes send by source,
- Label - Label - label assigned to this NetFlow (e.g. Background, Normal, Botnet)

It must be noted that the “Label” field is an additional attribute provided by authors of the dataset. Normally, the NetFlow will have 14 attributes and the “Label” will be assigned by the classifier.

Table 1. Effectiveness of different algorithms for Rbot malware activity detection.

Algorithms	Feature extraction method			
	Method 1		Method 2	
	TPR	FP	TP	FP
NaiveBayes	60,0%	33,3%	50,0%	0,2%
Logistic	40,0%	50,0%	33,3%	0,0%
MultilayerPerceptron	40,0%	25,0%	33,3%	0,0%
SimpleLogist	0,0%	0,0%	25,0%	0,0%
IBk	20,0%	41,7%	58,3%	0,0%
CVR	40,0%	25,0%	33,3%	0,0%
LogitBoost	40,0%	16,7%	33,3%	0,0%
RandomCommitte	20,0%	33,3%	66,7%	0,0%
JRip	0,0%	16,7%	50,0%	0,0%
PART	60,0%	50,0%	50,0%	0,0%
J48	60,0%	50,0%	50,0%	0,0%
RandomForest	40,0%	25,0%	66,7%	0,0%
RandomTree	20,0%	16,7%	58,3%	0,0%

5.3 Results

The proposed methods have been evaluated on the scenario concerning the Rbot malware. According to the scenario description, the malware realises ICMP DDoS attack.

The values of TPR and FPR ratios have been presented in Table 1. The results obtained with the second method for feature extraction have achieved better results. The average effectiveness of botnet detection for all the classifiers for the first method is 47.0% while for the second method is 63.0%. However, the classifiers combined with the first method for pattern extraction yielded high FP ratios.

The conclusion from this experiment is that the second feature extraction method combined with RandomForest (or RandomCommittee) allowed us to achieve 66.7% of malware detection while having no false positives.

6 Conclusions

In this paper, we have proposed preliminary results of the malware detection method. Our approach relies on the analysis of malware network activity that is captured by means of NetFlows. We have presented the architecture of the proposed system. The current implementation includes two methods for pattern extraction that analyses the NetFlows in disjoint time windows. The extracted feature vectors have been used to train different machine learning algorithms.

The methods have been evaluated on the publicly available dataset. Future work will be dedicated to the evaluation of scalability of the proposed methods and further improvements towards online machine learning.

References

1. Choras, M., Kozik, R., Renk, R., Holubowicz, W.: A practical framework and guidelines to enhance cyber security and privacy. In: Herrero, A., Baruque, B., Sedano, J., Quintan, H., Corchado, E. (eds.) International Joint Conference CISIS 2015 and ICEUTE 2015. AISC, pp. 485–496. Springer, Heidelberg (2015). ISBN: 978-3-319-19712-8
2. The Hacker News web page. Polish Banks Hacked using Malware Planted on their own Government Site. http://thehackernews.com/2017/02/bank-hacking-malware.html
3. WEKA Data Mining Software. http://www.cs.waikato.ac.nz/ml/weka/
4. SNORT. Project homepage. http://www.snort.org/
5. Andrysiak, T., Saganowski, Ł., Choraś, M., Kozik, R.: Network traffic prediction and anomaly detection based on ARFIMA model. In: Puerta, J.G., Ferreira, I.G., Bringas, P.G., Klett, F., Abraham, A., Carvalho, A.C.P.L.F., Herrero, Á., Baruque, B., Quintián, H., Corchado, E. (eds.) International Joint Conference SOCO 2014-CISIS 2014-ICEUTE 2014. AISC, vol. 299, pp. 545–554. Springer, Cham (2014). doi:10.1007/978-3-319-07995-0_54
6. Choras, M., Kozik, R., Puchalski, D., Holubowicz, W.: Correlation approach for SQL injection attacks detection. In: Herrero, A., et al. (eds.) International Joint Conference CISIS 2012-ICEUTE 2012-SOCO 2012 Special Sessions. AISC, vol. 189, pp. 177–186. Springer, Heidelberg (2012)
7. Claise, B.: Cisco Systems NetFlow Services Export Version 9. RFC 3954 (Informational) (2004)
8. Francis, J., Wang, S., State, R., Engel, T.: Bottrack: tracking botnets using netflow and pagerank. In: Proceedings of IFIP/TC6 Networking (2011)
9. Dean, J., Ghemawat, S.: MapReduce: simplified data processing on large clusters. In: Symposium on Operating Systems Design and Implementation (OSDI). USENIX Association (2004)
10. Lakhina, A., Crovella, M., Diot, C.: Diagnosing network-wide traffic anomalies. ACM SIGCOMM Comput. Commun. Rev. **34**, 357–374 (2004)
11. Lakhina, A., Crovella, M., Diot, C.: Mining anomalies using traffic feature distributions. ACM SIGCOMM Comput. Commun. Rev. **35**, 217–228 (2005)
12. Garcia, S., Grill, M., Stiborek, J., Zunino, A.: An empirical comparison of botnet detection methods. Comput. Secur. J. **45**, 100–123 (2014). Elsevier
13. BotHunter homepage. http://www.bothunter.net/about.html