Comparing and analysing binary classification algorithms when used to detect the Zeus malware

Mohamed Ali Kazi and Steve Woodhead
Faculty of Engineering and Science
University of Greenwich
London, UK
Mk0889h@gre.ac.uk

Diane Gan
Faculty of Liberal Arts and Sciences
University of Greenwich
London, UK

***Abstract*—The Zeus banking malware is one of the most prolific banking malware variants ever to be discovered. This paper examines and analyses the Support Vector Machine (SVM), Decision Tree and Random Forest machine learning algorithms when used in conjunction with a manual feature selection process to detect Zeus network traffic. Selecting the features manually provides the researcher with more control over which features that can and should be selected. The manual feature selection process will also allow the researcher to analyze the impact of the various features and then select the features that provide the best accuracy results during the classification and detection of Zeus. The algorithms in scope for this research are the Decision Tree algorithm, Random Forest algorithm and the SVM algorithm.**

Keywords—Zeus banking malware, Detecting banking malware using binary classification algorithms, decision tree algorithm, random forest algorithm, SVM algorithm, manual feature selection

I. INTRODUCTION

Banking malware is on the increase and a report by [1] states that as of May 2017, the threat from banking malware was two and a half times greater than that of ransomware. A report by [2] also demonstrates a similar pattern and reports that in 2017, 33% of infections were associated with banking malware as compared with the 22% that were associated with ransomware. In [3], the author estimates that some of the highest "earning" banking malware campaigns in 2018 could potentially earn a banking malware operator around US$1M-2M in revenue which is a significant increase from what could typically be "earned" in 2010 (estimated to be between US$100k and $300k). Researchers have developed various methods to detect malware and most of these are either signature based or anomaly-based detection methods. Research is now being conducted to use machine learning algorithms to detect malware and some of these are discussed in section III. This paper expands on this work to enhance and improve the detection accuracy and to also understand the viability to expand the use of machine learning algorithms to detect malware, especially banking malware. The initial focus of this paper is on a notorious banking malware called Zeus (discussed in section II A). This paper uses binary classification machine learning (ML) algorithms to detect and classify the Zeus banking malware network traffic and to also detect and classify benign traffic. An important differentiation between this research and the research discussed in section III is that this research proposes selecting the features manually allowing the authors to have a better understanding of the features and how they can impact the accuracy and detection results. It also provides the authors with more control over which features that are and should be selected by the algorithms. This research also expands on the work reported in [4]. The algorithms in scope for this research are the Decision Tree (DT), Random Forest (RF) and Support Vector Machine (SVM) algorithms. The paper will then compare the performance of these algorithms and the detection accuracy of each algorithm when used with a manual feature selection process. Section II provides an overview of the Zeus banking malware (Zeus), section III discusses previous work, section IV discusses the approach and methodology and section V discusses and analyses the results. Section VI presents the conclusions of the paper.

II. THE ZEUS BANKING MALWARE

According to [5], 9 out of the 10 most dangerous banking malware variants all belonged to the Zeus malware family and these variants include Zbot, Zeus Gameover, SpyEye, Ice IX, Citadel, Carberp, Bugat, Shylock and Torpig. Also, the authors in [6] state that Zeus has become the leading banking malware and is the most advanced credential stealing malware ever discovered. In 2011, the source code for Zeus was leaked, allowing malware authors to develop new variants of Zeus and some of these are listed above. Zeus is a customizable malware allowing malware authors to develop new modules which can be traded in underground forums [7].

A. An Overview of the Zeus Banking Malware

As discussed by [4], Zeus portrays the following characteristics:

- Propagates like a virus and targets Windows systems.
- The infection vector is usually spam email which is used to deliver a Trojan.
- Targets sensitive information such as credentials and banking passwords.

Zeus steals credentials using two methods. The first is by automatic actions hardcoded in the binary which allow Zeus to capture FTP and POP3 passwords [8]. The second is to steal information stored in the Windows PSTORE (protected storage) [8]. One of the most important aspects of the Zeus

malware is the command and control (C&C) communication. As discussed by [4], the communication architecture can either be centralized or peer to peer (P2P). Some versions of Zeus use the P2P architecture and other variants of Zeus use a centralized architecture [9]. An issue with the centralized C&C architecture is that the IP address of the C&C server is hard coded within the Zeus binary itself [9]. If the C&C server becomes unreachable or is taken down, the Zeus bots will not be able to communicate with the C&C server preventing the bots from receiving commands, updating themselves and downloading new configuration files [9].

Newer variants of Zeus use the P2P C&C architecture. These are much harder to block and are also more resilient to takedown efforts because the configuration file does not point to a static C&C server [9]. Instead, the C&C server information is obtained from a peer (proxy bot) which can be updated if the C&C server is taken down or becomes unreachable [10]. Stolen data is routed through the C&C network to the malware authors' C&C server where the stolen data is decrypted and saved to a database [11].

III. Related Work

A. Detecting Malware Infections using IDS Driven Dialogue Correlation

In [11], the authors proposed a passive monitoring system called Bothunter which consists of a correlation engine and three malware sensors. The first sensor uses SNORT and the other two sensors, SLADE and SCADE, were custom developed. SCADE is a Snort pre-processor which consists of two scan engines that scan inbound and outbound traffic. SCADE scans traffic based on the following criteria: Local hosts that conduct high-rate scans to external addresses; Outbound connection failures; A uniformly distributed scan pattern (a pattern which is likely to be malware). SLADE is a payload analysis engine which alerts the administrator if the byte distribution of the packets deviates from an established profile. SLADE is based on PAYL which examines the distribution of the payload i.e. it extracts 256 features from the payload and represents each feature based on its occurrence frequency [11].

Bothunter was developed as a perimeter scanning solution which attempts to detect botnet activities and attempts to detect network activity that occurs between an infected host and an external entity. The alerts that are generated are fed into a dialogue correlation tool which tracks activities over a temporal window and attempts to identify malware infections. Bothunter can also enhance the prediction of the malware activity by correlating some of the malicious outbound flows to inbound flows. Malware activities can be identified using the following characteristics:

- External to internal inbound scan (Bot scanning or direct exploit).
- External to internal inbound exploit (Bot scanning or direct exploit).
- Internal to external binary acquisition (Bot binary download).
- Internal to external C&C communication (C&C server activities).
- Internal to external outbound infection scanning (Attack propagation).

The correlation engine attempts to determine if the traffic patterns are malicious or benign. However, the limitation of Bothunter is that it is not able to scan local network traffic i.e. local DNS resolution traffic or the traffic generated by malware when it scans a local network for vulnerable hosts. Malware can also evade Bothunter by encrypting its payload as Bothunter is unable to inspect encrypted traffic.

B. Detecting Bots Using the C4.5 and CFS Feature Selection Algorithm

In [12] the authors proposed a machine learning (ML) approach called CONIFA to detect network traffic generated by Zeus. The authors in [12] used the C4.5 classification algorithm and the Correlation-Based feature selection (CFS) algorithm (automated) for feature selection to train and classify the ML algorithm to identify Zeus network traffic. The author in [12] developed a Cost-Sensitive C4.5 classification algorithm which used both a lenient and strict classifier to enhance CONIFA's ability to detect network traffic. CONIFA was actually developed to detect and classify application traffic generated by Skype however the author in [12] also evaluated CONIFA against the Zeus banking malware.

In [12] the authors also discussed a 'standard framework' which used the Cost-Insensitive version of C4.5 to classify network traffic and identify Zeus. The author in [12] then compared CONIFA's accuracy results with the 'standard framework' and the results showed that CONIFA was more effective than the standard framework at detecting Zeus network activity. The standard framework's detection rate was good when evaluating the training dataset, however, when evaluating the test data, the recall rate dropped to 56% resulting in around half of the Zeus network flows going undetected. CONIFA's results demonstrated that there was an improvement in detecting the Zeus network traffic with the accuracy rate increasing to 67%.

C. Detection of Randomized Bot Command and Control Traffic on an End-Point Host

In [13] the authors proposed a solution called RCC Detector (RCC) which analyses network traffic generated by a host to identify bot traffic and benign traffic. RCC aims to identify bots in the early phases of infection and also aims to detect bots that randomize their communication activities in an attempt to hide. RCC uses a Multi-Layer Perceptron (MLP) classifier and a Temporal Persistence (TP) classifier to identify botnet communication activities generated by a host. The MLP classifier has an input layer, an output layer and one hidden layer which has four neurons. The following criteria were used for the classification: Flow count - Flows that are counted over a period of time; Session length - Non-botnet HTTP traffic is bursty occurring over a short period of time whereas bot traffic is generally low profile persisting over a longer period of time;

Uniformity score - This is based on packet count values. Bots portray regularity in packet counts while benign traffic typically shows more varied packet counts; Kolmogorov–Smirnov Test - Which is used to compute the distance between flows. Benign traffic is bursty and generates traffic at very close time intervals whilst bots generate traffic at larger time intervals. These four features were used to calculate a temporal persistence value which was used by the classifier to detect botnet activities. The classifier was evaluated against a selection of bots and the authors in [13] reported that 99.8% of bots being identified from the samples however, the False Positive (FP) rate was calculated to be 48%. Although the results were promising, the false positives were quite high and more importantly the software tool was designed as a host-based detection method. This makes it difficult to implement as it would need to be installed on all the hosts in a network.

IV. Research Approach and Methodology

A. Introduction

ML algorithms have been used by researchers to detect banking malware and some of these were discussed in section III, however, these algorithms use automated feature selection algorithms to detect and select the appropriate features for the ML algorithms or were designed as a host-based detection tool. Also, Bothunter which was discussed in section III is unable to detect encrypted communication traffic which is an inherent weakness of signature based tools. There is no indication of research being conducted that uses binary classification algorithms alongside a manual feature selection method to detect Zeus network traffic. This research proposes selecting the features manually for the Decision Tree (DT), Random Forest (RF) and SVM classification algorithms allowing the appropriate features to be discovered manually. This allows greater flexibility and more control over which features that can be used by the algorithms and which features produce the best detection results. Precision, Recall and F-score will be used to determine the accuracy of the ML algorithms and a confusion matrix will also be generated, as shown in table 1, which identifies how many Zeus flows and benign traffic flows were correctly identified. The following variables are used to quantify the accuracy of the ML algorithms, and are defined below:

$$Recall = TP / (TP + FN) \quad (1)$$

$$Precision = TP / (TP + FP) \quad (2)$$

The F-score is the calculated mean of the Recall and Precision scores. If this value is 1 then the accuracy of the algorithm is 100% and if the value is 0 then the accuracy of the algorithm is 0%.

B. Research Methodology

Zeus can obfuscate its payload by encrypting the network datagrams however, this research will not attempt to decrypt the network communication traffic during the classification of the Zeus malware. This research will use the Decision Tree (DT), Random Forest (RF) and SVM algorithms to detect and classify the communication traffic as either malware or benign. This research focused on the general datagram structure of the network packets for the classification and detection activities and the following framework, described by the authors in [14] was followed:

- Obtained the Zeus and benign traffic samples and then split the data into a training and evaluation dataset.
- Selected the appropriate algorithms for the task and trained and evaluated the ML algorithms.

Table 1. Confusion matrix used to measure the detection accuracy

	Predicted Zeus	Predicted Benign
Actual Zeus	(TP)	(FN)
Actual benign	(FP)	(TN)

C. Data Collection

The Zeus samples for this research were obtained from Zeustracker [21], a website which monitors Zeus C&C activities, and were downloaded as pcap files in February this year. The benign traffic samples were collected manually from a new installation of Windows 10, version 10.0.17763.615, and were also collected during the same time. To ensure that both the Zeus and benign samples were comparable, 524 samples of each were used during this research (total of 1048). From each sample, 44 statistical features were extracted and a description of all the features can be found at [22]. Through experimentation, the following 13 features were identified to produce the best results and were subsequently used during this research: Total_fpackets - Total packets in the forward direction; Total_fvolume - Total bytes in the forward direction; Total_bpackets - Total packets in the backward direction; Total_bvolume - Total bytes in the backward direction; Min_fpktl - The size of the smallest packet sent in the forward direction (in bytes); Mean_fpktl - The mean size of packets sent in the forward direction (in bytes); Max_fpktl - The size of the largest packet sent in the forward direction (in bytes); Std_fpktl - The standard deviation from the mean of the packets sent in the forward direction (in bytes); Min_bpktl - The size of the smallest packet sent in the backward direction (in bytes), Mean_bpktl - The mean size of packets sent in the backward direction (in bytes), Max_bpktl - The size of the largest packet sent in the backward direction (in bytes), Std_bkptl - The standard deviation from the mean of the packets sent in the backward direction (in bytes), Min_fiat - The minimum amount of time between two packets sent in the forward direction (in microseconds).

D. Data Preparation

To prepare the data for the ML algorithms, the statistical features from the datagrams were extracted using Netmate-flowcalc (NF), a tool developed by [15], and then exported into a CSV file. A flow is a sequence of packets flowing between a source and a destination during a specific

period of time. NF was used because it is an open source tool that can extract the statistical features required by the ML algorithms and has been used by other researchers such as [12]. NF extracts 44 features of which 13 were used during this research and were discussed in section IV C. The statistical features of both Zeus and the benign traffic were then exported into a Pandas' data frame and an additional column called 'is_botnet' was created and was used as the label which is what the ML algorithms were trained against. The label was set to 1 to identify Zeus traffic and 0 to identify benign traffic. These were then combined and randomized in another Panda's frame and a sample of this dataset can be seen in table 2.

E. Feature Selection and Training the Dataset

One of the main issues in ML is selecting the appropriate features from the dataset. In [16] the author states that a dataset could have many features and selecting the best features has many benefits and these include:

- Variance (overfitting) is reduced, as this can produce incorrect results.
- Computational cost and the time for running the algorithm is reduced.
- Enables the ML algorithm to learn faster.

In [17] the authors discuss several techniques that can be used for feature selection and these predominately include:

- Filter methods - Feature selection is independent of the machine learning algorithm.
- Wrapper methods - A subset of the features are selected and the ML algorithm is trained. Based on the results, features are either removed or added. The test is then repeated until the required results are obtained.

The analysis conducted for this research used the wrapper method allowing the features to be manually selected. The number of features were increased and decreased manually which had an impact on the detection accuracy and the experimental results of this are discussed in section V. K-Fold cross validation (10-Fold) was used to analyze the training dataset and 70% of the dataset was used for training and validation and the remaining 30% of the dataset was used for testing. There were a total of 1048 samples used in this research experiment of which 524 belonged to Zeus and 524 belonged to benign traffic.

TABLE 2. COMBINED ZEUS AND BENIGN STAISTICS IN A SINGLE PANDAS' FRAME

sflow_fbytes	sflow_bpackets	sflow_bbytes	fpsh_cnt	bpsh_cnt	furg_cnt	burg_cnt	total_fhlen	total_bhlen	is_botnet
228	0	0	0	0	0	0	28	0	1
1204	10	4207	4	4	0	0	412	412	0
1280	9	4012	4	3	0	0	412	372	0
314	2	393	1	1	0	0	292	172	0
172	0	0	0	0	0	0	28	0	1
271	6	2594	0	0	0	0	48	120	1
668	0	0	0	0	0	0	48	0	1
65	1	130	0	0	0	0	28	20	0
321	0	0	0	0	0	0	468	0	1
334	6	2579	0	0	0	0	48	120	1

V. RESEARCH RESULTS

A. Evaluation of the Decision Tree (DT) Algorithm

The DT algorithm is one of the best and most important ML algorithms used for predictive modelling and is very good at predicting binary decisions and is very fast [18]. For this reason, it is well suited for this prediction problem as this analysis is aiming to detect if the datagram is Zeus or benign. The DT algorithm was trained by manually adding and removing features and it was observed that the best results were obtained when using the 13 features discussed in section IV C. Best is defined by the F-score and was calculated around 93% for Zeus and around 95% for the benign traffic. The accuracy results can be seen in table 3. The DT algorithm was then tested using the same 13 features and the predication results are shown in table 4. Table 4 shows that out of 158 Zeus samples 12 were incorrectly classified and out of 157 benign traffic samples 7 were incorrectly classified.

B.Evaluation of the Random Forest (RF) Algorithm

According to [19], the RF algorithm is a supervised ML algorithm which builds and combines multiple decision trees and can reduce overfitting and variance and can provide more accurate results than other binary classification algorithms. The RF algorithm was trained using the wrapper method and it was determined that the best results were achieved by using the following 3 features: total_fpackets; total_fvolume; total_bpackets. The accuracy results of using these 3 features can be seen in table 5 which show that the accuracy for Zeus was around 95% and at around 93% for benign traffic. However, for the comparison conducted during this research, the 13 features discussed in section IV C were also used during the analysis and the accuracy results of using these 13 features is depicted in table 6.

TABLE 3. ACCURACY RESULTS OF TRAINING THE DT ALGORITHM

	Zeus	**Benign**
Precision	93	93
Recall	92	96
F-score	93	95

TABLE 4. CONFUSION MATRIX USED TO MEASURE THE DETECTION ACCURACY

	Predicted Zeus	**Predicted Benign**
Actual Zeus	146	12
Actual benign	7	150

TABLE 5. ACCURACY RESULTS OF TRAINING THE RF ALGORITHM USING 3 FEATURES

	Zeus	**Benign**
Precision	93	93
Recall	96	93
F-score	95	93

TABLE 6. ACCURACY RESULTS OF TRAINING THE RF ALGORITHM USING THE 13 FEATURES DESCRIBED IN SECTION IV C

	Zeus	**Benign**
Precision	89	1
Recall	1	84
F-score	93	92

The RF algorithm was then tested using the 3 best features and the results can be seen in table 7. The 13 features discussed in section IV C were also used during the testing and the results of this can be seen in table 8. Table 7 shows that out of 164 Zeus samples 7 were incorrectly classified and out of 151 benign traffic samples 11 were incorrectly classified. Table 8 shows that out of 158 Zeus samples 0 were incorrectly classified and out of 157 benign traffic samples 25 were incorrectly classified.

B. *Evaluatoin of the SVM Algorithm*

According to [20], the SVM algorithm is a supervised ML algorithm which directly gives us the resultant classes, in this case the class is either Zeus or benign traffic. It works well with both structured and unstructured data and is able to solve complex problems and [20] also states that SVM can reduce overfitting and can produce good results. The SVM algorithm was trained using the same 13 features discussed in section IV C and the results are depicted in table 9 which shows an F-score of 91% for Zeus and 92% for benign traffic. The SVM algorithm was then tested using the same 13 features and table 10 shows the predication results. Table 10 shows that out of 158 Zeus samples 17 were incorrectly classified and out of 157 benign traffic samples 10 were incorrectly classified.

C. *Analysis*

Manually selecting and using the 13 features discussed in section IV C for the three algorithms have produced acceptable detection results and the accuracy of these algorithms are compared in Fig. 1. The DT algorithm produced an F-score of 93%, the RF algorithms produced an F-score of 95% and the SVM algorithm produced F-score of 91% when classifying Zeus. This shows that the RF algorithm performed the best with DT in second place and SVM in third.

TABLE 7. CONFUSION MATRIX USED TO MEASURE THE DETECTION ACCURACY OF RF WHEN USING 3 FEATURES

	Predicted Zeus	**Predicted Benign**
Actual Zeus	157	7
Actual benign	11	140

TABLE 8. CONFUSION MATRIX USED TO MEASURE THE DETECTION ACCURACY OF RF WHEN THE 13 FEATURES DISCUSSED IN SECTION IV C

	Predicted Zeus	**Predicted Benign**
Actual Zeus	158	0
Actual benign	25	132

TABLE 9. ACCURACY RESULTS OF TRAINING THE SVM ALGORITHM

	Zeus	**Benign**
Precision	93	90
Recall	89	94
F-score	91	92

TABLE 10. CONFUSION MATRIX USED TO MEASURE THE DETECTION ACCURACY OF SVM

	Predicted Zeus	**Predicted Benign**
Actual Zeus	141	17
Actual benign	10	147

A comparison of all the algorithms' precision, recall and F-scores using the same 13 features can be seen in Fig.2 and shows that the best performing algorithm was RF for detecting Zeus with SVM performing the worst. Fig. 3 shows how each of the algorithms performed when detecting benign traffic and Fig 4 shows the false positive rates for the algorithms when detecting Zeus. The analysis shows that the DT algorithm performs the best when detecting benign traffic with RF coming second. However, it can be noted that the RF algorithm doesn't mis-classify any of the Zeus network datagrams.

VI. CONCLUSION

This research has shown that ML algorithms using a manual feature selection process can be a viable solution for detecting banking malware. It has also shown that using a manual feature selection process produces better results than those produced by automated feature selection processes as discussed by the authors in section III. Furthermore, this research has shown that the Zeus banking malware can be detected even though it may be encrypted thus resolving the issue faced by signature-based detection techniques.

A. *Further Work*

It is acknowledged that further testing should be conducted on newer variant of Zeus and there is a need to test against other banking malware variants such as Neverquest. There is also a need to test against a larger dataset to enhance the detection results. Additionally, the manual feature selection can be expanded to other binary classification algorithms and then further expanded and tested on other machines learning approaches such as unsupervised machine learning algorithms.

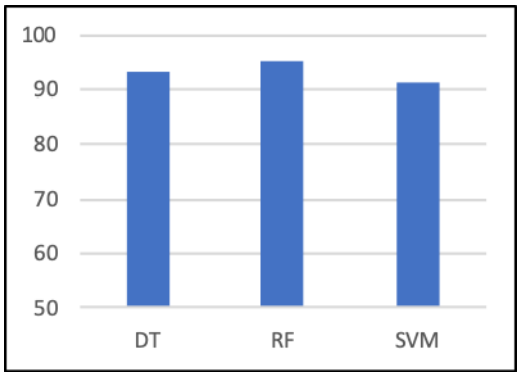

Fig.1. Accuracy (f1) results for detetcing Zeus when using the 13 features discussed in section IV C

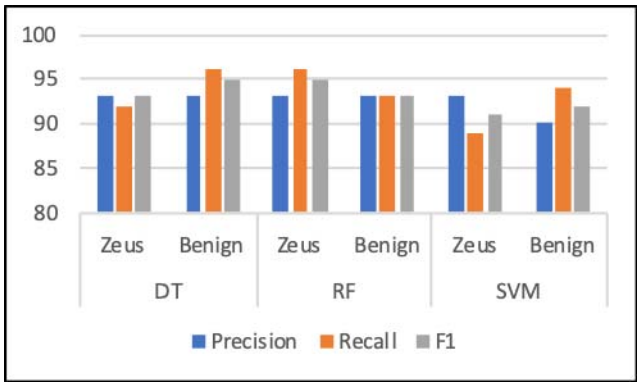

Fig.2. Comparison of the DT, RF and SVM training results when used with the manual feature selection process.

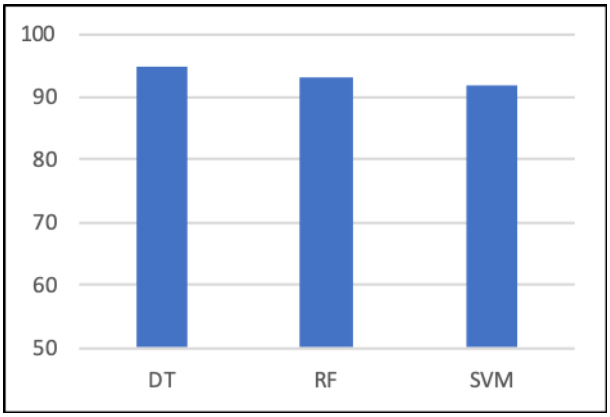

Fig.3. Comparison of the algorithms detection results for benign traffic.

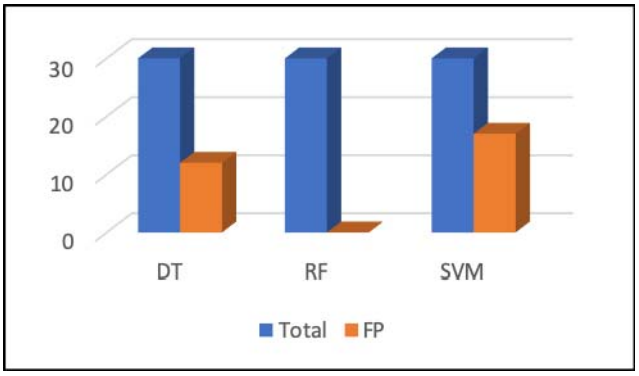

Fig.4. Comparison of the false positive rates of the DT, RF and SVM algorithms when deteing Zeus

REFERENCES

[1] Falliere, N. and Chien, E. "Zeus: King of the bots." https://www.forbes.com/sites/stevemorgan/2015/11/24/ibms-ceo-on-hackers-cyber-crime-is-the-greatest-threat-to-every-company-in-the-world/#1075f39873f0, (accessed Nov, 3, 2019).

[2] Jang-Jaccard, J. and Nepal, S., "A survey of emerging threats in cybersecurity," in Journal of Computer and System Sciences, Aug 2014. [online]. Available: https://www.sciencedirect.com/science/article/pii/S0022000014000178.

[3] Clarke, J. "Cyber crime 'cost UK residents £210 each in the last year." http://www.independent.co.uk/news/uk/crime/cyber-crime-hacking-fraud-213-a-year-a7365816.html (accessed May. 30, 2017).

[4] Kazi, M., Woodhead, S. and Gan, D, "A contempory Taxonomy of Banking Malware," presented at the First International Conference on Secure Cyber Computing and Communications. Jalandhar, India, Feb. 25, 2019.

[5] Zaharia, A. "The Top 10 Most Dangerous Malware That Can Empty Your Bank Account." https://heimdalsecurity.com/blog/top-financial-malware/ (accessed Aug. 17, 2019).

[6] Etaher, N., Weir, G.R. and Alazab, M., "From zeus to zitmo: Trends in banking malware," in Researchgate, Aug, 2015. [Online]. Available: https://www.researchgate.net/publication/308809590_From_ZeuS_to_Zitmo_Trends_in_Banking_Malware.

[7] Ibrahim, L.M. and Thanon, K.H., "Analysis and detection of the zeus botnet crimeware," in International Journal of Computer Science and Information Security, Sep, 2015. [Online]. Available: https://www.researchgate.net/publication/329220586_Analysis_and_Detection_of_the_Zeus_Botnet_Crimeware.

[8] Falliere, N. and Chien, E., "Zeus: King of the bots." https://www.forbes.com/sites/stevemorgan/2015/11/24/ibms-ceo-on-hackers-cyber-crime-is-the-greatest-threat-to-every-company-in-the-world/#1075f39873f0 (accessed Aug. 17, 2019).

[9] Researcher), L., "Gameover: ZeuS with P2P Functionality Disrupted," in TrendLabs Security Intelligence Blog, Jun, 2014. [Online]. Available: https://blog.trendmicro.com/trendlabs-security-intelligence/gameover-zeus-with-p2p-functionality-disrupted/.

[10] Wyke, J. "Gameover malware returns from the dead" https://nakedsecurity.sophos.com/2014/07/13/gameover-malware-returns-from-the-dead/ (accessed Sep. 18, 2017).

[11] Gu, G., Porras, P.A., Yegneswaran, V., Fong, M.W. and Lee, W., "Bothunter: Detecting malware infection through ids-driven dialog correlation," in Researchgate, Jan, 2008. [Online]. Available: https://www.researchgate.net/publication/221260587_BotMiner_Clustering_Analysis_of_Network_Traffic_for_Protocol-_and_Structure-Independent_Botnet_Detection.

[12] Azab, A., Alazab, M. and Aiash, M., "Machine learning based botnet identification traffic," in Researchgate, Aug, 2016. [Online]. Available: [https://www.researchgate.net/publication/313539120_Machine_Learning_Based_Botnet_Identification_Traffic.

[13] Soniya, B. and Wilscy, M., "Detection of randomized bot command and control traffic on an end-point host," in Alexandria Engineering Journal, Nov, 2015. [Online]. Available: https://core.ac.uk/download/pdf/82700010.pdf.

[14] Mayo, M. "Frameworks for Approaching the Machine Learning Process." https://www.kdnuggets.com/2018/05/general-approaches-machine-learning-process.html (accessed May. 20, 2019].

[15] Zander, S. and Schmoll, C. "DanielArndt/netmate-flowcalc." https://github.com/DanielArndt/netmate-flowcalc (accessed Apr. 1, 2019].

[16] Albon, C. "Feature Selection Using Random Forest." https://chrisalbon.com/machine_learning/trees_and_forests/feature_selection_using_random_forest/ (accessed Apr. 4, 2019).

[17] Kaushik, S. "Feature Selection methods with example." https://www.analyticsvidhya.com/blog/2016/12/introduction-to-feature-selection-methods-with-an-example-or-how-to-select-the-right-variables/ (accessed Jun. 4, 2019).

[18] Anand, G. "TOP 10 Machine Learning Algorithms." https://blog.goodaudience.com/top-10-machine-learning-algorithms-2a9a3e1bdaff (accessed May. 31, 2019).

[19] Liberman, N. "Decision Trees and Random Forests." https://towardsdatascience.com/decision-trees-and-random-forests-df0c3123f991 (accessed Jun. 2, 2019).

[20] Statinfer.com. "SVM: Advantages Disadvantages and Applications – Statinfer." https://statinfer.com/204-6-8-svm-advantages-disadvantages-applications/ (accessed Aug. 19, 2019).

[21] Huessy, R. "Zeustracker.abuse.ch." https://zeustracker.abuse.ch/ (accessed Feb. 1, 2019).

[22] Code.google.com.https://code.google.com/archive/p/netmate-flowcalc/wikis/Features.wiki (accessed Mar. 1, 2019).