ISSN: 0975-833X

Available online at http://www.journalcra.com

International Journal of Current Research
Vol. 15, Issue, 02, pp.23809-23813, February, 2023
DOI: https://doi.org/10.24941/ijcr.44849.02.2023

INTERNATIONAL JOURNAL OF CURRENT RESEARCH

RESEARCH ARTICLE

CRIMEWARE TOOLKIT SIGNATURE GENERATION AND DETECTION USING MACHINE LEARNING

[1,] *Yasir N. S. Alkhateem and [2]Prof Mejri, M.

[1]Sudan University of Science and Technology, Khartoum, Sudan
[2]Laval University, Quebec, G1K 7P4, Canada

ARTICLE INFO

Article History:
Received 24th November, 2022
Received in revised form
27th December, 2022
Accepted 20th January, 2023
Published online 28th February, 2023

Key words:

Malware Toolkits; Deep Learning; Malware; Fixed-Target Training; Automatic Signature Generation; Malware Toolkit Signature; Zeus.

***Corresponding Author:**
Yasir N. S. Alkhateem

ABSTRACT

Background: Nowadays, malware samples are automatically created by a custom crimeware toolkit that essentially creates a batch of functionally of the same malware in a different look using different obfuscation techniques and renders static signature-based detection. While there has been substantial research in automated malware classification, it remains challenging in the research community and the main role of crime ware toolkits in the explosion of crime ware has been ignored. ***Objective(s):*** Although such approaches have shown satisfactory performance on a large set of datasets, practical defense systems require precise detection during malware outbreaks where only a handful of samples created using a certain toolkit are available. The problem of toolkit signature generation and detection aims at detecting whether a binary file is created by a given toolkit or not. It has many security applications including signature generation and detection for crime ware toolkits, packers, and metamorphism engines. ***Methods:*** This paper presents a novel deep learning-based model for malware toolkit signature generation and detection. The method uses a deep belief network (DBN), implemented with a deep stack of denoising auto encoders trained by the fixed-target strategy for generating an effective toolkit signature that helps detect new malware samples generated using the same toolkit once a handful of malware samples are available. ***Results:*** The results show how powerful the toolkit signatures generated by the DBN allow for the accurate detection of new malware samples. Using a dataset containing a few training samples created by the same toolkit (Zeus), our method achieves up to 97% detection accuracy using 10 training samples and 1800 test samples, 0.002 sec average detection time (including sample preprocessing time), and 3.08 sec average model build time. Additionally, we introduce a novel concept of toolkit signature.

Citation: ***Yasir N. S. Alkhateem and Prof Mejri, M.*** **2023.** "Crimeware Toolkit Signature Generation and Detection using Machine Learning". *International Journal of Current Research*, 15, (02), 23809-23813.

INTRODUCTION

Historically, stealth malware has been difficult to write and thus relatively rare. However, commercial crime ware operations like Zeus have changed everything. What are Malicious Software Tools(Malware Toolkits)? According to Beek *et al.* (2) Malware Toolkits are malicious software programs that have been designed for automatically creating viruses, worms, or trojans, conducting DoS attacks on remote servers, hacking other computers, and more. They are divided into many subclasses (Constructor, DoS, Email-Flooder, Flooder, IM-Flooder, SMS-Flooder, Hack-Tool, Hoax, Spoofer, and VirTool) according to the payload. The most dangerous subclasses are Constructors and VirTool that used for creating new viruses, worms, trojans, and modifying other malicious programs – to try to prevent an antivirus solution from detecting the malicious software. Although they do not present a direct threat to the computer that it's actually running on, they are considered the spearhead or swiss army knife in malware development. According to the McAfee reports (1) and (3), 43 rootkit samples were found in January 2007, 133090 unique samples are found in July 2010, and in 2013 about an average of 160,000 new malware samples appeared every day according to (4).

The total cost of cybercrime has been estimated to exceed US $ 388 billion annually which could rise to USD 10.5 trillion by the year 2025 as mentioned by Morgan (5). That is mainly due to the availability of crimeware toolkits that lower the bar for entry to the world of cybercrime; with very little technical knowledge required, Cyber criminals can create, and deploy malware through a point-and-click graphical user interface that can cost less than US $ 1000. Technical support is also available for a fee, including technical infrastructure and servers to store harvested data (5). Furthermore, malware generation toolkits like Zeus (6) can generate thousands of variants of the same malware by using different obfuscation techniques. Traditional signature generation is often a human-driven process that is bound to become infeasible with the current malware growth. Smith *et al.* (7), highlight the increasing professionalism of cybercrime, with the emergence of cybercrime products (i.e. sales of malware creation software) over the past year. Nowadays, malware samples are automatically created by a custom crimeware toolkit that essentially creates a batch of functionally of the same malware in a different look using different obfuscation techniques. It renders static signature-based detection obsolete. Nonetheless, the observation is the distribution of the program instruction sequence remains relatively intact.While there has been substantial research in automated malware classification, it remains challenging in the research community and the aforementioned evidence of the main role of crimeware toolkits in the explosion of crimeware has been ignored. Today's signatures are pattern-matching rules commonly defined on static or dynamic properties of applications under analysis and, even though they are assisted by heuristics and AI-based solutions, still represent the most reliable antivirus technology as mentioned by Atzeni *et al*. (8). So, various methods have been proposed and applied for automatic malware signature generation, e.g., signatures based on specific vulnerabilities, payloads, honeypots, etc. Hence, these methods target specific aspects of malware, malware developers are able to use toolkits that automatically create new undetected malware by modifying small parts of their code. The problem of toolkit signature generation and detection aims at detecting whether a binary file is created by a given toolkit or not. It has many security applications including signature generation and detection for crimeware toolkits, packers, and metamorphism engines. Since every tool is an idea or a style for software creation, it has a systematic approach, therefore it has fingerprints. In this paper, we present a novel method for signature generation that does not rely on any specific aspect of the malware and works by extracting the common features of the training data based on the fixed-target training strategy introduced by Alkhateem and Megri (9), thus being able to detect any new malware having the same features. The method relies on training deep denoising autoencoders(AE), which would create a compact representation of the common features of the malware samples created using the same crimeware toolkit. Additionally, we introduce a novel concept of toolkit signature.

The proposed method consists of the following steps in the supervised training phase: Given a dataset of malware binaries created by the same toolkit. First, we use a fixed-target strategy for training a deep belief network implemented using deep denoising autoencoders(AE) and use the encoder part of the trained AE as the Signature Generator and use it to generate the toolkit signature. Next, depending on the toolkit signature, we build a Chi-Square Predictor that we use to decide whether a sample is created by the same toolkit or not, based on its signature. Our dataset contains 900 samples of Zeus Banking Trojan and 910 benign samples, for a total of 1810 samples. The trained DBN generates a signature for each test program sample. The quality and representation power of these generated signatures is examined by the Chi-Square Predictor. The results show that the chi-square predictor achieves up to 97% classification accuracy when tested on the signature of unseen data generated by the DBN trained by 10 samples only, which attests to the representation power of the signatures due to the fixed-target strategy. In the next section, we review several previous approaches for automatic signature generation. In Section III, we describe our approach, and Section IV presents the implementation details and experimental results. Section V contains our concluding remarks.

RELATED WORK

The topic of toolkit signature is not available in the literature. So, we tried to review works related to malware signature generation and zero-day malware detection, specifically paying attention to the problems of the most advanced techniques and strategies used by security researchers that make these methods ineffective against zero-day malware, as stated in (10, 11, 12). There are several problems with the current approaches as mentioned by Filiol *et al*. (10). First, they have high FP and FNR. Second, they require a large dataset and a large feature set for training. Third, they have a very long detection time according to Rhode *et al*. (13). These problems hinder the early detection of malware. A few works tried to address these problems. Gandotra *et al*. (14), address the first and second problems by selecting the relevant set of features, so the building time could be reduced and the accuracy would be improved. They selected seven features to build the classification model using machine learning algorithms in Waikato Environment for Knowledge Analysis (WEKA) library. Their test results showed that random forest gave the best accuracy with 99.97% and the time to build a model was 0.09 s. However, the data preprocessing time was ignored. Moreover, a large dataset was used for training, which is not the case during the outbreak period. The third problem was addressed in the work done by Khammas *et al*. (15), by minimizing the n-gram features space and using sub-signatures to detect the new malware before or during the software installation by analyzing the binary files of the malware files. The result shows that using a very small number of Snort sub-signature 4-gram features (500 features only) will minimize the searching space of n-gram features and achieves a detection accuracy of more than 99.78% and zero FPR. Park *et al*. (12) introduced a method that detects malware variants with a handful of malware samples captured at the very early outbreak as what occurs in a real-world malware detection scenario, with high accuracy for malicious samples and low false positives for benign samples. They use a small number of samples for training with an adversarial auto-encoder and achieved 98.9% detection accuracy. Several approaches have been suggested to improve the signature generation and detection process. Authors of DeepSign (16) apply DBN to solve the problem of malware signature generation and classification, DeepSign uses the Cuckoo sandbox to record the execution behavior of each malware. Then, it uses uni-grams to convert each report into a 20,000-bit vector. The bit vectors are then fed into the deep belief network to generate signatures. Finally, the signatures are fed into a support vector machine for classification. Experiments on 1800 malware samples without benign applications show that DeepSign is able to reach 96.4% accuracy. However, in order to obtain the high reliability of the experiment, the test set should contain, in addition to the malware, also benign applications, since the rate of false positives is no less important than the accuracy. Note that there are many similarities between our approach and that used in (16, 17), as all use deep autoencoders to create short signatures for the content; in our case, the content is the program binary, in the former the content is the high-level behavior of the program, and in the former's case, it is the high-level objects appearing in the image.

Sewak *et al*. (18) used only the opcode frequency directly for feature selection/ extraction, and experiments with different combinations of Deep Learning architectures including Auto-Encoders, and Deep Neural Networks with varying layers over Malicia malware dataset to improve the accuracy and a False Positive Rate. they improved the accuracy by 3.31% (99.21% compared to 95.9%) while simultaneously improving FPR by 10.5 times (0.19% compared to 2.0%) from the previous best results of studies done (using opcode frequency directly) on Malicia. However, the data was not complex enough to get the best out of the deepest AE and DNN combination in the study. Moreover, multiple thresholds in the system need to be adjusted for each different dataset. Vinayakumar *et al*. (19) focused on malware detection based on process behavior in possible infected terminals. They applied DNN in 2 stages, the first stage is for extracting process activities by RNN and converting them into feature vectors. Feature vectors were then treated as images that were classified by CNN. They assumed the framework is scalable and capable of analyzing a large number of malware samples in real time. However, The limitation of this work is the need for a detailed analysis on the hyper parameter tuning method and the robustness of the deep learning architectures is not discussed. Lima *et al*. (20) developed an antivirus program, using a flat artificial neural network, that can detect PE 1 file malware (Windows) with an average accuracy of 91.33%. They extracted 630 features from each executable file used as inputs of a flat artificial neural network. They used 6272 executables for validation and claimed to achieve an average performance of 98.32% in the distinction between benign and malware executables, accompanied by an average response time of only 0.07 s. Kumar *et al*. (21) have focused on carrying out the process of malware detection without having in-depth knowledge of malware and its analysis based on transfer learning for malware image classification procedures. Two Neural Networks were used; one was fully connected, and the other was a Recurrent Neural Network. They converted the window's portable executable files(PEs) into gray-scale images to serve as input to the customized deep CNN architecture for feature extraction. They claimed to achieve an accuracy of 98.92%, but their work did not cope with obfuscation.

The comparison in Table 1 shows that the majority of these approaches use a large dataset, which causes a long time to build the classification model. In addition, according to Alhanahnah *et al.* (22) a lot of effort and domain expertise are required in creating and identifying important features and uses time-consuming dynamic feature extraction which is why data preprocessing time is ignored in the reviewed papers which is essential especially when dynamic features are used. Furthermore, the majority of anti-virus programs analyze the executable files using approaches mostly relying on the specific behavior of malware for signature generation. Unfortunately, with minimal modifications, they would not be detected by the current methods.

METHODOLOGY

Malware Toolkit generates new malware that shares most of the code and small changes in their code (i.e., small changes in behavior), which is sufficient to evade the classical signature generation methods. We would like to generate a signature for the malware toolkit, which can be used for detecting all the generated samples, which is resilient to these changes (an invariant representation, similar to that used for computer vision). This section provides our novel approach for signature generation and detection. We first provide the details of the signature generator, then the chi-square predictor, and describe the malware dataset. The main questions we are trying to answer are the following:

Is it possible to generate a signature for a toolkit using a few programs created?

And if so, *Is it possible to develop a detection technique for detecting any new program created by that toolkit?*

Our method uses the AE-FT common features extraction approach introduced by Alkhateem and Megri (9) for extracting a representation of the footprints of a toolkit on the programs created by(i.e., Toolkit Signature). We create a deep belief network (DBN) by training a deep stack of denoising autoencoders, a fixed-target training strategy. When a DBN's training is complete, we discard the decoder layer, fix the values of the encoder layer, and use the encoder as a signature generator.

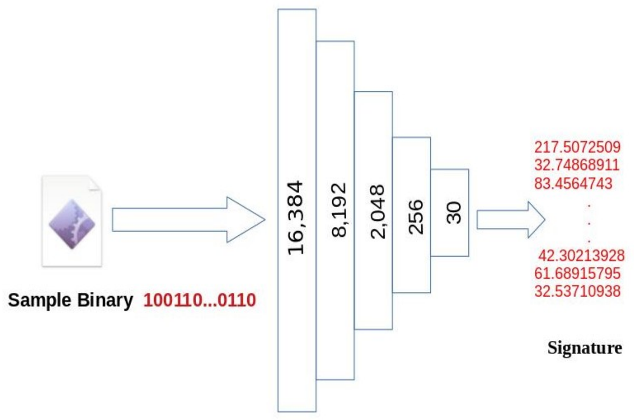

Fig. 1. Illustration the signature generating stages from feeding the program binary to signature generation using AE

The Chi-Squire Predictor: We use a statistical measure to differentiate between malware and benign programs. To detect malware, we measure the difference between the proportions of the common features in the toolkit signature and that of a test program using the Chi-square test as mentioned by Sokal *et al.* (23). The chi-square test is a maximum likelihood statistical significance test that measures the difference between proportions in two independent samples. The signature, S_t, for a toolkit specifies the weights of the features that a sample of malware created by that toolkit is expected to have.

To test the membership of a given test sample, its signature Ss is extracted and compared to S_t. The Chi-square is then computed as:

$\chi^2_i = (S_s i - S_t i)^2 S_t$ I ;$1 \leq i \leq n$

The value of χ^2_i is compared to a threshold value € from a standard Chi-square distribution table with (n − 1) degrees of freedom and significance level α (0.05, 0.01, 0.001). See Figure 2.

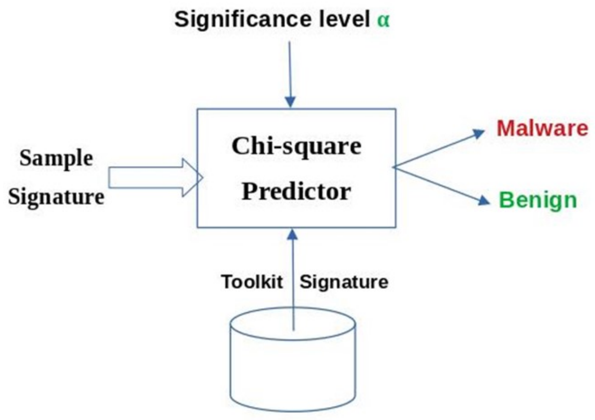

Fig. 2. Chi-Square Predictor

Degree of Membership μ: Let U = {χ^2i |χ^2i ≤ €}. We define the degree of membership μ_s as:

$\mu_s = |U|/n$

Degree of membership μ_s is a measure of the belongingness of test samples to a toolkit. The membership degree calculator algorithm is shown in Algorithm 1.

Algorithm 1. The Membership Degree Calculator

```
Input: Signature for a file $S_s = \{S s_i \;;1 \leqslant i \leqslant n\}$, signature for a toolkit,
  $S_t = \{S t_i \;;1 \leqslant i \leqslant n\}$ and $\alpha$;
Output: Degree of membership, $\mu_s$
Function Mu
    for $i$ = 1 to n and $S s_i \geqslant 5$ do
        $\chi^2_i = \frac{(S s_i - S t_i)^2}{S t_i}$ ;
    end
    $U = \{\chi^2_i \;|\chi^2_i \leqslant \alpha\}$; $\mu_s = \frac{\|U\|}{n}$;
    return $\mu_s$;
end
```

Degree of membership threshold μ_T.: Suppose we have N-training samples and Let μ_si be the degree of membership of sample S_i. Then we define the degree of membership threshold μ_T to be the minimum of the membership degrees of the training samples i.e.,

$\mu_T = \min(\mu_s i)$;$1 \leq i \leq N$

Detection Strategy: Let μ_T be the degree of membership threshold for a crimeware toolkit, and μ_s be the degree of membership for a program. Then, the program is classified as malware created by the toolkit when $\mu_s \geq \mu_T$. Putting the above steps together, we have constructed an end-to-end method for automatic signature generation, see Figure 3. The program binary bit-string is fed to the neural network, and the deep neural network generates a 30-sized vector at its output layer, which we treat as the signature of the program. The signature is fed to the Chi-Square predictor for class decision.

Description of Dataset: Our dataset contains 900 samples of Zeus Banking Trojan and 910 benign samples, for a total of 1810 samples. ZeuS, called "ZeuS, King of the Underground Crimeware Toolkits" as mentioned by Alazab *et al.* (6), is a crimeware toolkit that steals credentials from various online services like social networks, online banking accounts, ftp accounts, and email accounts.

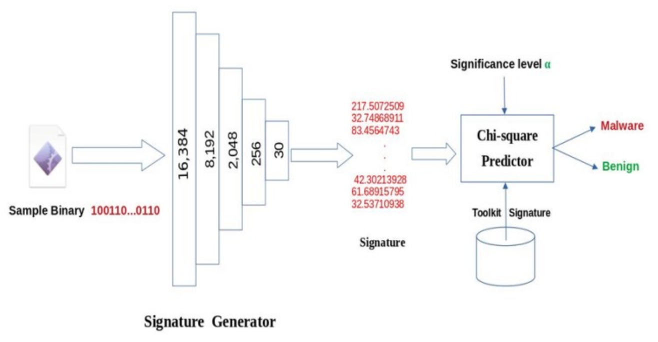

Fig. 3. Overview of our deep learning approach for Crimeware Signature generation and detection , illustration of all stages from feeding the sample binary to the signature generator ending with the decision made by the Chi-Square Predictor

The crimeware kit contains a tool to create the trojan binaries called exe builder. These samples were obtained from VirusTotal (https://www.virustotal.com/), VirusShare (https://virusshare.com/), and privately collected samples of benign and malware samples. The next section provides implementation details and experimental results and demonstrates that the resulting 30-sized vector (i.e., the signature) indeed provides a good representation of the toolkit footprints.

IMPLEMENTATION AND EXPERIMENTAL RESULTS

To test our proposed approach and specify the minimum number of training samples needed, we randomly split the malware samples into training and testing samples. For the first time, 2 samples for training and 898 samples for testing in addition to 910 benign testing samples. The second time, 5 samples for training and 895 samples for testing in addition to 910 benign testing samples. The third time, 7 samples for training and 893 samples for testing in addition to 900 benign testing samples. And in the last time, 10 samples were for training and 890 samples for testing in addition to 900 benign testing samples.

Training the Signature Generator: As described in the previous section, we train a deep denoising autoencoder consisting of five layers (16,384–8,192–2,048–256–30), with fixed-target training, we randomly pick one of the training samples as a pivot sample. Each time a new input is given to the network, we put the pivot sample in the output as the target. This way, we enforce the autoencoder to learn the common features of the training samples. Fixed-target training is essential for the network to learn even with few samples and in a very small time. We use rectified linear units (ReLU) for the non-linearity function when training deep neural networks to diminish the gradient vanishing problem and result in faster convergence as stated by Alkhateem and Megri (9). We build the model using machine learning algorithms in the Keras library. Other parameters we use are 50 training epochs, a learning rate of 0.003, batch size of 2 and 5, and no momentum.

Experimental Results: We now examine the efficiency of the signature generator, and the chi-square predictor, and specify the minimum number of training samples needed. To do so, we randomly split the malware samples into training and testing samples and use the benign sample for testing only, we did so four times, see Table 2. We train the DBN with the training sample and keep the decoder as the signature generator, as we mentioned earlier. We feed the pivot sample to the signature generator and use the generated signature as the toolkit signature. We feed all of our training samples of size 16,384-bit to the DBN that converts them to 30-sized representations (signature). Our model achieves 0.379 sec training samples preprocessing time on average, 0.002 sec average detection time (including sample preprocessing time), and 3.08 sec average model build time. To specify the minimum number of training samples needed, we experimented four times with different numbers of training samples as follows.

The First Experiment: In this experiment, we use 2 samples for training and 898 samples for testing in addition to 900 benign testing samples. The resulting accuracy is 79.0%. See Figure 4.

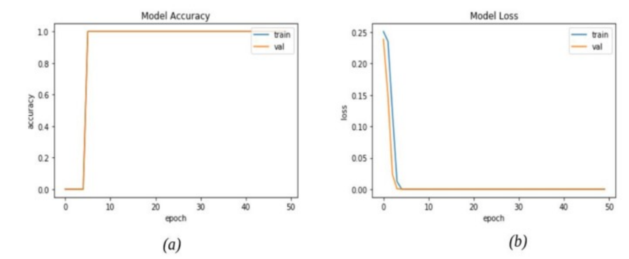

Fig. 4. Signature Generator 2-samples training and validation. (a)-Accuracy, (b)-Loss

The Second Experiment: In this experiment, we use 5 samples for training and 895 samples for testing in addition to 900 benign testing samples. The resulting accuracy is 89.0%. See Figure 5.

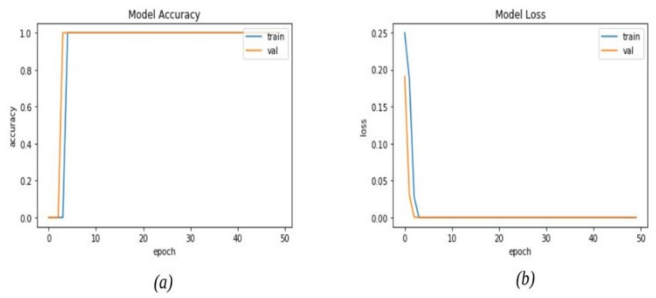

Fig. 5. Signature Generator 5-samples training and validation. (a)-Accuracy, (b)-Loss

The Third Experiment: In this experiment, we use 7 samples for training and 893 samples for testing in addition to 900 benign testing samples. The resulting accuracy is 94.0%. See Figure 6.

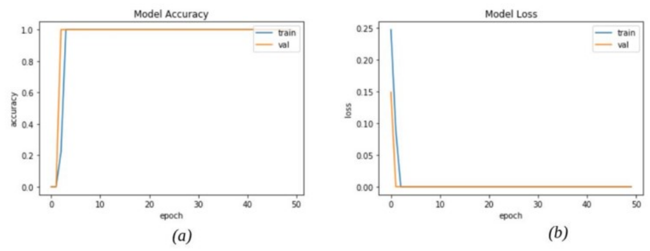

Fig. 6. Signature Generator 7-samples training and validation. (a)-Accuracy, (b)-Loss

The Last Experiment: In this experiment, we use 10 samples for training and 890 samples for testing in addition to 900 benign testing samples. The resulting accuracy is 97.0%. See Figure 7.

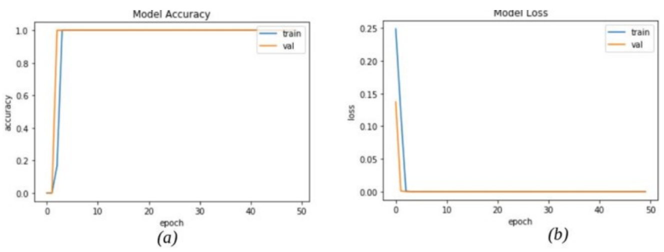

Fig. 7. Signature Generator 10-samples training and validation. (a)-Accuracy, (b)-Loss

We use the chi-square predictor to predict the correct class (malware or benign) on the test samples generated signatures. See Table 2. The high accuracy obtained attests to the efficiency of the signature generator and the Chi-square predictor, resulting in the successful detection of a high percentage of malware generated with the same toolkit, demonstrating that the signatures due to the proposed approach indeed capture the common features of malware and the efficiency of the Chi-square predictor.

Table 2. Experiments parameters and results

Experiment #	Training Samples	Test Samples		TP	FN	TN	FP	ACU
		Malware	Benign					
1	2	898	910	898	0	546	364	79%
2	5	895	910	776	119	819	91	89%
3	7	893	910	824	69	865	45	94%
4	10	890	910	890	0	865	45	97%

CONCLUDING REMARKS

In this paper, we reviewed past approaches for generating and detecting signatures for programs and proposed a novel method for toolkit signature generation and detection based on deep belief networks and chi-square for detecting zero-day malware created by a given toolkit. Current approaches for malware signature generation and detection use specific aspects of malware (e.g., certain network traffic normality or sub-string in the program) and need big training datasets, which make these methods ineffective against zero-day malware. Our proposed approach is inspired by the recent success in deep neural networks and the need to combine machine learning with statistical approaches to get better results in the security domain. We first trained a deep 5-layered neural network using a fixed-target training strategy on the training sample's bit-string which produces 30 values in its output layer. These values are used as the signature of the program. Then we used the chi-Square predictor to decide whether a sample is malware or benign. The experimental results show that the signatures produced by the DBN trained using the fixed-target strategy are highly successful for new malware detection using the Chi-square membership predictor. Using only 10 training samples, we achieved up to 97% new malware detection accuracy, 0.002 sec average detection time (including sample preprocessing time), and 3.08 sec average model build time. The results presented here demonstrate that the proposed signature generate strategy and toolkit signature are powerful methods that can be applied successfully to detect new malware during the outbreak time of a toolkit, whereas a few samples are available, which can shorten the toolkit lifetime significantly. Furthermore, we have found that our proposed approaches are feasible and their performance can match state-of-art approaches. Its training and classification time is among the smallest found and the accuracy when detecting malware.

REFERENCES

1. Dave Marcus (2014). The New Reality of Stealth Crimeware, https://www.intel.co.id/content/dam/www/public/us/en/documents/white-papers/new-reality-of-stealth-crimeware-paper.pdf.
2. Beek, C., Dunton, T., Fokker, J., Grobman, S., Hux, T., Polzer, T., ... & Sherstobitof, R. (2019). Mcafee labs threats report: August 2019. *McAfee Labs*.
3. McAfee (2013). Infographic : the state of malware, http://www.mcafee.com/in/security awareness/articles/state-of-malware-2013.aspx.
4. Lino, J., Torres, A., López, S., & Ramírez, S. (2015). Análisis de la reforma fiscal mexicana para el primer trimestre del 2014. *Desarrollo Gerencial*, *7*(2), 47-62.
5. Morgan, S. (2021). Cybercrime to cost the world $10.5 trillion annually by 2025. Cybercrime Magazine 13 (11).
6. Alazab, M., Venkatraman, S., Watters, P., Alazab, M., & Alazab, A. (2012). Cybercrime: the case of obfuscated malware. In Global Security, Safety and Sustainability & e-Democracy (pp.204-211). Springer Berlin Heidelberg.
7. Smith, S. (2015). Cybercrime will cost businesses over $2 trillion by 2019. *Retrieved from Juniper Research: https://www. juniperresearch. com/press/pressreleases/cybercrime-cost-businesses-over-2trillion*.
8. Atzeni, A., Díaz, F., Marcelli, A., Sánchez, A., Squillero, G., & Tonda, A. (2018). Countering android malware: A scalable semi-supervised approach for family-signature generation. *IEEE Access*, *6*, 59540-59556.
9. N. S. Alkhateem , Y., & Mejri , M. (2023). Auto Encoder Fixed-Target Training Features Extraction Approach for Binary Classification Problems . *Asian Journal of Research in Computer Science*, *15*(1), 32–43. https://doi.org/10.9734/ajrcos/2023/v15i1313
10. Filiol, E., & Josse, S. (2007). A statistical model for undecidable viral detection. *Journal in Computer Virology*, *3*(2), 65-74.
11. Tang, Y., & Chen, S. (2005, March). Defending against internet worms: A signature-based approach. In *Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies.* (Vol. 2, pp. 1384-1394). IEEE.
12. Park, S., Gondal, I., Kamruzzaman, J., & Oliver, J. (2019, February). Generative malware outbreak detection. In *2019 IEEE International Conference on Industrial Technology (ICIT)* (pp. 1149-1154). IEEE.
13. Rhode, M., Burnap, P., & Jones, K. (2018). Early-stage malware prediction using recurrent neural networks. *computers & security*, *77*, 578-594.
14. Gandotra, E., Bansal, D., & Sofat, S. (2016, December). Zero-day malware detection. In *2016 Sixth international symposium on embedded computing and system design (ISED)* (pp. 171-175). IEEE.
15. Khammas, B. (2018). Malware detection using sub-signatures and machine learning technique. *Journal of Information Security Research*, *9*(3), 96-106.
16. David, O. E., & Netanyahu, N. S. (2015, July). Deepsign: Deep learning for automatic malware signature generation and classification. In *2015 International Joint Conference on Neural Networks (IJCNN)* (pp. 1-8). IEEE.
17. Krizhevsky, A., & Hinton, G. E. (2011, April). Using very deep autoencoders for content-based image retrieval. In *ESANN* (Vol. 1, p. 2).
18. Sewak, M., Sahay, S. K., & Rathore, H. (2018, August). An investigation of a deep learning based malware detection system. In *Proceedings of the 13th International Conference on Availability, Reliability and Security* (pp. 1-5).
19. Vinayakumar, R., Alazab, M., Soman, K. P., Poornachandran, P., & Venkatraman, S. (2019). Robust intelligent malware detection using deep learning. *IEEE Access*, *7*, 46717-46738.
20. de Lima, S. M., Silva, H. K. D. L., Luz, J. H. D. S., Lima, H. J. D. N., Silva, S. L. D. P., de Andrade, A. B., & da Silva, A. M. (2021). Artificial intelligence-based antivirus in order to detect malware preventively. *Progress in Artificial Intelligence*, *10*(1), 1-22.
21. Kumar, S., & Janet, B. (2022). DTMIC: Deep transfer learning for malware image classification. *Journal of Information Security and Applications*, *64*, 103063.
22. Alhanahnah, M., Lin, Q., Yan, Q., Zhang, N., & Chen, Z. (2018, May). Efficient signature generation for classifying cross-architecture IoT malware. In *2018 IEEE conference on communications and network security (CNS)* (pp. 1-9). IEEE.
23. Sokal, R. R. (1995). The principles and practice of statistics in biological research. *Biometry*, 451-554.

****