European Journal of Molecular & Clinical Medicine
ISSN 2515-8260 Volume 07, Issue 03, 2020

A Survey of different machine learning models for static and dynamic malware detection

L.Srinivasa Reddy
Research Scholar, Department of Computer science and Engineering
Koneru Lakshmsiah Education Foundation,AP, India.
s_linga85@yahoo.com

Srikanth Vemuru
Professor, Department of Computer Engineering,
Koneru Lakshmsiah Education Foundation,AP, India.
vsrikanth@kluniversity.in

ABSTRACT: Malicious software (malware) plays a vital role in cybercrime security. As the number of malicious attacks and its target sources is increasing, it is difficult to find and prevent the attack due to its change in behaviour. Most of the traditional malware detection models are based on the statistical, analytical, and machine learning models. Detection of malware usually utilizes virus signature methods to defend against malicious software. Most antivirus tools to categorize malware depend on regular expression and pattern. Antivirus is less likely to update their databases to detect and prevent malware as file features have to update a newly created malware. The practically maximum human effort was required in order to generate attack signatures. In this paper, different types of malware detection models and their problems are discussed. This paper provides an extensive survey on the malware attack detection using traditional supervised, unsupervised models. Different types of malware attacks and their variations in behaviour are discussed in offline and online systems.

KEYWORDS: Malware detection, machine learning, statistical models.

1. INTRODUCTION

Today, the world moves towards a digital era in which the use of cyber technologies has become an integral part of everyday life. Computer and Internet use is not only limited to personal computing and access to information, but its role is also extended to everything through the use of technologies such as the Internet of Things (IoT), crypto-currency, etc. The world is currently talking to cyber colonies about the digital economy, such a deep involvement of computers and other technologies is bringing new challenges to the digital world. Every day, from e-banking to e-commerce, people and firms witness different types of cyberattacks on different cyberinfrastructures. The malware was coined by the combination of two words, Malicious and Software (Carlin,2019) [1]. It is software designed to disrupt damage or gain authorized access to a computer system, or any computer program is specifically written. The fraudulent acts include theft of information in credit cards, identity, destructive online action, the use of computer networks to gain unauthorized access to confidential information, and much more. With cybercrime's enormous growth rate dealing with potential danger, manually analyzing and understanding this vast sea of malware is simply very unreasonable. What helps analysts is that there is very few unique malware that the authors create by frequently reusing the patterns of code and code, thus creating new malware. The major drawback and difficulty that analysts can overwhelm is the malware action of

inheriting patterns and similarities between related patterns. To exploit the similarity and shared patterns in malware, the antimalware industry has started with the concept of machine learning, a field where computers are taught to discover and recognize the patterns inherited. They can quickly learn and find malware patterns and have an advantage over human analysis. Machine learning and analyzing malware are many areas and have much overlap in it.

The malware is detected by several tools and techniques whose signatures are available with antivirus and spyware removal software (tools) databases. This is a signature-based method in which each signature represents a pattern of code or unique identification extracted from the original malware. Once malicious code has been detected, analysis and dissection commence. The source code for malware is reversed by relying on the file for information. Some common methods of attack include phishing, Distributed Denial of Service (DDoS), identity theft, Ransomware, etc. Many of the modern cyber-attacks are carried out through the Malware's direct or indirect usage. The malicious program, commonly known as Malware, is a computer program that is intentionally developed for malicious activities such as attacking computer networks, system hijacks, deletion of files, stealing of information, spamming, and downloading of malware (Chen, 2018) [2]. The list of malicious activities is very extensive and grows at a rapid and regular rhythm with new entries. For example, Stuxnet added Critical-Infrastructure to the malicious list as its target (Cohen, 2018) [3]. There are two malware classes: the newly produced malware and malware variants (Cohen et al., 2016) [4]. Figure 1.1 shows the voluminous growth rate from 2015 through June 2019 for total malware and new malware.

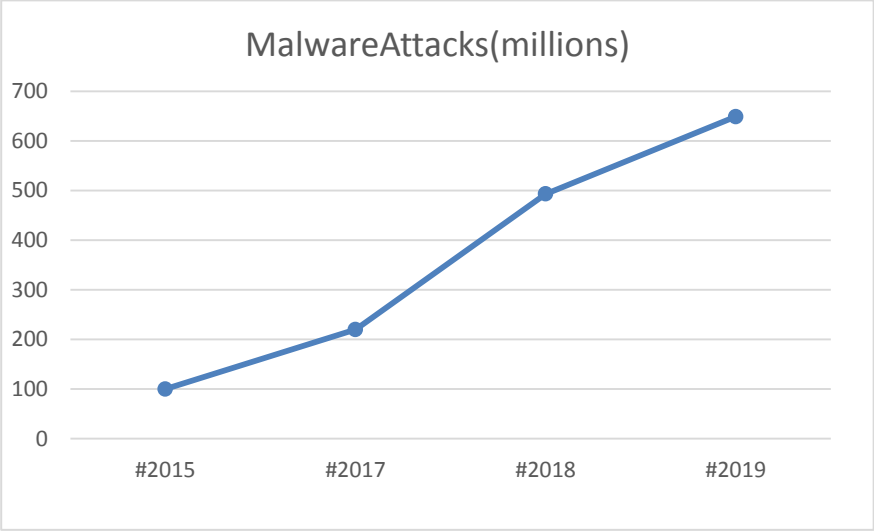

Figure 1: Malware growth rate from 2015 to 2019.

A critical challenge for the digital world is the exponential growth and sophistication of malware. Many security solutions, i.e., to control and minimize the loss caused by malware. Four techniques for anti-virus (AV) have been developed, and new approaches have been investigated[5]. In the past, researchers have grouped malware detection techniques in different ways; two main classes of malware detection techniques are Signature-based and Anomaly-based [6]. Detection based on anomaly is further branched to detection based on specifications. All the above-mentioned detection techniques are grouped into static, dynamic, and hybrid techniques based on the underlying analytical techniques. Detection based on a

signature uses the signature to detect malicious programs. Signature is a sequence of bytes that have been extracted from known malware before. Static signature-based detection is not running the program, while dynamic signature-based detection is running the program in a safe environment and checking for signature. Signature-based detection is also limited to the signature database requiring a regular update of newly created signatures, and the end-host storage of the signature database requires space proportional to the number of signatures.

Anomaly-based detection, a type of non-signature-based detection, can overcome the signature-based detection limitations. The detection based on anomalies does not use a specific signature for malware. A normal profile is developed for anomaly detection, and any deviation from the normal profile is treated as malicious. This work focuses on the detection of malware based on static analysis using machine learning algorithms that are a type of detection based on anomalies.

Cybersecurity data are mostly distorted and contain more positive data than malicious data. This leads to a few challenges during training and testing classifiers. First, traditional machine learning techniques in an imbalanced dataset often migrate towards the majority class[7]. Therefore, the actual performance of a model can not be expressed in standard metrics such as accuracy. In a distorted dataset that includes 95% of good examples and 5% of bad examples, the precision of 95% could be attributed to the classification system predicting neutral labels 100% of the time. In essence, the features are the core of the malware detector based on Machine Learning (ML)[8]. Analysis of binary code is based on the extraction of static features, and it is the only medium through which the content and structure of the program information are transmitted. These are the fundamentals, and so many applications use it to be interesting. The application includes binary matching, performance profiling, binary change, binary translation, debugging, performance modeling parameter extraction, computer security, and forensics. The advantage of dynamic analysis is that malware performs its designed-in functionality when it is started and when it is run. The data collected, such as memory allocation, files are written, registry read and written, and processes created during dynamic analysis is more useful than the data collected during static analysis. Dynamic runtime information can be used directly to evaluate potential damage malware that can cause new threats to be detected and classified. In other words, dynamic analysis can be used to test the context information at runtime, since most register or memory values can only be generated.

Problem statement: The main technique used for malware detection by antivirus programs[10] is a signature-based detection approach with a conventional classification of malicious and benign Windows PE32 executables. Usually, the binary classification is done with signatures, partial matching, regular expressions, or heuristics. This clustering approach is no longer sufficient, as, e.g., malware diversification that focuses on avoiding similarity-based malware matching by randomly diversifying code and data regions to reduce the similarity between malware mutants. Hence, the classification of malware needs refinement when it comes to detecting similar samples of malware functionality belonging to the same category. Besides, static signature-based malware detection is outdated and becomes less relevant with increasing malware threats each year. Multinomial detection and classification of malware based on dynamic memory, disk, and network compromise indicators, which could significantly improve anti-malware solutions, need to be explored and improved. Another challenge for the anti-malware infrastructure is the lack of agreement among anti-virus vendors about how to name the malware. For

example, uploading a hash value from the WannaCry Decryptor (MD5: 7bf2b57f2a205768755c07f238fb32cc) to the VirusTotal online scanning engine, which combines numerous anti-virus products, results in many different naming conventions for the various anti-virus software.

2. **LITERATURE REVIEW**

Static malware analysis is mostly used in anti-virus software for common malware detection. While this is a fast and easy way to identify malware, it is also quite unreliable. Malware authors often obscure or dynamically change their malware so that there is no longer enough signature-based or static analysis as a whole. This is demonstrated by Kim et al.[11] and their proposed malware diversification approach, which diversifies code and data regions at random. It reduces the similarity between the same malware instances to make direct, similarity-based matching worse or even disabled. This makes matching based on static similarity no longer effective. However, the subject of finding similarities between malware based on dynamic methods is dealt with in much recent research. Ye et al.[12]'s approach is to detect similarity-based malware by analyzing assembly instruction sequences in hard disk executables. Li et al.[13] propose an approach using DepSim to find weekly matches between malicious software based on control and data-dependence graphs obtained by identifying the maximum common subsection. Therefore DepSim uses techniques of dynamic taint analysis and backtracking. The experiment has shown that DepSim can find semantine similarities successfully, and can even deal with malware that is obfuscated or packed. Liu et al.[14] calculates the similarity level by analyzing function-call graphs based on the flooding algorithm for similarity graphs. Liu [15] detects malware using similarities in calls to an API. Additionally, clustering malware into both benign and malicious categories is also no longer sufficient. A new approach is to classify malware based on its functionality, which means the kind of family of malware they derive from training data. Liu et al.[15] proposes an approach to classifying malware and is based on the sequence characteristics of API calls. Another approach is based on dynamic API call counts in which Liu et al.[16] conduct a similarity analysis on the results of an investigation of a frequency API call. The authors use the Cuckoo Sandbox open-source tool to extract and align API calls based on the number of calls. Afterward, the API calls are classified into nine types based on the variant of malware. The ten APIs which have the highest call frequencies are used to define the type of malware. In another paper Wang et al.[17] presents a malware detection and classification approach based on an API call sequence alignment and visualization. The authors propose a system consisting of five functional steps: (1) collection of data and extraction of sequences; (2) extraction and-processing of features; (3) clustering; (4) extraction of behavioral sequences; and (5) detection and classification. 1790 malware samples and 1138 benign API call sequences were used to produce a 94.3 percent F-measure using the LCS (longest common subsequence) method of similarity. Ndichu et al.[18] presents an approach that is not based on static nor dynamic malware analysis. The malware is viewed as a binary pattern, re-shaped as a 2D matrix, then visualized as a picture. The authors state that visualizing the malware programs as images opens the path for a wider analysis of the spectrum. Zemmari [19] present an approach to classifying malware based on a similarity of content and directory structure. The authors capture malware in the extraction phase of the feature with the help of a honeypot system and a decompressor to extract file structure and content. They focus only on source code files during their work and will ignore all binary files. Finally, the malware will be clustered and classified into

a database within the Cluster system. The proposed system's accuracy is stated at 96.25%. Singh et al.[20] present an approach in which the Longest Common Subsequence between two traces of malware is calculated based on bigrams that appear. Two same variants of malware should share particular features or attributes of their family of malware. The authors can then make assumptions about which memory content of similar malware code was generated. Based on this technique of malware trace similarity, they built a clustering application on top of grouping similar samples of malware, and another application to find cases of reuse of code. They analyzed 16,248 malware samples in their system, producing an average precision value of 0.843 for the (static and dynamic) reference clustering sets. To extract malware features, Rabbani et al.[21] use Opcode n-gram, grey-scale images, and import function. In the decision-making phase, classifiers are trained with machine learning algorithms to group the suspicious samples of malware within the clustering phase into their corresponding family. The authors use Python's machine learning module SCikitlearn that contains the classification algorithms to classify malware. 23,740 malware samples from nine families are used for their experiment, achieving the best clustering accuracy of 0.853 malware when n is equal to three and all seven classifiers combined. The authors used 900 malware samples, 810 from known samples, and 90 from new ones when it came to detecting new malware. Of the 90 categories, 78 were correctly assigned, while 12 were wrongly assigned to different categories resulting in a precision of 0.867. Sahoo et al.[22] uses the Cuckoo Sandbox as a dynamic method for extracting API behavior data, grouping them using an n-gram model, and calculating the similarities to grouping the malware mutants in a database. The extracted API sequences are then compared using the cosine similitude method, and the malware codes are categorized into groups with the local cluster coefficient. The experiment showed the following results; as the threshold for similarity increases, the number of members in each group decreases, however, the accuracy of malware group members increases. For example, for the 95 percent threshold, 213 groups with 2065 members were created, which means that a total of 2639 analyzed malware samples could group 78.25 percent of malicious codes. Shekhawat et al.[23] propose an integrated method for classifying malware with static and dynamic features. To classify the malware, they use the HookMe trace tool to analyze API functions and a collection of machine learning algorithms called the Weka library. Wang et al.[30] are proposing an automatic malware detection system with string signatures, but the system fails with packed or metamorphosed malware. Their system is also limited if it can't generate good signatures that occur when the average number of variants in the malware family is too low. Mohamed and Ithnin[31] present major disadvantages of traditional signature-based malware detection systems based on data mining, machine learning, SVM, and API call graphing techniques. It lists the requirement of an up-to-date and maintained signature database, the impossibility of detecting new attacks, the so-called zero-day attacks, and the fact that simple techniques of obfuscation can evade detection. Yu et al.[25] allude to the limitations of visualization techniques for detecting and classifying malware. Malware binaries are visualized as gray-scale images, and the Euclidean distance uses a k-nearest neighbor approach. They state that the authors of malware might relocate sections or add redundant data to a binary to avoid detection. Also, machine learning techniques can easily produce many false positives, which diminish the confidence of users in machine-learning-based approaches[26]. Alqahtani et al.[27] employs a hybrid approach to improve classification accuracy by taking static and dynamic features into considerations. Dynamic features include files (create, edit, read, delete, memory-mapped), registries (create a key, delete key, monitor key, change the value, read a value, delete value), processes (create process, delete process, create a thread, read shared memory, write shared memory) and network features (TCP, SMTP, UDP, HTTP, FTP, ping requests, DNS queries, data). Alzaylaee et al.[28]

record traces of the execution of malware variants. From these traces of dynamic execution, API calls, return value[s], and modulename[s] are extracted as features, and their number of occurrences are stored in a trace frequency table. Badhani et al.[29] extract behavioral features from API system calls by using the trace tool ' HookMe' to collect run-time trace reports. They then use the Weka library's collection of machine learning algorithms to sort the malware out. For their approach to multinomial malware classification, Belouch [30] uses dynamic characteristics and machine-learning. They analyze disk activities with two low-level access sub-domains through the application, which includes modification, deletion, and writing to the file on disk storage and modifications to the registry. Besides, network traffic is analyzed as malware typically attempts to download payloads, communicate with the attacker, or upload sensitive network user information. Last, in the classification of multinomial malware, the authors mention memory footprints as a possible option for dynamic characteristics. They do not elaborate it any further, however, since the collection process hardly yields acceptable results and is considerably more cumbersome than the other two methods. Belouch et al.[39] are proposing a set approach for malware instructions in which they extract dynamic behavioral features as an instruction sequence. Thus, individual thread and process execution flows are sequentially attached to a single report for performing multinomial classification with machine learning techniques. For their multinomial malware classification method, the authors achieve an F-measure of over 0.96 percent.

Fang et al. [40] are proposing a multinomial malware classification system based on the malware's network behaviour. By extracting network flows from network traces collected in pcap files during malware execution, the malware is classified into their respective family. A conduct graph is generated to represent network activity, and dependence between network flows. Lastly, important features are extracted from the behavior graphs, such as graph size, root out-degree, average out-degree, maximum out-degree [and] number of specific nodes [33] to classify the malware with Weka library machine learning algorithms. Hashmi et al. [34] experiment on malware classification by their respective type. The authors use Cuckoo Sandbox to execute the malware and derive behavioral features from it, and the Weka library's Random Forest machine learning algorithm. Yen et al. presented their exploration of discriminatory features for classifying ML-based malware. For classification of malware families, they have experimented with various feature sets (byte-n-gram, opcode-n-gram, PE header fields, and dynamic trace). They presented their results to show which algorithm works best with which set of features and how many are optimal features. They have suggested with their study that Decision Tree (DT) performance is highest among all or equal to the performance of Support Vector Machine (SVM) for all features and provides maximum precision with minimum features [35].

Author	Method	Objective	Limitations
[36]	Random forest, naïve bayes	Prediction of malware using the three-phase classification algorithm	This model recorded a high false-positive rate on the training data.
[37]	Decision tree	Detecting malware scripts using the API system calls and user call requests.	Only applicable to known malware attack scripts.
[38]	SVM+Graph based algorithm for malware classification	In this paper, a graph-based SVM model is used to classify the malware attacks using the training dataset.	Unable to find the essential patterns for malware detection.

[39]	Artificial neural network	In this work, a feature selection based classification model is proposed on the real-time limited dataset.	Difficult to classify the patterns on the large real-time data.
[40]	Hierarchical clustering with a decision tree model	In this paper, a hybrid hierarchical clustering with the decision tree approach is proposed on the small malware dataset.	Difficult to detect multi-class attacks on real-time malware dataset.
[41]	Feature selection based random forest model	In this paper, an advanced random forest model is applied to the unknown malware detection on limited databases.	In the future , this work can be extended to imbalanced and large datasets.
[42]	Implemented different malware attack detection models on the training dataset.	In this paper, different classification algorithms are designed and tested on multiple learning models for statistical analysis.	In this work, aata imbalance problem affects the true positive rate and error rate.
[43]	Implemented a new Monte Carlo tree search model for predicting the active malware in the training dataset.	In this paper, a new search model is designed on the active malware attacks to predict the new type of malware as the test data.	Problem to predict the large number of active malware using the search model.
[44]	Implemented a new machine learning-based visualization approach on the malware dataset.	In this paper, data visualization and adversarial learning models are proposed on the malware detection process.	In future work, an advanced adversarial learning model is proposed on the large training malware databases.
[45]	Implemented a new malware classification algorithm using word2vec and deep learning framework on the training dataset.	In this paper, a novel deep learning framework is proposed on the malware training data for the classification problem.	In the future, fasttext and Bi-LSTM models are used to improve the performance of the proposed model.

3. Conclusion

Current malicious code detection systems are based mostly on traditional syntactic signature methods, which specify byte sequences that are characteristic of specific machine instances.

This approach has its limitation in identifying and generating a signature that is a time-consuming process, so between the appearance of a new malicious code instance and the availability of a signature that can detect it, there is always a vulnerability window. Decision trees use a pre-classified dataset to learn how to categorize data based on current patterns and trends. After the tree has been created, the decision tree logic can be incorporated into several different malware detection technologies including firewalls and signatures based on the host. In this paper, different types of malware detection models and their problems are discussed. This paper provides an extensive survey on the malware attack detection

using traditional supervised, unsupervised models. Different types of malware attacks and their variations in behaviour are discussed in offline and online systems.

References

[1] D. Carlin, P. O'Kane, and S. Sezer, "A cost analysis of machine learning using dynamic runtime opcodes for malware detection," Computers & Security, vol. 85, pp. 138–155, Aug. 2019, doi: 10.1016/j.cose.2019.04.018.
[2] Z. Chen et al., "Machine learning based mobile malware detection using highly imbalanced network traffic," Information Sciences, vol. 433–434, pp. 346–364, Apr. 2018, doi: 10.1016/j.ins.2017.04.044.
[3] A. Cohen, N. Nissim, and Y. Elovici, "Novel set of general descriptive features for enhanced detection of malicious emails using machine learning methods," Expert Systems with Applications, vol. 110, pp. 143–169, Nov. 2018, doi: 10.1016/j.eswa.2018.05.031.
[4] A. Cohen, N. Nissim, L. Rokach, and Y. Elovici, "SFEM: Structural feature extraction methodology for the detection of malicious office documents using machine learning methods," Expert Systems with Applications, vol. 63, pp. 324–343, Nov. 2016, doi: 10.1016/j.eswa.2016.07.010.
[5] Z. Cui, L. Du, P. Wang, X. Cai, and W. Zhang, "Malicious code detection based on CNNs and multi-objective algorithm," Journal of Parallel and Distributed Computing, vol. 129, pp. 50–58, Jul. 2019, doi: 10.1016/j.jpdc.2019.03.010.
[6] S. Das Bhattacharjee, W. J. Tolone, and V. S. Paranjape, "Identifying malicious social media contents using multi-view Context-Aware active learning," Future Generation Computer Systems, vol. 100, pp. 365–379, Nov. 2019, doi: 10.1016/j.future.2019.03.015.
[7] Y. Fang, C. Huang, Y. Su, and Y. Qiu, "Detecting Malicious JavaScript Code Based on Semantic Analysis," Computers & Security, p. 101764, Feb. 2020, doi: 10.1016/j.cose.2020.101764.
[8] D. Gibert, C. Mateu, and J. Planes, "The rise of machine learning for detection and classification of malware: Research developments, trends and challenges," Journal of Network and Computer Applications, vol. 153, p. 102526, Mar. 2020, doi: 10.1016/j.jnca.2019.102526.
[9] M. Hasan, Md. M. Islam, M. I. I. Zarif, and M. M. A. Hashem, "Attack and anomaly detection in IoT sensors in IoT sites using machine learning approaches," Internet of Things, vol. 7, p. 100059, Sep. 2019, doi: 10.1016/j.iot.2019.100059.
[10] N. Hosseini, F. Fakhar, B. Kiani, and S. Eslami, "Enhancing the security of patients' portals and websites by detecting malicious web crawlers using machine learning techniques," International Journal of Medical Informatics, vol. 132, p. 103976, Dec. 2019, doi: 10.1016/j.ijmedinf.2019.103976.
[11] S. Kim, J. Kim, S. Nam, and D. Kim, "WebMon: ML- and YARA-based malicious webpage detection," Computer Networks, vol. 137, pp. 119–131, Jun. 2018, doi: 10.1016/j.comnet.2018.03.006.
[12] B. Li, R. Ye, G. Gu, R. Liang, W. Liu, and K. Cai, "A detection mechanism on malicious nodes in IoT," Computer Communications, vol. 151, pp. 51–59, Feb. 2020, doi: 10.1016/j.comcom.2019.12.037.
[13] T. Li, G. Kou, and Y. Peng, "Improving malicious URLs detection via feature engineering: Linear and nonlinear space transformation methods," Information Systems, vol. 91, p. 101494, Jul. 2020, doi: 10.1016/j.is.2020.101494.
[14] L. Liu, Z. Ma, and W. Meng, "Detection of multiple-mix-attack malicious nodes using perceptron-based trust in IoT networks," Future Generation Computer Systems, vol. 101, pp. 865–879, Dec. 2019, doi: 10.1016/j.future.2019.07.021.

[15] L. Liu, J. Yang, and W. Meng, "Detecting malicious nodes via gradient descent and support vector machine in Internet of Things," Computers & Electrical Engineering, vol. 77, pp. 339–353, Jul. 2019, doi: 10.1016/j.compeleceng.2019.06.013.
[16] L. Liu, X. He, L. Liu, L. Qing, Y. Fang, and J. Liu, "Capturing the symptoms of malicious code in electronic documents by file's entropy signal combined with machine learning," Applied Soft Computing, vol. 82, p. 105598, Sep. 2019, doi: 10.1016/j.asoc.2019.105598.
[17] L. Lv, W. Wang, Z. Zhang, and X. Liu, "A novel intrusion detection system based on an optimal hybrid kernel extreme learning machine," Knowledge-Based Systems, p. 105648, Feb. 2020, doi: 10.1016/j.knosys.2020.105648.
[18] S. Ndichu, S. Kim, S. Ozawa, T. Misu, and K. Makishima, "A machine learning approach to detection of JavaScript-based attacks using AST features and paragraph vectors," Applied Soft Computing, vol. 84, p. 105721, Nov. 2019, doi: 10.1016/j.asoc.2019.105721.
[19] V. P., A. Zemmari, and M. Conti, "A machine learning based approach to detect malicious android apps using discriminant system calls," Future Generation Computer Systems, vol. 94, pp. 333–350, May 2019, doi: 10.1016/j.future.2018.11.021.
[20] J. Singh and J. Singh, "Detection of malicious software by analyzing the behavioral artifacts using machine learning algorithms," Information and Software Technology, vol. 121, p. 106273, May 2020, doi: 10.1016/j.infsof.2020.106273.
[21] M. Rabbani, Y. L. Wang, R. Khoshkangini, H. Jelodar, R. Zhao, and P. Hu, "A hybrid machine learning approach for malicious behaviour detection and recognition in cloud computing," Journal of Network and Computer Applications, vol. 151, p. 102507, Feb. 2020, doi: 10.1016/j.jnca.2019.102507.
[22] S. R. Sahoo and B. B. Gupta, "Hybrid approach for detection of malicious profiles in twitter," Computers & Electrical Engineering, vol. 76, pp. 65–81, Jun. 2019, doi: 10.1016/j.compeleceng.2019.03.003.
[23] A. S. Shekhawat, F. D. Troia, and M. Stamp, "Feature analysis of encrypted malicious traffic," Expert Systems with Applications, vol. 125, pp. 130–141, Jul. 2019, doi: 10.1016/j.eswa.2019.01.064.
[24] R. Wang, Y. Zhu, J. Tan, and B. Zhou, "Detection of malicious web pages based on hybrid analysis," Journal of Information Security and Applications, vol. 35, pp. 68–74, Aug. 2017, doi: 10.1016/j.jisa.2017.05.008.
[25] M. Yu, "Malicious documents detection for business process management based on multi-layer abstract model," Future Generation Computer Systems, vol. 99, pp. 517–526, Oct. 2019, doi: 10.1016/j.future.2019.04.012.
[26] M. Alauthman, N. Aslam, M. Al-kasassbeh, S. Khan, A. Al-Qerem, and K.-K. Raymond Choo, "An efficient reinforcement learning-based Botnet detection approach," Journal of Network and Computer Applications, vol. 150, p. 102479, Jan. 2020, doi: 10.1016/j.jnca.2019.102479.
[27] F. H. Alqahtani and F. A. Alsulaiman, "Is image-based CAPTCHA secure against attacks based on machine learning? An experimental study," Computers & Security, vol. 88, p. 101635, Jan. 2020, doi: 10.1016/j.cose.2019.101635.
[28] M. K. Alzaylaee, S. Y. Yerima, and S. Sezer, "DL-Droid: Deep learning based android malware detection using real devices," Computers & Security, vol. 89, p. 101663, Feb. 2020, doi: 10.1016/j.cose.2019.101663.
[29] S. Badhani and S. K. Muttoo, "CENDroid—A cluster-ensemble classifier for detecting malicious Android applications," Computers & Security, vol. 85, pp. 25–40, Aug. 2019, doi: 10.1016/j.cose.2019.04.004.
[30] M. Belouch, S. El Hadaj, and M. Idhammad, "Performance evaluation of intrusion detection based on machine learning using Apache Spark," Procedia Computer Science, vol. 127, pp. 1–6, Jan. 2018, doi: 10.1016/j.procs.2018.01.091.

[31] M. G. C. A. Cimino, N. De Francesco, F. Mercaldo, A. Santone, and G. Vaglini, "Model checking for malicious family detection and phylogenetic analysis in mobile environment," Computers & Security, vol. 90, p. 101691, Mar. 2020, doi: 10.1016/j.cose.2019.101691.
[32] W. Fang, X. Tan, and D. Wilbur, "Application of intrusion detection technology in network safety based on machine learning," Safety Science, vol. 124, p. 104604, Apr. 2020, doi: 10.1016/j.ssci.2020.104604.
[33] R. Gupta, S. Tanwar, S. Tyagi, and N. Kumar, "Machine Learning Models for Secure Data Analytics: A taxonomy and threat model," Computer Communications, vol. 153, pp. 406–440, Mar. 2020, doi: 10.1016/j.comcom.2020.02.008.
[34] A. S. Hashmi and T. Ahmad, "GP-ELM-RNN: Garson-pruned extreme learning machine based replicator neural network for anomaly detection," Journal of King Saud University - Computer and Information Sciences, Sep. 2019, doi: 10.1016/j.jksuci.2019.09.007.
[35] M. Jerbi, Z. C. Dagdia, S. Bechikh, M. Makhlouf, and L. B. Said, "On the use of artificial malicious patterns for android malware detection," Computers & Security, vol. 92, p. 101743, May 2020, doi: 10.1016/j.cose.2020.101743.
[36] E. B. Karbab and M. Debbabi, "MalDy: Portable, data-driven malware detection using natural language processing and machine learning techniques on behavioral analysis reports," Digital Investigation, vol. 28, pp. S77–S87, Apr. 2019, doi: 10.1016/j.diin.2019.01.017.
[37] H. B. Kazemian and S. Ahmed, "Comparisons of machine learning techniques for detecting malicious webpages," Expert Systems with Applications, vol. 42, no. 3, pp. 1166–1177, Feb. 2015, doi: 10.1016/j.eswa.2014.08.046.
[38] A. K. M.a. and J. C.d., "Automated multi-level malware detection system based on reconstructed semantic view of executables using machine learning techniques at VMM," Future Generation Computer Systems, vol. 79, pp. 431–446, Feb. 2018, doi: 10.1016/j.future.2017.06.002.
[39] N. Pitropakis, E. Panaousis, T. Giannetsos, E. Anastasiadis, and G. Loukas, "A taxonomy and survey of attacks against machine learning," Computer Science Review, vol. 34, p. 100199, Nov. 2019, doi: 10.1016/j.cosrev.2019.100199.
[40] Z. Ren, H. Wu, Q. Ning, I. Hussain, and B. Chen, "End-to-end malware detection for android IoT devices using deep learning," Ad Hoc Networks, vol. 101, p. 102098, Apr. 2020, doi: 10.1016/j.adhoc.2020.102098.
[41] J. Selvi, R. J. Rodríguez, and E. Soria-Olivas, "Detection of algorithmically generated malicious domain names using masked N-grams," Expert Systems with Applications, vol. 124, pp. 156–163, Jun. 2019, doi: 10.1016/j.eswa.2019.01.050.
[42] M. Aghaeikheirabady, "A New Approach to Malware Detection by Comparative Analysis of Data Structures in a Memory Image," no. Ictck, pp. 26–27, 2014.
[43] S. S. Hansen, T. Mark, T. Larsen, M. Stevanovic, and J. M. Pedersen, "An Approach for Detection and Family Classification of Malware Based on Behavioral Analysis," 2016.
[44] B. Kolosnjaji, G. Eraisha, G. Webster, A. Zarras, and C. Eckert, "Empowering convolutional networks for malware classification and analysis," Proc. Int. Jt. Conf. Neural Networks, vol. 2017-May, pp. 3838–3845, 2017.
[45] M. Chandramohan, H. K. Tan, L. C. Briand, and L. K. Shar, "A Scalable Approach for Malware Detection through Bounded Feature Space Behavior Modeling," pp. 312–322, 2013.