Grenze International Journal of Engineering and Technology, June Issue

Comprehensive Review on Malware Analysis and Detection Technique

Parvathi S J[1] and Yogeesh A C[2]
[1]Assistant Professor, Dept. of ISE, JSS Science and Technology University, Mysuru, Karnataka, India
Email: sjparvathi@jssstuniv.in
[2]Assistant Professor Dept. of CSE, Government Engineering College, Chamarajanagara,Karnataka, India
Email: yogeesh13@gmail.com

***Abstract*— Recent research indicates that the quantity of potentially dangerous software, or malware, is growing at an alarming rate. Malware can use a variety of techniques to hide its existence from the system. Device networks and internet connections must be protected from malware in order to stop it from infecting many machines. Many research on malware detection techniques have been conducted in the past few years. Malware detection is still challenging, though. When it comes to detecting malware that has previously been found, heuristic and signature-based methods are equally efficient and quick; however, signature-based methods have not shown to be very useful for recognising malware that has not yet been found. While there isn't a method that can identify every potential virus variation, deep learning algorithms can identify certain viruses, both new and old. However, behavior-based approaches are more successful in treating infections that are difficult to cure. This demonstrates how hard it is to design a viral screening programme that works and how much opportunity there is for creative ideas and methods. This paper provides a thorough examination of malware identification methods, encompassing more contemporary approaches that make use of tried-and-true methods.**

***Index Terms*— Malware features, malware detection techniques, cyber security and malware classification**

I. INTRODUCTION

Using a personal computer or a smartphone for using the internet has grown into a daily need due to the recent spike in digitalization and improvements in smart technology. Because of this level of interconnectedness, there are numerous security and privacy risks, as well as a range of illegal activities. For example, in June 2021, information belonging to 700 million people on the corporate networking website LinkedIn was leaked over a dark web forum, which was affecting about 90% of its subscribers. Several Facebook app information were released to the public online in April 2019.According to www.csoonline.com, the information relates to over 530 million Facebook users and include telephone numbers, login names, and Facebook IDs. Fear around the Coronavirus (COVID-19) has been exploited by cybercriminals extensively. University Hospital. The information like medical results was listed for sale on eBay. When malware first emerged, it was simple to detect because it was created with clear purposes. Malware classified as conventional (basic) malware falls under this category. On the other hand, malware that can operate in the kernel, is more hazardous, and is more difficult to detect than traditional malware is referred to as modern malware. Table 1 compares malware from the past and current generations.

Grenze ID: 01.GIJET.10.2.746

TABLE 1: MALWARE FROM OLDER AND PRESENT GENERATIONS COMPARED

Parameter for comparison	Conventional malware(old)	Current generation malware(present)
Level of Implementation	Usually, standalone executable files or scripts are used for implementation.	makes use of complex methods such as polymorphic, memory-resident, and fileless code
Status of Actions	Most actions are static and have limited versatility	Intelligent decision-making and self-modification are examples of dynamic behaviours with adaptive capacities.
Proliferation	It is primarily spread by removable media, infected files, and email attachments.	Distributes through a variety of channels, such as supply chain attacks, drive-by downloads, social media, and exploit kits.
Attack Type	Common varieties comprise ransomware, Trojan horses, worms, and viruses.	Sophisticated attack types include AI-powered attacks, polymorphic malware, ,zero-day exploits, and fileless malware.
The Defensive Challenge	comparatively simpler to identify and address with conventional security tools	poses severe challenges because of sophisticated evasion strategies, encryption, and anti-analysis measures

Cyberattacks are becoming more frequent and more severe, affecting not just public and private organisations but also individuals, governments, and other organisations. According to Check Point Software report, ransomware is the number one threat from cyberattacks, which are now state-level weapons disrupting daily life. And when compared to 2020, cyberattacks on business networks increased by 50% in 2021. A new global analysis from McAfee titled "The Hidden Costs of Cybercrime" highlights the huge financial and covert effects that cybercrime has on the whole of society. The report, which was produced in association with the (CSIS), estimates that the cost of cybercrime to the global economy seems to be more than $1 trillion, or more than one percent of the global GDP. This is a rise in more than 50% from a 2018 study that estimated that the price of cybercrime worldwide was close to $600 billion. Beyond the global statistic, the survey also looked into harm reported that went beyond monetary losses and discovered 92 percent of businesses had repercussions that went beyond monetary losses. This indicates the difficulty of developing an efficient malware detection tool and the great need for further research and techniques. The present status of malware detection techniques is examined by this study through a review of the literature.

II. DEFINITION OF A PROBLEM

This section looks at the likelihood of finding malware. From this, we may conclude that developing an algorithm to recognise every kind of infection is impossible. Malware detection is NP-complete, which leads to this.Since malicious software programmers employ complex strategies like obfuscation to render the procedure very difficult, malware detection is still a tricky operation in real life while being a difficult operation in principle.The finding of viruses was crucial to theoretical study since it was the first hazardous programming discovered in the field. Initial investigations reveal that virus identification is difficult [3] to [5] and NP-complete [6]–[9]. It is challenging to ascertain whether a virus has been discovered, according to F. Cohen, because computer virus identification techniques rely on discrepancies [3][5] [6]. If D, the decision-maker, views the detection problem as a difficulty involving decision-making, then D must determine the likelihood that P is a virus. According to Cohen, it is hard to tell if P is a virus because if it were, D would have identified it as such and prevented it from altering other programmes or acted in a way that would suggest it was a virus. In the unlikely event that D's decision-maker fails to recognise P as a virus, P would collaborate with other apps to propagate and infect individuals. It is challenging to identify P given the steps involved in this decision-making process.

III. IMPLICATIONS OF THE PROBLEM'S DIFFICULTY

The antiviral industry has faced several difficulties. Because creating malicious software aims to hide its existence. Depending on how it conceals its actual identity, malicious software may be classified into several categories, such as encrypted, polymorphic, metamorphic, and oligomorphic software. In the world of cyber security, it is well recognised that developing a method that can conclusively detect each piece of malware is challenging. There are several reasons for these limitations, including:

*Encryption***:** Malware uses encryption to hide a harmful code block throughout the entirety of its code [10]. As a result, malware hides inside the host.

*Oligomorphic***:** The malware payloads is encrypted and deciphered using a separate key in the oligomorphic technique [11]. Thus, malware that use oligomorphic technique rather than encryption is more difficult to find.
*Polymorphic***:** The phrase "polymorphic" describes the ability of malware to change into multiple forms; the Greek terms "poly" and "morph" mean "many" and "form," respectively. Malware that uses the polymorphism methodology uses a distinct key for encryption as well as decryption, just as much as the oligomorphic method [12]. The payload component may perform layer-by-layer encryption and contains numerous copies of the decoder [13]. As a result, polymorphic malware is more difficult to identify than oligomorphic malware.
*Malware that is unknown***:** Since malware continually evolving, new varieties are often produced by attackers. These malware strains, which can also be referred to as "zero-day" or "unknown," don't exhibit any recognisable signs and have never been observed before. If an algorithm has no past knowledge, it might be challenging for it to identify something it hadn't seen before.
*Metamorphic***:** There is no encryption used in the metamorphic method. Instead, during execution, the malicious activity employs dynamic code obfuscation, in which the opcode is modified each time [14]. The signature of this kind of virus is incredibly cryptic, and it is quite difficult to identify each new clone.
*Stealth***:** Also known as code protection, this tactic employs many defences to keep it from being fully investigated [11]. For instance, it has the ability to modify the system without drawing the notice of detection systems. This section explains the limitations of malware detection techniques. According to recent research, there is no way to create a technique that can accurately identify every kind of illness. It is possible to greatly increase detection rates by combining many detection techniques, such as behavioural analysis, signature-based, machine learning, heuristic, and human expertise (threat hunting). This multi-layered protective technique offers a more thorough and dependable protection against malware assaults while also mitigating some of the shortcomings of individual techniques.

IV. Techniques And Methods For Detecting Malware

A program's content has to be investigated to see if it contains malware. It makes use of malware detecting technologies. The three steps in the process are feature extraction, classification, and malware analysis. In recent years, data mining, machine learning, and deep learning have all been used often to detect the presence of malware.
A) Malware analysis: Malware must be examined in order to comprehend its behaviours and contents. The process of determining how malware operates and providing answers to the following queries is known as malware analysis [17], [18]. What information is deleted or destroyed, how malware operates, which devices and applications are impacted, etc. The two primary techniques for analysing malware are static and dynamic approaches [17]. Without running the code, static analysis investigates the malware [19]. However, dynamic analysis looks at how the virus moves while it executes its instructions. Advanced dynamic analysis comes after basic static analysis when it comes to malware analysis.
B) Method for obtaining malware features: Data mining techniques are used to extract malware characteristics. Finding important and previously unknown information inside enormous data sets or databases is known as data mining. Datamining has led to the development of novel approaches and datasets recently [21].The n-gram and graph frameworks are two of the approaches available for building malware datasets and properties.
*1) n-gram Model***:** A popular statistical model of language in the domain of natural language processing, which is used for tasks like text synthesis, machine translation, and speech recognition is the n-gram model. However, it might also be changed to remove undesirable characteristics. During malware research, binary files or API call patterns could be subjected to n-grams in order to extract characteristics for categorisation or anomaly detection.
*2) Graph-based Model***:** Graph-based methods for identifying malware make use of the connections and exchanges that naturally occur between different entities—like files, functions, system calls, or API requests—that are recorded in the resulting graph. Apart from offering easily understood viewpoints regarding the order and connections between risky behaviours, these models are also highly effective in spotting unusual trends that indicate the presence of malicious software. Because they consider the contextual and temporal features of these links, graph-based models are a valuable tool in the ongoing efforts to improve cybersecurity measures. This assists in comprehending the behaviour of malware dynamics.
3) Malware Dataset: There aren't many publicly accessible datasets that have been approved as reliable sources for malware detection, similar to other study fields. Additionally, the majority of the datasets that are now accessible are not suitable for research uses, and those that are frequently have inadequate formatting that makes them ineffective for use with data mining and machine learning techniques. A few of the datasets utilised in malware analysis are the AAGM, EMBER, Microsoft malware category challenge, ClaMP (classification of malware using PE headers), and NSL-KDD libraries.

C) Malware Classification: Without explicit coding, machine learning (ML), a collection of algorithms, can accurately predict application results. Machine learning attempts to convert the input that is given data into suitable intervals using statistical analysis. Numerous related data processing tasks, such as clustering, regression, and classification, can be handled using machine learning. Machine learning approaches have been used for a long time in malware detection [28]. Some well-known machine learning algorithms are the Bayesian network , naive Bayes , logistic model trees , random forest tree, KNN, multilayer perceptron, SVM, sequential minimal optimisation (SMO), and simple logistic regression. These algorithms perform especially well in conjunction with particular other detection methods, including behavior-based detection. While each algorithm has benefits and drawbacks, it is challenging to determine which approach is better than the others. However, one strategy might work better than another in terms given the quantity of attributes, data distribution, and property correlations.

V. Approaches For Malware Detection

The amount of scholarly research on malware detection has increased dramatically in the last several years. In the past, a lot of individuals used signature-based detection techniques. This approach performs poorly when handling zero-day malware, even though it is quick and effective when handling known malware [21], [29]. In addition to cutting edge techniques like deep learning-based detection, researchers have traditionally used behaviour and heuristic-based detection methods. An overview of the traits, methods, and strategies applied in malware detection is provided in Figure 1. Before a further explanation, Table 2 provides a synopsis of several well-known strategies employed in each detection technique along with links to pertinent literature.

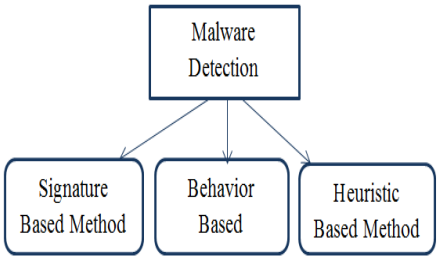

Figure 1: Malware Detection Methods

Table II. An Overview Of Recent Research On Methods For Detecting Malware.

Paper	Method	Objective
[30]	system calls for printable strings, DLL and API	finding new malware
[31]	assembly guidelines, cosinc resemblance	Determine the various types of malware
[32]	File hash vector transformation	Detect several viruses with a single signature
[33]	file header data, names of DLLs and APIs	detection of recognized malware with a 99% success rate
[34]	obtaining parameters using hybrid-based text mining method	may identify malware kinds with little overhead
[35]	instructions for conditional branching	Determine different paths of execution
[36]	phylogenetic tree, hellinger distance, and system calls	recognize current malware and various types of malware
[37]	constructing diagrams for system calls	distinguish between several malware type
[38]	system call frequency, n-gram	recognize fresh malware and various forms of malware
[39]	executive history, nonsparse matrix, and XML file	distinguish between fresh and ancient malware
[40]	The sequences of n gram bytes	distinguish between common and unique forms of malware
[41]	Byte sequences	Identifying different malware variants
[42]	Markov chain, n-gram and diagram	Identifying different malware variants with 96.41%
[43]	Printable strings, frequencies of API methods	recognizing novel and distinct malware types with 97%
[44]	Diagram showing the relationships and system calls	Identify an attack using code insertion

A. Signature-based malware Detection

The usage of signatures is a popular method for detecting malware. Known viruses can be recognised by certain characteristics or signatures. The antivirus programme uses a sizable library of malware signatures to examine the binary patterns it discovers during files or system scans. If a suitable match is found, the application sounds a message and initiates the appropriate action.

Related Work on Signature Based Detection

Yong Tang et al. [64] proposed a bioinformatics technique that reliably generates exploit-based IDs for polymorphic worms. There are three phases to this approach: When noise is reduced after the first round of noise reduction, the reduced regular expression a signature is compared to the existing IDSs by signature transformation. Multiple substring extractions lead to multiple sequence alignment. The proposed schema, according to the authors, is more precise and noise-tolerant than some of the exploit-based signature generating schemas that are already in use. This is due to the fact that it extracts several polymorphic worm characteristics, such as distance restrictions between invariant bytes and byte-by-byte invariants. The paradigm put out here, however, is unique to polymorphic worms & is not applicable to other kinds of harmful software. Abadi and Borojerdi created the novel MalHunter identification approach, which is based on sequence alignment and classification [65]. Based on how it functions, the infection automatically generates polymorphic malware signatures. The novel strategy works as follows: Initially, several malware strains are used to generate behaviour sequences. Subsequently, many groupings are established and modified inside the database by employing similar behavioural patterns as a basis. To identify malware samples, behaviour sequences are gathered and compared to sequences that have already been developed and saved in databases. The comparison determines whether or not to identify the sample as harmless. They were successful in achieving an FPR of 0.80% and an accuracy of recognising 90.83% with a cluster radius of 0.4 & a similarity threshold of 0.05, according to the test results. The authors suggest employing this schema to broadly detect any polymorphic virus that is immune to obfuscation tactics. Moreover, it is not limited to a certain type of virus. Additionally, the suggested approach outperformed cutting-edge methods for producing signatures, such as those Tang et al. had previously shown in a study [64]. Newsome et al. [66] and Perdisci et al. [67]. The recommended method can only be used to combat polymorphic malware, and a small sample of malware samples was used to assess how successful it was. In [41], the process for creating automatic string signatures (Hancock) is explained. The study indicates that the optimal string signatures with the greatest viral spread might be automatically created using the proposed schema and the smallest number of false positives.The proposed method ensures that every context in which a signature appears has files containing malware that are similar to each other by utilising diversity-based heuristics and other library code detection approaches [41]. Though this won't affect the analysis of benign data, the authors claim that Hancock can automatically build string signatures with an FPR of less than 0.1%. This is because the benign set is always evolving, making it impossible to extrapolate a satisfying result from a subset of the set to the full set. For future research to develop, these obstacles must be overcome. To detect malware, Santos et al. developed n-grams-based file signatures [68]. First, n-grams are extracted from each file that currently has been identified in order to establish the file signature. Then, with the measurement function and k-nearest neighbour approach, the file is classified as either dangerous or harmless [69]. Next, an n-gram is generated for each unknown occurrence. According to a study, the identification capability of n-grams-based signatures for illnesses that are unknown is low.

An efficient signature-based malware detection technique for mobile devices is proposed in [70]. Initially, a signature was acquired. Second, to speed up scanning, the hashing data for signatures was saved in a hash table. Lastly, the signatures are compared using the signature matching algorithm. The possibility that signature bytes may show up in innocuous information has been considered in order to eliminate the discrepancies. The recommended schema, according to the authors, performs better than the clam-AV scanner and provides notable memory saves at a high scanning speed. The only instrument available for comparing the proposed system was the Clam-AV scanner, which is not enough for a thorough examination. Zheng et al. [71] presented Droid Analytics, an Android malware detection programme. It has the ability to recognise potentially harmful code segments, gather malware automatically, supply application signatures, and link the malware under examination to further database infestations. The proposed technique identifies each application individually using a three-level signature generation mechanism. The authors state that the suggested signature technique offers several benefits over conventional cryptographic hashes, such as MD5-based signatures, and is resistant to packing and mutations.

B. Behavior based Detection Process

Behavior-based malware detection uses monitoring tools to observe a program's actions and determine the level of threat it poses. With the use of this method, most newly discovered malware may be found even when

programme behaviour changes but programme code stays the same [29]. Some harmful applications, nonetheless, struggle to function under secure settings like a virtual machine or sandbox. This might cause malware samples to be incorrectly categorised as safe.

Related Resources for Behavior-Based Detection:

System calls provided an explanation for the programmes' commonalities [36]. Wagener et al. presented a flexible and automated method for collecting malicious behaviour via system calls. Commonalities were found using the alignment approach, and associated distances could be calculated thanks to the Hellinger distance. As to the research, discernible characteristics enable the differentiation of various forms of malware that are disguising themselves. According to the authors, a phylogenetic tree that illustrates the characteristics that malware shares might help with the categorising process. The behavior-based detection approach is provided by Fukushima et al. in [72]. Malware that is unknown or encrypted can be found using the Windows operating system's suggested method. The suggested method examines both the particular actions that malware does as well as the routine chores that it typically shies away from. In the absence of FP, the researchers discovered that DR varied from 60% to 67%. The degree of regression (DR) is quite low; more tests will be conducted to assess the efficacy of the innovative technique if more harmful habits are found, at which point the DR will rise. In [73], a notion for semantic-aware viral detection is introduced. The authors have come to the conclusion that some harmful traits, like a polymorphic virus's decryption loop, are present in all strains of a certain infection. The researchers claim that an experimental investigation demonstrated the algorithm's ability to accurately identify any type of malware with no false positives and resistance to obfuscation tweaks. The approach is limited in terms of obfuscation adjustments.For example, it has issues detecting malware that use this method and rewriting instructions. Fixing issues with instruction substitution and changing the memory update procedure can improve performance. Algorithms for behaviour identification that restrict the number of characteristics are provided by [74], [75], and [38]. A behaviour model that is system-centric was proposed by Lanzi et al. [74]. According to the suggested paradigm, when malicious software communicates with computer resources (files, directories, registry entries, etc.), it behaves differently from benign software. The software under evaluation was compared to the behaviour sequences of the malware and benign categories. According to the authors, a significant portion of malware was detected by the suggested approach using only a few FP. The recommended approach was unable to find any potentially harmful activities, such as malware that just propagates across computer networks and makes no attempt to hide or take over the operating system. Rules and suggestions on the network for malicious software that disregards other applications and the OS can be implemented to improve performance. Bounded Feature Space Behaviour Modelling (BOFM) by M. Chandramohan et al. minimises the number of attributes required for malware identification [75]. Using this method, system calls were transformed into high-level behaviours, which were subsequently utilised to construct features. To ascertain if an application is risky or not, a feature vector was created using machine learning techniques. Because of its finite size, BOFM cannot grow to accommodate more malware samples. BOFM is hence incredibly scalable and efficient. The number of system calls was ignored by this technique. In order to extract the system calls and the features from the behaviours, the scientists employed a frequency-centralized model (FCM). The ML classifier was trained with characteristics from both benign and malicious samples to detect malware. As to the study, the proposed method exhibits superior classification accuracy, rapid deep learning, low power consumption, and the capacity to identify novel malware samples. Furthermore, the performance of the proposed approach has only been compared with BOFM and n-gram, which is not enough to evaluate the efficacy of the proposed algorithm. MapReduce was utilised by Liu et al. [76] to identify malware and categorise its behaviours. The majority of research, according to the authors, has been process-oriented and has only classified an activity as malicious when it makes system calls. On the other hand, the majority of contemporary malware, also referred to as sophisticated malware, is installed on the computer via DLLs or drivers and consists of many processes [77]. Malware occasionally uses several processes rather than just one to operate on a victim's computer. If malware just examines a single process, it may be considered innocuous. In the investigation, Auto-Start Extensibility Points (ASEP) were used to find recurrent patterns that helped differentiate between safe software and malware. The experimental results showed that the DR was 28% more successful than the previous study. However, there are several problems with the recommended course of action. To lessen the drawbacks of this strategy, the actions listed below can be implemented:

- In the absence of ASEP, certain malware binaries exhibit persistent activity.
- The cost of transmitting information cannot be measured; system calls are necessary for continuous malware activity.

A graph-based detection method was created in [79][37]. Kolbitsch et al. presented a detection approach based on graphs [79]. The process entails projecting system calls onto a behaviour graph, in which they are shown as nodes

and transitions between them are indicated by edges that indicate data dependency. The stated programme graph is obtained and compared to the present graph in order to determine whether a software is malware. The above-mentioned approach works well against known malware, however it has problems recognising fresh infections. provides a graph-based method that clarifies the usual traits of both malicious and benign samples [37]. The suggested system found kernel objects using system calls, which were then used to construct behaviours. The authors state that the recommended method for identifying unknown malware is scalable, has a high discovery ratio (DR), and has a low false positive (FP) rate. Moreover, even with more instances added, the proposed paradigm remains incredibly scalable and resilient to system call assaults. However, the proposed method can only identify a partially executable activity. It might be more precise to look at different uses for this technique.

C) Malware Detection Based On Heuristics Method

In recent years, heuristic-based detection techniques have become increasingly popular [81]. This advanced detection technique combines first-hand expertise with several techniques, such as machine learning methodologies and restrictions. [10]. To a certain degree, it can correctly identify zero-day malware, but it cannot detect complex malware.

Related Works For Heuristic-Based Detection

AIn order to detect viruses, Reynolds and Tesauro created a synthetic Win32 heuristic in [82]. They build many neural network classifiers automatically that are capable of identifying unknown Win32 infections. The authors assert that by combining the outcomes of every classification via a voting process, the chance of false positives (FP), which are frequently present in heuristic schemas, may be arbitrarily reduced. The study may be expanded to cover more potential risks, even though it is now restricted to Win32 viruses. More malware has to be examined for this strategy. Performance can be enhanced by the heuristic traits that experts have established. Associative classification post-processing approaches were proposed by Yanfang et al. [83] for malware detection. The proposed method greatly reduced the number of developed rules by using rule creation, rule categorization, and rule trimming. This approach reduced both the rate of efficiency and identification time and did away with the requirement for technologies to operate on a big rule database. The research-based recommendation method fared better than well-known anti-virus software programmes like McAfee, Virus Scan, and Norton AntiVirus, as well as data-mining-based detection techniques like naive Bayes, SVM, and decision tree algorithms. Gathering additional API calls might improve speed by revealing more details about malware and complex relationships between requests.

Employing statistical analysis of opcode frequency distributions, [86] describes how to differentiate and identify between the two types of malware that are now in circulation: polymorphic and metamorphic. After they were statically disassembled, the statistical opcode patterns of frequency of 67 malicious programme files were gathered and compared with the whole data for 20 non-malicious cases. The test's findings showed that there were statistically significant differences in the opcode distributions of malicious and legitimate files. To obtain more reliable results, additional samples need to be analysed, and the results of the suggested approach need to be compared with the findings from other well-known heuristic techniques. A hybrid detection technique that uses both static and dynamic data is presented in [43]. The feature vector created by integrating these attributes was classified using an ML classifier. The FPR is high, yet it's still rare to find undiscovered malware. To increase the effectiveness of the technique for undetected malware, more malware and differentiators may be used to train the framework. Naval et al. suggested a strategy for adaptable malware identification [44]. After compiling the system calls, a graph that shows the logically connected paths is created. Finding every semantically significant route in a network is NP complete. In order to reduce temporal complexity and demonstrate malware characteristics lacking in benign samples, the authors evaluated the most significant channels. According to the authors, the suggested method outperforms the others since it has a higher detection rate for malware, even in the presence of system-call injection attacks. Among the study's drawbacks are a call-injection vulnerability, an efficiency cost during path computing, and challenges in accurately identifying all pathways that are semantically significant. Removing these restrictions could enhance the output. Every technique varies in the characteristics it adopts from the others. It was difficult to determine which detection technique was superior because each had pros and cons. By using these methods, new malware may potentially be discovered. They are yet unable to recognise every kind of virus.

VI. Conclusion

Although these various approaches have caused the development of numerous new malware recognition algorithms, no single method has been able to recognize each new type of sophisticated malware. Heuristic and

signature-based approaches are effective in identifying known malware. Behavioral techniques, on the other hand, work better for novel and complex disorders. But these techniques weren't able to identify every malware. This demonstrates the challenges involved in developing an effective malware detection system as well as the ongoing need for research and improvement. Even though market trends are always shifting, this knowledge is nevertheless useful for computer engineers and researchers that work in the field of malware production and detection. Future studies must propose novel strategies and techniques. To do this, combining several virus detection methods can be helpful. For example, a more robust detection system will surely arise from the simultaneous use of deep learning, model testing, and behavior-based approaches. Malware attacks have become more severe, frequent, and sophisticated in recent years, and they have also raised the cost to the world economy. These software variations have the ability to initiate devastating attacks that put the resources of individuals, governments, and private enterprises in grave danger. Finding antivirus software is necessary to safeguard the company's priceless assets. On the other hand, there are currently insufficient research in several domains on malware detection and avoidance. Research on malware is the aim of this project. The methods and algorithms utilised in malware detection, as well as the latest research on these topics, are all carefully examined in this paper.

VI. Discussion

This section presents a statistical evaluation of the studied methods for data mining malware detection. The statistical flowchart for each of the classification techniques used in the chosen malware detection strategies is displayed in Figure 2. The SVM technique has the highest proportion (29%), followed by j48 (17%), NB (10%), RF (5), and ANN 3% in this report's malware detection strategy. The other methods' utilisation in data mining findings is less than 2%. Through data mining, we find that among signature-based malware detection techniques, the SMV method simply has the greatest accuracy.

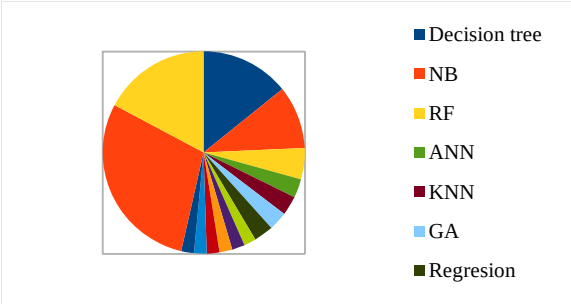

Figure 2: Techniques of classification used in malware detection systems

The several malware datasets that researchers have utilised are shown in Table 3.

Table III. Researchers' Use Of Malware Databases

Malware database	**Description**
Microsoft Malware Classification Challenge (BIG 2015)	dataset utilised in a competition to create malware classification algorithms. includes benign samples and malware with labels.
The Adversarial Malware Learning (AML) Dataset	An executable's maliciousness may be detected using a machine learning model trained on this dataset. includes labelled malware samples for scientific use.
VirusShare	malware samples supplied by researchers and security experts in a private repository. includes a huge assortment of harmful scripts, documents, executables, etc.
Malware-Traffic-Analysis.net	gives access to PCAP files that record actual harmful network traffic for studies on the behaviour of malware in network settings.
Contagio Malware Dump	community-driven repository where people exchange samples of different kinds of malware.
MalShare	Community-driven archive providing a vast array of various malware samples for study and examination.

REFERENCES

[1] Steve Morgan. (2019). The 2019 Cybersecurity Almanac has one hundred facts, figures, forecasts, and statistics. Cybersecurity Ventures and Cisco. taken from [Cybersecurity Adventures: https://cybersecurity-almanac-2019]

[2] Davis, G., and Samani, R. (2019). Q1 of the McAfee Mobile Threat Report. from [https://www.mcafee.com/enterprise/en-us/assets/reports/rp-mobile-threat-report-2019.pdf] obtained the following.

[3] Fred. Cohen (1986). (Doctoral Thesis, Graduate School of Southern California, Los Angeles, USA): Computer infections.

[4] Cohen fred. (1992). A precise description of computer worms along with some associated findings. 11(7) Computer Security, 641–652.

[5] White, S. R., and D. M. Chess (2000). an unnoticeable virus on computers. Virus Bull. Conf. Proc. (Vol. 5).

[6] F. Cohen,(1987) [6]. Theory and experimentation on computer viruses. Computer Security, 6 (1).

[7] L. M. Adleman (1990). a theoretical framework for computer viruses. Springer-Verlag, in Innovations in Cryptology—CRYPTO.

[8] Spinellis (2003) p. It is NP-complete to reliably identify bounded-length viruses. IEEE Information Theory Transactions, 49(1).

[9] Zuo (2005), Zhou (2005), and Zhu (2005). Regarding the intricacy of computer viruses, see imeickecs2024@sjcit.ac.in. IEEE Information Theory Transactions, 51 (8).

[10] K. Mohamed Abdelrahman Y AlzaroониAlzarooni (2012), Malware variant identification (Doctoral thesis, Department of Computer Science, University College of London, U.K.,).

[11] Peter Szor. (2005) . Computer Virus Research and Defense: An Art Form. The (Symantec Press) Paperback – 17.

[12] lawrie Brown., and William Stallings (2012). Computer Security: Fundamental and Applications. Pearson .

[13] Ferrie, P., and Szor, P. (2001). Hunting for metamorphic. Proceedings of the Virus Bull. Conference,123-143.

[14] Alam, Sogukpinar, I., Horspool, R., and Traore, I. (2015). as a part of new framework for analyzing metamorphic malware and detecting it instantly. 48. Computer Security.

[15] M. D. Preda (n.d.). Abstract Interpretation for Malware Detection and Code Obfuscation. from [https://iris.univr.it/retrieve/handle/11562/337972/3306/main.pdf] obtained.

[16] Yan (2008), Zhang (2008), and Ansari N. revealing packaged malware. Journal of IEEE Security & Privacy, 6(5), 65–69.

[17] Y. Alosefer (2012). Using client honeypots to analyze the behavior of Web-based malware (Doctoral dissertation, Cardiff University, Cardiff, U.K., School of Computer Science and Informatics).

[18] Honig, A., & Sikorski, M. (n.d.). The hands-on guide to dissecting malicious software, Practical Malware Analysis. USA: San Francisco, CA.

[19] Mathur, P., & Idika, N. (2007). An examination of malware detection methods (Tech. Rep. No. 48). University of Purdue, West Lafayette, IN, USA.

[20] Eilam (2011), Eilam. Reversing: Reverse Engineering Secrets. Wiley.

[21] Hosseini, R., and Souri, A. (2018). An advanced analysis of malware detection methods utilizing data mining techniques. Centric Hum

[22] M. Tavallaee (2009). A thorough examination of the KDD CUP 99 dataset. In Proc. IEEE Symp. (CISDA2009).

[23] Arp, D., Hubner, M., Gascon, H., & Rieck, K. (2014); Spreitzenbarth, M., Hubner, M. Drebin: A portable, clear way to identify Android malware. In Proceedings. NDSS .

[24] Ronen, Radu, Feuerstein, C., Yom-Tov, E., & Ahmadi, M. (2018). Microsoft malware classification contest. 1802.10135 arXiv. taken from [https://arxiv.org/abs/1802.10135]

[25] Malware PE Header classification. (As of now). taken from [https://github.com/urwithajit9/ClaMP]

[26] Lashkari, A. H., Mbah, K. F., Gonzalez, H., Kadir, A. F. A., & Ghorbani, A. A. (2017). Moving toward a network-based architecture for characterizing and detecting Android malware. In Proceedings of the 15th Annual Conference on Privacy and Security (PST), August 2017.

[27] Roth, P., & Anderson, S. (2018). EMBER: A publicly available dataset used to train machine learning algorithms for static PE malware. arXiv:1804.04637. Extracted from [https://arxiv.org/abs/1804.04637]

[28] Gandotra, Bansal, & Sofat (2014); Gandotra, E. A survey on the analysis and categorization of malware. Information Security Journal, 5(2), 56–64.

[29] Samet, R., and Aslan, O. (2017). examination of the potential for malware detection with the techniques already in use. In Proceedings of the 14th International Conference on Computer Science and Applications (IEEE/ACS), October 2017.

[30] Zadok, F., Eskin, E., Schultz, M. G., & Stolfo, S. J. (2001). techniques for data mining to find new dangerous executables. In IEEE Symp. Secur. Privacy Proceedings, May 2001.

[31] Goswami, S., Guha, R., and Karnik, A. (2007). Cosine similarity analysis for the detection of obfuscated viruses. First Asia International Conference on Modeling Simulation (AMS), March 2007.

[32] Cha, S. K., & Andersen, D. G. (2011); Moraru, I., Jang, J., Truelove, J., & Brumley, D. SplitScreen: Facilitating effective, decentralized malware identification. 187–200 in Journal of Communications and Networks, 13(2).

[33] Baldangombo, U., Horng, S.-J., & Jambaljav, N. (2013). a data-mining-based static malware detection solution. arXiv:1308.2831. Excerpted from [https://arxiv.org/abs/1308.2831]

[34] Bajaj, K., & Malhotra, A. (2016). A hybrid pattern-based text mining method that makes use of DBScan for malware detection. 141–149 in CSI Transactions, 4(2-4).

[35] Kruegel, C., Kirda, E., & Moser, A. (2007). investigating many ways for malware to execute. In IEEE Symp. Privacy (SP) Proceedings, May 2007.
[36] Dulaunoy, A., Wagener, G., and State, R. (2008). An investigation of malware activity. Computer Virology Journal, 4(4), 279–287.
[37] Park, Y., Stamp, M., & Reeves, D. S. (2013). using graph clustering to infer typical virus activity. 419–430 in Computers & Security, 39.
[38] Chandramohan, M., Zhang, W., Liu, Y., & Das, S. (2016). Online malware detection using semantics: A path toward effective, real-time virus prevention. 11(2), 289–302, the IEEE Transactions on Information Forensics and Security.
[39] Zamini, S. M., Souri, A., and Norouzi, M. (2016). a classification method based on data mining for behavioral malware identification. Journal of Communications and Computer Networks, 1 (2016).
[40] Zhang (2007), Yin (2007), Hao (2007), Zhang D., & Wang S. Ensemble learning-based detection of malicious programs. Pages 468–477 in Autonomic and Trusted Computing. Springer.
[41] Griffin, K., Chiueh, T.-C., Schneider, S., and Hu, X. (2009). String signatures are automatically generated for malware detection. In Proceedings of International Workshop on Advanced Intrusion Detection. Springer.
[42] Lane, T., Neil, J., Quist, D., Anderson, B., & Storlie, C. (2011). Dynamic analysis for graph-based malware identification. Computer Virology Journal, 7(4), 247–258.
[43] Islam, R., Versteeg, S., Tian, R., and Batten, L. M. (2013). Malware classification using combined dynamic and static characteristics.36(2), 646–656 in Journal of Network and Computer Applications.
[44] Gaur, M. S., Naval, S., Laxmi, V., Rajarajan, M., & Conti, M. (2015). using program semantics to identify malicious software. IEEE Information Forensics and Security Transactions, 10(12), 2591–2604.
[45] Lakhotia, A., and Singh, P. (2003). Model checking is used to verify worm and virus behavior statically in binary executables. In the IEEE Systems, Man-Soc Assurance Workshop Proceedings, March 2003.
[46] Veith, H., Schallhart, C., Kinder, J., & Katzenbeisser, S. (2005). use model checking to identify fraudulent code. In Proc. Int. Conf. Malware, Vulnerability Assessment, Detection Intrusions. Springer, Berlin, Germany.
[47] Marion, J., and Beaucamps, P. (2009). about behavioral detection. in EICAR Proceedings, vol.
[48] Touili, T., and Song, F. (2012). Model checking for effective malware detection. Proc. Int. Symp. Formal Techn. Springer, Berlin, Germany.
[49] Santone, A., Nardone, V., Cimitile, A., Martinelli, F., Mercaldo, F., & Vaglini, G. (2017). Model checking for the growth of Android mobile malware. In Proceedings of the IEEE/ACM Fifth International FME Workshop on Formal Methods of Software Engineering.
[50] Berlin, K. & Saxe, J. (2015). Using two-dimensional binary program characteristics, malware is detected using deep neural networks. Proceedings. 10th International. Conference. Malicious Unwanted Software (MALWARE), October 2015.
[51] Stokes, J. W., & Huang, W. (2016). MtNet: A multitask neural network designed to classify malware dynamically. In Proc. Int. Conf. Malware, Vulnerability Assessment, Detection Intrusions. Springer, Cham, Switzerland.
[52] Wu, D., Chen, W., Jin, H., Yang, Y., and Zhu, D. (2017). DeepFlow: An Android application that uses deep learning to identify malware by looking for unusual uses of sensitive data. In IEEE Symposium. Comput. Commun. (ISCC) Proceedings, July 2017.
[53] In 2018, Ye Y., Chen L., Hou S., Hardy W., & Li X. published a paper. DeepAM: An intelligent malware detection system utilizing heterogeneous deep learning. Systems of Knowledge and Information, 54(2), 265–285.
[54] Bruschi, D., Martignoni, L., and Paleari, R. (2009). A cloud-based malware analysis methodology based on behavior. Proceedings of the International. Conf. Inform. System. Security. Springer, Berlin, Germany.
[55] Chen, P., Su, J., Wang, X., & Sun, H. (2015). RScam: Reversible sketch anti-malware deployed on the cloud. Proceedings. International. Conference. Security. Privacy Commun. Syst. Springer, Cham, Switzerland.
[56] Li, Y., X. Huang, X. Xiao, & X. Du (2017). Offloading mobile virus detection game based on the cloud. IEEE Mobile Computing Transactions.
[57] Takemori, K., Isohara, T., and Kubota, A. (2011). Behavior analysis based on the kernel for Android malware identification. In Proceedings of the 7th International Conference on Computer-Assisted Intelligence Security.
[58] Shabtai, A., Glezer, C., Elovici, Y., & Weiss, Y. (2012). Kanonov, U. Andromaly: A framework for detecting behavioral malware on Android smartphones. Journal of intelligent information system.
[59] Chen, L., Liu, Y., Chandramohan, M., and Narayanan, A. (2017). Online learning for context-aware, adaptive, and scalable Android malware detection. 1(3), 157–175 in IEEE Transactions on Emerging Topics in Computing.
[60] Choo, K.-K.-R., Conti, M., Dehghantanha, A., and Azmoodeh, A. (2018). identifying crypto-ransomware in Internet of Things networks by looking at energy footprints. Journal of Humanized Computing and Ambient Intelligence.
[61] K. Hahn. (2014). thorough static analysis of malware using portable executables. In Proc. Leipzig HTWK.
[62] Mnemonics hooked me in the end.(2011). From [http://hooked-on-mnemonics.blogspot.com/2011/01/intro-to-creating-anti-virus-signatures.html] (retrieved)
[63] "A framework for malware detection using combination technique and signature generation," by M. F. Zolkipli and A. Jantan. In Proceeding. 2nd Int. Conf. on Computer-Assisted Reading Development.
[64] Tang, Y., Xiao, B., & Lu, X. "Generating accurate exploit-based signatures for polymorphic worms using a bioinformatics approach." Computers and Security,636-646.
[65] Abadi, M., & Razeghi Borojerdi, H. (2013). MalHunter: Multiple behavioral fingerprints are automatically generated for the identification of polymorphic malware. Ferdowsi University of Mashhad, Iran, Proceedings of the ICCKE.

[66] Karp, B., Song, D., & Newsome, J. (2005). Polygraph: Producing polymorphic worm signatures automatically. Proceedings of the IEEE Symposium on Security and Privacy (S&P), Oakland, CA, USA, May 2005.
[67] Feamster, N., Lee, W., and Perdisci, R. (2010). HTTP-based malware behavioral clustering and signature development through the use of harmful network traces. Proceedings of the 7th USENIX Conference on Network Systems Design and Implementation, San Jose, CA, USA, pp. 391–404.
[68] Bringas, P. G., Devesa, J., Penya, Y. K., and Santos, I. (2009). N-grams-based file signatures to identify malicious software. pp. 317–320) in Proceedings. 11th Int. Conference. Enterprise Inf., vol. 9.
[69] Fix, E., and J. L. Hodges (1952). Nonparametric discrimination in discriminatory analysis: Small sample performance (Tech. Rep. No. 11). California University, Berkeley, Berkeley, California, USA.
[70] Hu, G., & Venugopal, D. (2008). Effective virus detection on mobile devices using signatures. 33–49 in Mobile Information Systems, 4(1).
[71] Lui, J. C., Zheng, M., and Sun, M. (2013). Droid analytics: An analytical system based on signatures that gathers, extracts, examines, and links Android malware. 12th IEEE International Conference on Trust, Security, and Privacy in Computing, July 2013 (pp. 312–322).
[72] Sakai, A., Hori, Y., Sakurai, K., & Fukushima, Y. (2010). a malware detection method that is behavior-based to prevent false positives. 79–84 in Proceedings of the 6th IEEE Workshop on Secure Network Protocols, October 2010.
[73] Christodorescu, M., Bryant, R. (2005), Song, D., Jha, S., and Seshia, S. malware detection with semantic awareness. In May 2005, Proceedings. IEEE Symposium. Security. Privacy (S&P).
[74] Lanzi, A., Christodorescu, M., Balzarotti, D., Kruegel, C., & Kirda, E. (2010). AccessMiner: Protecting against malware with system-centric concepts. (pp. 399–412) in Proceedings. 17th ACM Conference. Comp. Communi. Secur.
[75] Tan, H. B. K., Briand, L. C., Shar, L. K., Chandramohan, M., & Padmanabhuni, B. M. (2013). a scalable method using limited feature space behavior modeling to identify malware. 28th IEEE/ACM Int. Conference. Automated. Software. Engineering. (ASE), Nov. 2013 (pp. 312–322).
[76] Huang, H.-C., Liu, S.-T., & Chen, Y.-M. (2011).a MapReduce-based system call analysis technique for malware identification. In IEEE 17th International Conference on Parallel Distributed Systems, December 2011
[77] U.,Bayer, I.Habibi, Kirda,E, Balzarotti,D. & Kruegel, C. (2009). An analysis of contemporary malware activities. Proceedings of USENIX Workshop.
[78] Pajouh, H. H., Choo, K.-K.-R., Dehghantanha, A., and Khayami, R. (2018). Code inspection combined with intelligent OS X malware threat detection. 14(3), 213–223 in Journal of Computer Virology and Hacking Techniques.
[79] Zhou, X.-Y., Kruegel, C., Kirda, E., Kolbitsch, C., Comparetti, P. M., & Wang, X. (2009). Malware detection that is both successful and efficient at the end host. Proceedings of the USENIX Security Symposium, August 2009.
[80] Hashemi, S., & M. Eskandari (2012). a method for identifying unknown viruses via graph mining. 23(3), 154–162 in Journal of Visual Languages & Computing.
[81] Adkins, F., Upchurch, J., Jones, L., and Carlisle, M. (2013). Basic block comparison is used for heuristic malware detection. In Proceedings of the 8th International Conference on Malicious Unwanted Software, America (MALWARE), October 2013.
[82] Tesauro, G., and Arnold, W. (2000). virus detection using Win32 heuristic created automatically. Virus Bull. Conf. Proc. Int., vol. 200.
[83] Wang, Y., Jiang, Q., Li, T., and Ye, Y. (2010). CIMDS: Using associative classification postprocessing algorithms to identify malware. Part C (Applications and Reviews) of IEEE Transactions on Systems, Man, and Cybernetics, 40(3), 298-307.
[84] In 2008, Ye Y., Wang D., Li T., Ye D., & Jiang Q. published [84]. an association mining-based clever PE-malware detection method. Computer Virology Journal, 4(4), 323–334.
[85] Hashemi, H., Fard, S. M. H., Bazrafshan, Z., and Hamzeh, A. (2013). An overview of heuristic methods for detecting malware. In Proceedings of the Fifth International Conference on Information and Technology, May 2013.
[86] D. Bilar (2007). Opcodes as a malware predictor. International Journal of Digital Forensics and Electronic Security, 1(2), 156.