Volume 9, Issue 3, March – 2024 International Journal of Innovative Science and Research Technology
ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAR615

A Machine Learning-Based PE Header Analysis for Malware Detection

Shruti Gujar[1]
Department of Computer Science and Engineering
Dr. D Y Patil College of Engineering and Technology, Kolhapur

Saurabh Patil[2]
Department of Computer Science and Engineering
Walchand College of Engineering, Sangli

Abstract:- The malware is file or piece of code which is delivered over network that infects or conducts any behavior as attacker desired. So, it is one of the most serious threats to modern world specially who are in touch of computerand information technology. The older signature-based detection is not convenient all the time. This was not the perfect approach as it was detection which uses unique signature or digital footprint from software running on secured system. This method is used in antivirus programs. These programs scan any software program and try to identify the signatures. These signatures are then compared to signature of known malwares. But signatures may not be known to us every time. This method has some lots of limitations. It is unable to the new patterns or indicators of new threats that are not already known. As a result, security professionals often this method in conjunction with tools that provide context into their network behavior. The PE is actually file layout that is present in .exe, .dll file formats and other machine level code and their PE headers contains information that can help us distinguish between malicious malware files and legitimate files. This method is helpful to find hidden patterns and to establish new techniques to recognize the files. The virus share suffixedfiles are also the files which performs malicious activities and malware in nature. Even this type of files can be trained and recognized with the help of PE headers-based method to recognize the nature of file. Its possible to identify the malware by looking at some key features from headers such as checksum, section name, initialized data Size, DLL characteristics and major image version. Looking at the PE header is much faster than scanning the whole information in the PE. Thus, the prediction of files are possible even with faster rate. In this paper, we will understand the different attributes available in PE headers to carefully analyses the trends and to distinguish the given executable files as malicious or legitimate on basis of PE headers using advance machine learning algorithms.

Keywords:- *Machine Learning, PE Headers, Classifications, Malwaredetection, PE Header Table.*

I. INTRODUCTION

Malwares are malicious files and it can damage to our important files in computer systems [1]. We can preventthis threat by identifying it. These files are traditionally judged on the basis of signatures [2]. The signature-based method is one of the methods which helps to distinguishthe difference between malicious and legitimate files.

- *Signature based Technique:*

Antivirus solutions play a crucial role in protecting computers and devices from the pervasive threat of malicious software, commonly referred to as malware. These security tools employ a robust defense mechanism known as signature-based detection, a stalwart method inthe cybersecurity domain for many decades.

Signature-based detection entails a meticulous examination of a computer or device for traces, or footprints, of previously identified malware. These distinctive footprints are carefully cataloged and stored in a comprehensive database [3]. During a scan, antivirusproducts diligently compare the files and processes on the system against this extensive database. The primary objective is straightforward: if a footprint matching that of a known malware is detected, the antivirus software promptly flags it as a security threat.

The effectiveness of this approach hinges on the collaborative efforts between cybersecurity experts and the antivirus community. In response to the emergence of new malware types, experts thoroughly analyze their characteristics and expeditiously add their signatures, or footprints, to the continually expanding database [4]. This collective endeavor ensures that all antivirus products leveraging this shared repository can promptly recognize and neutralize newly identified malware during routine scans.

- *PE Headers-based Technique:*

PE (Portable Executable) file format is a data structure that tells the Windows OS loader what information is required to manage the wrapped executable code. This includes dynamic library references for linking, APIexport, import tables, resource management data, and TLS data [5].

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAR615

The data structures on disk are the same data structures used in the memory, and if you know how to find something in a PE file, you can almost certainly find the exact information after the file is loaded into the memory. It is important to note that PE files are not just mapped into memory as a single memory-mapped file. Instead, the Win32 loader looks at the PE file and decides what portions of the file to map in.

A module in memory represents all the code, data, and resources from an executable file needed by the process. Other parts of a PE file may be read but not mapped in (for instance, relocations). Some parts may not be mapped in at all, for example, when debug information is placed at the end of the file.

A field in the PE header tells the system how much memory needs to be set aside for mapping the executableinto memory. Data that won't be mapped in is placed at the end of the file, past any parts that will be mapped in. These files can be studied to identify trends and finally we can get the nature of files and their behavior.

II. LITERATURE SURVEY

In the dynamic realm of cybersecurity, our exploration ofinnovative avenues finds inspiration in a compellingbody of research focused on malware detection. Several noteworthy studies have captivated our attention and ignited collective curiosity within our team.

One such impactful study [1] stands out for its ingenious use of information theory in classifying packed and encoded files. The application of theoretical frameworks to enhance our understanding and detection capabilities in the face of sophisticated file manipulations has sparked our interest and set the stage for further exploration.

Another significant contribution [2] offers a comprehensive overview of signature-based techniques in antivirus products. This review prompts us to critically reassess and refine our approach to signature-based detection, aiming to elevate our current practices and contribute to more effective antivirus solutions.

The exploration of the pipeline process in both signature- based and behavior-based malware detection, as outlined in [3] provides a valuable roadmap for our team. The systematic breakdown of stages and processes offers a blueprint to enhance our operational frameworks and potentially uncover synergies between these detection approaches.

Additionally, [4] introduces the integration of machine learning into malware detection. This study fuels our enthusiasm for incorporating intelligent systems into our cybersecurity efforts, envisioning the application of diverse machine learning algorithms to fortify our classification capabilities.

As we absorb the insights from these impactful studies, a collective sense of purpose emerges, propelling us to contribute to the evolving landscape of cybersecurity. The intricate dance between theoretical foundations, signature-based methodologies, and the potential unleashed by machine learning beckons us to chart our own course in advancing the realm of malware detection.

III. METHODOLOGY

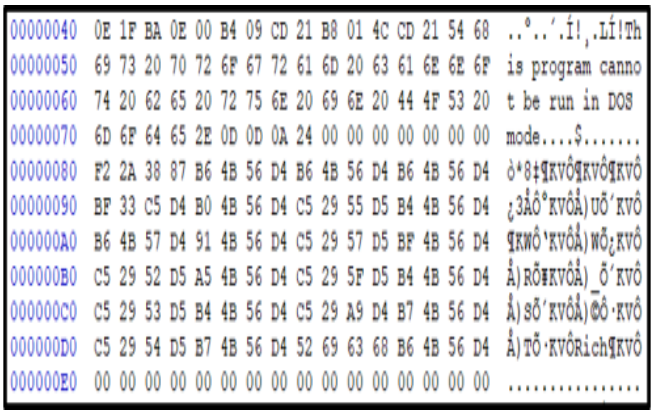

Fig 1 DOB Stub Program

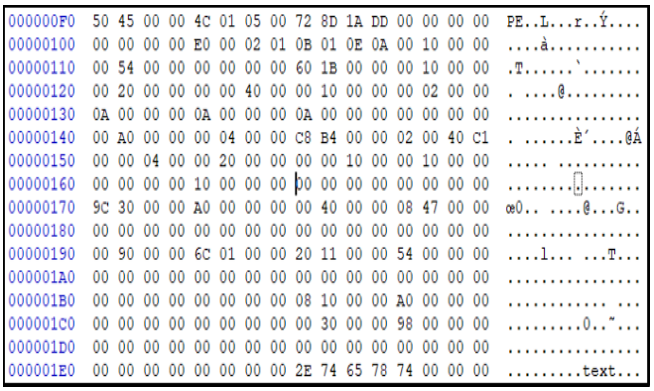

Fig 2 PE File Header

Starting Byte	Type	Information
1	WORD	Machine
3	WORD	Number of sections
5	DWORD	Time Date stamp
9	DWORD	Pointer to symbol table
13	DWORD	Number of symbols
17	WORD	Size of optional header

Fig 3 Image Header

➢ *Structure of PE:*

The above figures state the MS-DOS Stub Program which serves as an initial code segment that runs automatically when a Win32-based program encounters an incompatible environment. Its primary purpose is to display the message "This program cannot be run in DOS mode," offering an informative prompt in cases where the program is not compatible with the DOS operating system. This stub program is executed by MS-DOS and provides an indication that the executable is designed for the Windows environment.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAR615

Moving into the PE (Portable Executable) file structure, the PE File Header is located using the e_lfanew field of the MS-DOS Header. Comprising the SIGNATURE, IMAGE_FILE_HEADER, and IMAGE_OPTIONAL_HEADER, the PE Header offers crucial insights into the layout and characteristics of the executable file. It stands as a key reference point for understanding the underlying architecture and operational parameters of the program.

The IMAGE_OPTIONAL_HEADER, several pivotal fields dictate the behavior of the executable.The Magic field distinguishes between 32-bit (IMAGE_NT_OPTIONAL_ HDR32_MAGIC), 64- bit (IMAGE_NT_OPTIONAL_ HDR64_ MAGIC) applications. AddressOfEntryPoint designates the starting address for Windows loader execution, while ImageBase determines the memory-mapped address for the executable. Other fields, such as SectionAlignment, FileAlignment, SizeOfImage, and Subsystem, contribute to shaping the file's memory layout and operational characteristics.

The IMAGE_DATA_DIRECTORY array holds pointers to critical components within the executable, with entries like Export Directory, Import Directory, Resource Directory, and others. These directories play a pivotal role in establishing connections with external resources and functionalities, delineating the executable's dependencies and interactions. imported functions, respectively. These sections contribute to the overall functionality of the executable by defining its interactions with external components. The .edata and .idata sections are linked to export and import directory entries in the DataDirectory array, forming integral components of the executable's structure.

The PE file structure, encompassing the MS-DOS Stub Program, PE File Header, IMAGE_OPTIONAL_HEADER, and IMAGE_DATA_DIRECTORY, Section Header Table, and various sections, provides a comprehensive framework for understanding the architecture, dependencies, and behavior of Windows executable files [5]. See the given fig.4 PE file format and fig.5 structure of file for more information.

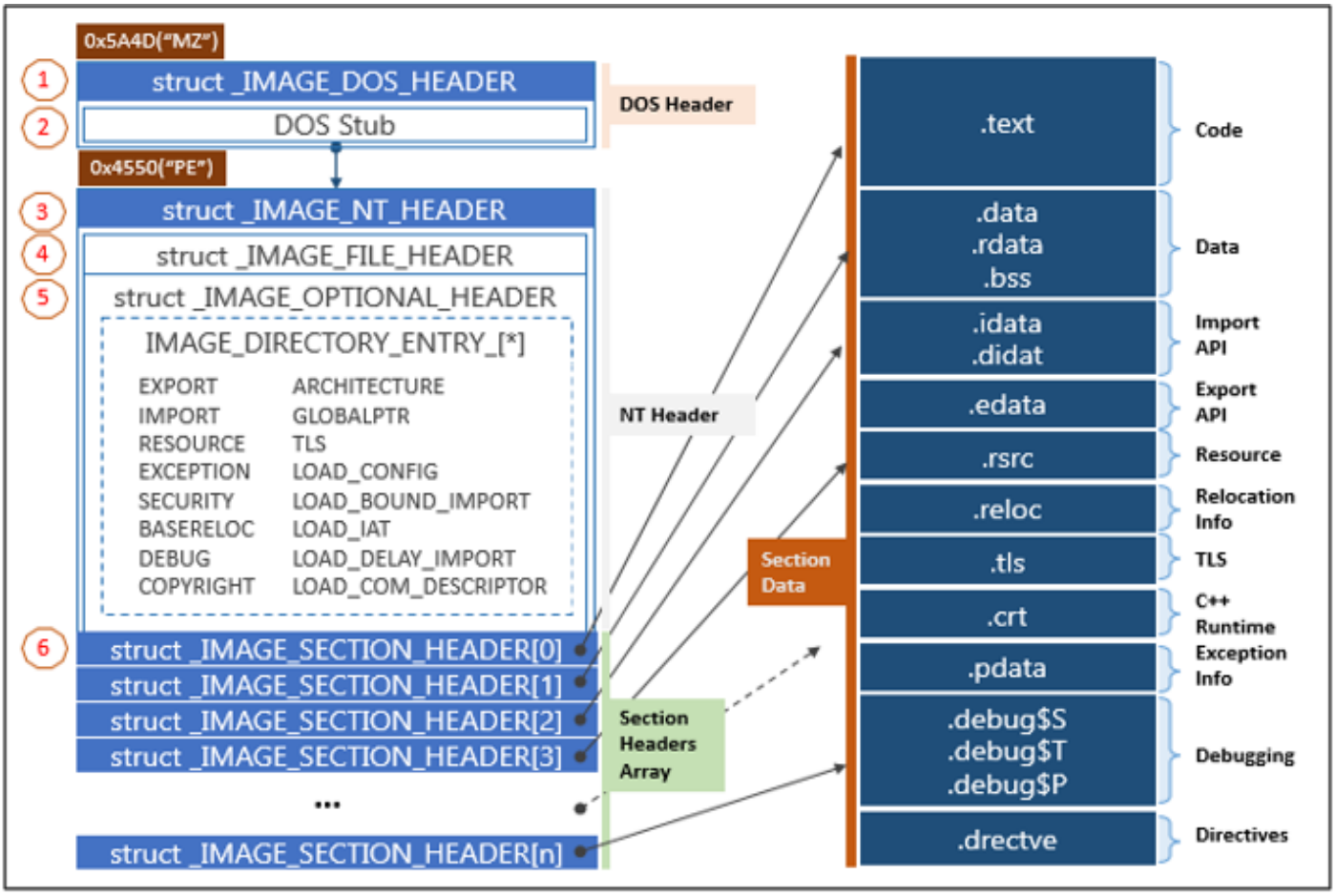

Fig 4 PE File Format

The Section Header Table, an array of IMAGE_SECTION_HEADER structures, provides a detailed account of the various sections within the executable. Fields like SizeOfRawData, VirtualSize, PointerToRawData, VirtualAddress, and Characteristics contribute to understanding the size,location, and properties of each section. These sections include .text for executable code, .data for initialized data, and others such as .rdata, .idata, .reloc, .rsrc, and .debug.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAR615

➢ *Execution:*

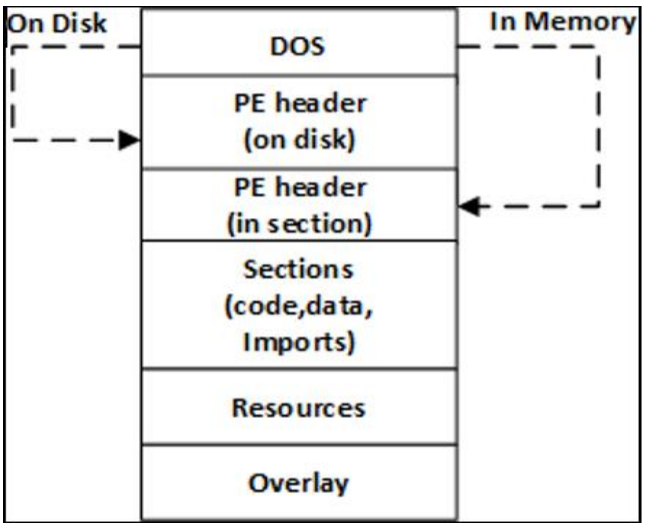

Fig 5 Structure of PE File

Two specific sections, .edata and .idata, merit attention for their role in listing exported and In conducting this research, the initial step involves the collection of a diverse dataset encompassing information related to files, with a particular emphasis on attributes indicative of their legitimacy and potential malware status. Following the data acquisition, the pandas library is employed to facilitate the importation of the dataset, reading it from a CSV file and specifying the appropriate separator. The subsequent data preprocessing stage involves a detailed exploratory analysis to comprehend the structure and characteristics of the dataset. Any instances of missing or irrelevant data are addressed meticulously to ensure the dataset is prepared for rigorous analysis.

The data exploration techniques are employed, leveraging descriptive statistics and visualizations to derive valuable insights. The presentation of the initial and final five records of the dataset, scrutiny of column names, and the provision of summary statistics contribute to a comprehensive understanding of the dataset. Visual representations, such as pie charts, are utilized to illustrate the distribution of legitimate and malware files within the dataset.

Feature selection is a critical aspect of the research methodology, employing machine learning techniques to identify key variables essential for effective classification. The Extra Trees Classifier is utilized to assess feature importance, facilitating the selection of pertinent features. Irrelevant columns, including 'Name,' 'md5,' and 'legitimate,' are excluded to construct the feature matrix (X) and target variable (y).

Data splitting is conducted to partition the dataset into training and testing sets, typically allocating 70% for training and 30% for testing. Stratification techniques are applied to maintain a balanced distribution of legitimate and malware samples in both sets.

The subsequent phase involves the evaluation of various classifiers, such as Decision Tree, Random Forest, and Gradient Boosting, assessing the accuracy of each model on the testing set. The classifier demonstrating the highest accuracy is chosen as the final model mainly Random Forest.

IV. RESULTS AND DISCUSSIONS

➢ *Accuracy of Classifiers for this Model:*

Table 1 Accuracy of Classifiers for this Model

Sr No.	Classifier	Accuracy
1	Decision Tree	99%
2	Random Forest	99.38%
3	Gradient Boosting	98.99%

The dataset has been structured to encompass a broad array of attributes related to files, offering an extensive basis for analysis and classification. Each column holds specific information, contributing to anuanced understanding of the dataset. Here is a breakdown of the key features without the use of double inverted commas:

The Name column serves as a unique identifier for files in the dataset. Meanwhile, md5 contains MD5 hash values, crucial for integrity verification. The Machine column specifies the file's targeted architecture, indicating the intended platform [6].

Size Of Optional Header denotes the size of the optional header in bytes, a vital aspect of the Portable Executable (PE) file format. Characteristics includes flags conveying various file attributes, such as executability or DLL status.

Major Linker Version and Minor Linker Version represent the major and minor version numbers of the linker used during compilation. Size-related information is provided by columns like Size Of Code, Size Of Initialized Data, and Size Of Uninitialized Data.

Address Of Entry Point designates the starting point for execution, while Base Of Code and Base Of Data indicate base addresses. Image Base specifies the preferred base address when loaded into memory.

Alignment details are captured by Section Alignment and File Alignment. Operating system requirements are outlined through Major Operating System Version and Minor Operating System Version. File versioning details are included in Major Image Version and Minor Image Version.

Size Of Image and Size Of Headers quantify the size of the image and combined header size. Check Sum ensures file integrity, and Subsystem specifies the required subsystem. Dll Characteristics focuses on characteristics specific to DLL files.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAR615

Memory-related details are covered by columns like Size Of Stack Reserve, Size Of Stack Commit,Size Of Heap Reserve, and Size Of Heap Commit. Loader Flags include flags indicating properties of the image loader.

Number Of Rva And Sizes represents the number of data-directory entries, and Sections Nb specifies the number of sections in the PE file. Entropy details for sections are conveyed by columns like Sections Mean Entropy, Sections Min Entropy, and Sections Max Entropy Raw and virtual sizes of sections are captured by columns like Sections Mean Raw size, Sections Min Raw size, Section Max Raw size, Sections Mean Virtual size, Sections Min Virtual size, and Section Max Virtual size.

Import-related insights are provided by Imports Nb DLL, Imports Nb, and Imports Nb Ordinal. Export Nb signifies the number of exported functions, and Resources Nb denotes the number of resources. Entropy statistics related to resources are presented in columns like Resources Mean Entropy, Resources Min Entropy, and Resources Max Entropy. Resource sizes are captured by Resources Mean Size,Resources Min Size, and Resources Max Size.

- *The Important Columns Selection using Extra – Tree Classifier Fitting:*

Table 2 The Important Columns Selection using Extra – Tree Classifier Fitting

Sr No	Features	Importance level
1	Dll Characteristics	(0.154796)
2	Characteristics	(0.117378)
3	Machine	(0.098500)
4	Subsystem	(0.064115)
5	Version Information Size	(0.060187)
6	Sections Max Entropy	(0.058773)
7	Major Sub System Version	(0.053747)
8	Size Of Optional Header	(0.046769)
9	Image Base	(0.044347)
10	Resources Min Entropy	(0.038087)
11	Resources Max Entropy	(0.033732)
12	Major Operating System Version	(0.022470)
13	Size Of Stack Reserve	(0.019694

- *The Nature of Files Tested again Model.*

Table 3 The Nature of Files Tested again Model

Sr No	File Name	Nature
1	msinfo.exe	legitimate
2	link.exe	malicious

The msinfo.exe is file present in C drive and it was legitimate according to model whereas the link.exe is malicious application file which was taken from internet to test and detected malicious [9].

V. CONCLUSIONS AND FUTURE WORK

This includes the selection of only important features to reduce the problem of latency. But if thecompatibility and computing power is quite high, thepower of system can make it possible to train even with all features of dataset. The successful integration of both signatures and PE headers withina unified system has demonstrated its effectiveness in detecting malware, showcasing a comprehensive approach that enhances accuracy in the detection process. This dual methodology effectively combines the strengths of signatures, which identifyknown malware patterns, with PE headers, providing crucial insights into the structural aspects of executable files [7].

Furthermore, this project establishes a robust foundation for potential extensions, particularly in the specialized domain of ransomware analysis and detection. By expanding upon the existingframework, future endeavors can concentrate on a thorough exploration of new patterns within headers associated with ransomware strains. This extension holds significant promise in tackling the dynamic challenges posed by ransomware within the cybersecurity landscape.

The proposed enhancement entails the developmentof advanced algorithms and methodologies capable of identifying novel patterns and distinctive characteristics linked to ransomware within PE headers. By proactively adapting to emerging threats, the system can evolve as a resilient defense mechanism against the ever-changing nature of ransomware attacks.

In essence, the incorporation of signatures and PE headers not only enhances accuracy in malware detection but also lays the groundwork for proactive measures against ransomware. This extension represents a forward-thinking approach to cybersecurity, emphasizing adaptability and patternrecognition to effectively counteract evolving threats in the digital landscape [8].

The findings of this classification study hold considerable potential for enhancing the effectiveness of anti-virus programs, thereby contributing to an improvement in the detectionrates of malware [10]. This research lays a solid foundation for future endeavors, where the amalgamation of diverse dynamic features within the contextual framework can be explored. Such a comprehensive approach is envisioned to furtherelevate the accuracy of malware detection systems, addressing the evolving landscape of cyber threats and establishing a more resilient defense against malicious activities.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAR615

REFERENCES

[1]. J. Raphel and P. Vinod, "Information theoretic method for classification of packed and encoded files", Procedings of the 8th International Conference on Security of Information and Networks - SIN ‟15, 2015.

[2]. M. Al-Asli and T. A. Ghaleb, "Review of Signature-based Techniques in Antivirus Products," 2019 International Conference on Computer and Information Sciences (ICCIS), Sakaka, Saudi Arabia, 2019.

[3]. M. Goyal and R. Kumar, "The Pipeline Process of Signature-based and Behavior-based Malware Detection," 2020 IEEE 5th International Conference on Computing Communication and Automation (ICCCA), Greater Noida, India, 2020.

[4]. S. Choudhary and A. Sharma, "Malware Detection & Classification using Machine Learning," 2020 International Conference on Emerging Trends in Communication, Control and Computing (ICONC3), Lakshmangarh, India, 2020.

[5]. M. Goyal and R. Kumar, "The Pipeline Process of Signature-based and Behavior-based Malware Detection," 2020 IEEE 5th International Conference on Computing Communication and Automation (ICCCA), Greater Noida, India, 2020.

[6]. A. M. Radwan, "Machine Learning Techniques to Detect Maliciousness of Portable Executable Files," 2019 International Conference on Promising Electronic Technologies (ICPET), Gaza, Palestine, 2019.

[7]. Z. Khorsand and A. Hamzeh, "A novel compression-based approach for malware detection using PE header," The 5th Conference on Information and Knowledge Technology, Shiraz,Iran, 2013.

[8]. L. El Neel, A. Copiaco, W. Obaid and H. Mukhtar, "Comparison of Feature Extraction and Classification Techniques of PE Malware," 2022 5th International Conference on Signal Processing and Information Security (ICSPIS), Dubai, United Arab Emirates, 2022.

[9]. Al-Khshali, Hasan H., Muhammad Ilyas and Osman Nuri Ucan. “Effect of PE File Header Features on Accuracy.” *2020*

[10]. KOLTER, J. Z. and M. A. MALOOF. Learning to detect malicious executables in the wild. In: Proceedings of the 2004 ACM SIGKDD international conference on Knowledge discovery and data mining. New York: ACM Press, 2004.