Towards Cyber Security: An Evaluation of methods for Malware Analysis

[1]Prudence Kadebu, [2]Kudakwashe Zvarevashe. [3]Addlight Mukwazvure, [4]Fungai N Mukora
[1234]Faculty of Computer Engineering, Informatics and Communications
University of Zimbabwe
630 Churchill Avenue, Mount Pleasant, Harare, Zimbabwe
[1]pkadebu@ceic.uz.ac.zw , [2]kzvarevashe@ ceic.uz.ac.zw, [3]amukwazvure@ ceic.uz.ac.zw ,
[4]fmukora@ ceic.uz.ac.zw

[5]Innocent Mapanga
Zimbabwe Information and Communication Technology, a Division of Zimbabwe Institute of Engineers
Conquenar House, P O Box 660, 256 Samora Machel Avenue, Eastlea, Harare, Zimbabwe
imapanga@gmail.com

[6]Tatenda T Gotora
Midlands State University
Senga, Gweru, Zimbabwe
tatenda.gotora88@gmail.com

Abstract

Cyber Security researchers, professionals and enthusiasts are currently faced with the challenge of coming up with advanced methods to deal with the bludgeoning of highly sophisticated Malware attacks on critical infrastructures housing sensitive information. The task of detecting, analysing and investigating the presence of Malware in a system is a daunting one, necessitating deep knowledge and skills in Malware Analysis. This analysis helps practitioners to uncover the behaviour and also the characteristics of Malware, thereby enabling Cyber Security practitioners and businesses to be proactive in building tools and techniques for defence against this type of Cyber-attacks. Malware authors are becoming more sophisticated in the methods they are using to evade detection. State of the art techniques are being applied to Malware Analysis with Machine Learning increasingly becoming popular while yielding very promising results. This paper proposes a hybrid framework for Unsupervised and Supervised Machine Learning methods for dynamic and static Malware analysis.

Keywords
Malware Analysis, Malicious Software, Malware detection, Machine Learning methods

1. Introduction
Malware which refers to any type of Malicious Software such as viruses, trojans, rootkits, trapdoors, adware, ransomware etc (Distler 2007), is among the myriad of threats that have become ubiquitous across the Cyberspace wreaking havoc to organisations worldwide. These have an adverse impact not only on business operations and service delivery but also the security of individuals' personal information. Those using Microsoft Windows operating system have been the most vulnerable. In addition, huge financial losses due to Malware, are recorded incessantly. In the past, cases of Ransomware attacks have mainly been popular in large organisations, but recently even individuals have lost data owing to such attacks. Merchants Point of Sale (POS) machines have been infected by various variants of POS Malware aimed at scraping credit card details of customers by using remote access tools and credential dumpers to access the system and inject Malware (Rautmare 2020). Malware can be introduced into the system through phishing attacks whereby an insider may naively open a link sent to their email with Malware embedded

therein. This is one very common way by which threat actors gain remote access to the systems. (Ucci, Aniello, and Baldoni 2019) asserts that as security technology advances, immediately an evasion follows, an indication that malicious actors are ever busy trying to bypass security mechanisms put in place to defend against their exploits. Malware Analysis involving detection, analysis, and investigation of malware in a system is the way Cyber Security practitioners gain traction ahead of threat actors while taking into cognisance the fact that the these criminals are also doing the same, hence the need for continuous improvement on methods for dealing with Malware.

Malware Analysis is divided into two main types that is Dynamic analysis and Static analysis (Oktavianto and Muhardianto 2013). These refer to methods aimed at profiling actions of the malware binary at runtime and those aimed at decompiling and analysing the internal structure of the binary itself, respectively (Sharif et al. 2008). In static analysis there is no execution of the program code, thus the Malware program code is inspected to identify hidden patterns and all behavioural scenarios of the Malware can be discovered (Chumachenko 2017). Techniques such as file fingerprinting are used with static analysis to ensure that the file does not change during the analysis. MD5sums and MD5deep are some examples used for this purpose (Kendall and Mcmillan 2007). The source code may not be available for static analysis, thus the analysis has to start with some reverse engineering tasks of disassembling and decompiling for static analysis to be conducted (Oktavianto and Muhardianto 2013). Dynamic analysis entails that, the behaviour of the malware is monitored during execution of the file. Malware authors make use of techniques such as Obfuscation (Bhojani 2014), Viral Polymorphism (Bashari Rad, Masrom, and Ibrahim 2012) Packers, anti-debugging, anti-disassembling, sandbox evasion, antivirus evasion etc. These hide the traces of their existence and behaviour patterns thus allowing them to evade the mechanisms meant to detect and thwart them. These deter the progress of static analysis which would normally assist in reverse engineering of Malware code. Thus, Dynamic analysis tends to be more superior to static analysis. Sandboxing is a technique used for dynamic analysis by executing files in a virtual environment for comprehensive analysis (Sethi et al. 2017). It is important so that suspicious files can be analysed without affecting the system. Cuckoo Sandbox is an example of such (Oktavianto and Muhardianto 2013). If a system is compromised, first action in line with Incident Response is isolation of the infected system. If it is a server, it should be taken off the network to reduce the risk of spreading the damage to other parts of the system. Analysis then follows, be it static or dynamic.

Most tools and techniques such as Antivirus Software tools are dependent upon Malware signatures thus any techniques that are not able to detect stealth attacks are futile in trying to combat them. Thanks to the developments in the area of Machine Learning which have seen an upsurge of techniques which are more potent in dealing with malware and have capacity to ward off the challenges posed by malware exploits. To this end this research work aims to apply Ensemble Machine Learning techniques which take advantage of the strengths of various Machine Learning techniques to improve on detection of Malware. The problem of Malware detection is a classification problem. This research is important as Malware has certainly become the most potent tool used by criminals to attack systems on the cyberspace (Lengyel et al. 2014). There is need to perform Malware Analysis to uncover Indicators of Compromise (IOC) which may be hidden, to understand the intent of the attacker, the level of damage and also to discover vulnerabilities within the system (Kendall and Mcmillan 2007). Considering the way present day Malware authors have become very advanced, using very sophisticated methods, it is consistent that techniques be devised that can thwart the most potent threats that exist especially where the malware is stealth making it difficult to detect IOCs. Once analysis has been done the output can be used to detect and subsequently mitigate the attack (Baker 2020).

The research paper is organised as follows: Section 2 focuses on the review of literature on Malware Analysis with more focus on Malware detection. The Methodology will come in Section 3 to describe the proposed work. Section 4 looks at the methods for Malware analysis followed in Section 4 by a proposed framework for Hybrid Malware Analysis using Supervised and Unsupervised Machine Learning, then in 5 by Results and Discussion. Finally, in Section 6, we conclude the work.

1.1 Objectives

The objective of the research is to uncover the methods used in Malware Analysis by surveying related literature. The research also aims to expose the various means by which the malware authors try to ensure the malware remains undetected. The authors also aim to assess the extent to which Machine Learning techniques are being applied in Malware Analysis and propose a hybrid framework for Unsupervised and Supervised Machine Learning for Malware analysis.

2. Literature Review

(Sharif et al. 2008) present a framework termed Eureka for enabling static Internet malware binaries analysis using a technique for de-obfuscation. Eureka takes the subject-packed malware binary and checks for evidence of tracing or debugging, in a Eureka managed VM environment. Next the malware binary is unpacked. and Eureka employs a course-grained execution tracker which applies a heuristic based as well as a binary n-gram statistical trigger that assesses when a stable state is reached in the unpacked process image. The reconstructed process image is passed on to the IDA-Pro disassembler which then disassembles it and conducts API resolution, a process that applies automated de-obfuscation to recover hidden API invocations enabling static analysis to proceed. The code image is then passed to the analysability metrics module which assesses the results of the static analysis. Finally, Eureka invokes the code graph and ontology generation module which gives the call graph and ontology labels then it also extracts, annotates and simplifies the structure. This work is important in dealing with the challenge of obfuscation which would normally hinder static analysis. However, it will not be able to assist towards classification of the detected Malware.

A dynamic malware analysis system they termed DRAKVUF designed to improve stealth by executing malware samples while leaving no trace in the analysis VMs is presented in (Lengyel et al. 2014). They propose novel techniques to eliminate blind-spots created by kernel-mode rootkits by extending the scope of monitoring to include kernel internal functions, and to monitor file-system accesses through the kernel's heap allocations. They demonstrate that DRAKVUF achieves a great improvement efficiency in performance and hardware resources utilisation as well as stealthy in the analysis into the behaviour of modern malware. DRAKVUF is implemented based upon four requirements of scalability, which allows for reduction in performance overhead in sample analysis while improving on the capacity to analyse large samples concurrently. The second requirement fidelity, ensures that enough data is collected at runtime while also allowing for resistance to against tampering for accurately analysis. The third requirement Stealth, ensures that DRAKVUF remains undetectable even within a monitored environment. Lastly, isolation means DRAKVUF should be protected against tampering by isolating it from the analysis VMs. The advantage of this Malware Analysis system is its ability to execute stealthily thus making it easy to detect Malware without the intruder being aware. However, like in (Lengyel et al. 2014), it cannot assist in classification of Malware.

Min and Varadharajan (2014) proposed a technique of 'feature-distributed malware' which applies a method for bypassing security defence mechanisms such as application whitelisting and also evades behavioural detection by anti-virus' software. The Malware is sophisticated in the way it distributes its features to several software components dynamically, thus making its detection very difficult. The paper aims to expose the risks associated with such advanced Malware and also suggests ways to circumvent them. The defence utilises digital certificates of software components to thwart any attempts to load malicious components.

A survey of Malware Analysis using Machine Learning is presented in (Ucci, Aniello, and Baldoni 2019) where they expose the need to make trade-offs between maintaining high accuracy and performance of malware analysis and supplying the required equipment, a study termed malware analysis economics. They also make a connection between feature extraction and execution time and the work goes on to assess whether desired features come from static or dynamic analysis, which has a bearing on execution time since unlike dynamic analysis, static analysis does not require running the samples. Although the research gives a survey of techniques, they also provide a qualitative, and simplified illustration of analysis leveraging on the introduced trade-off.

A two level framework for detecting (Macro) and classifying (Micro) Malware is proposed in (Sethi et al. 2017). The Cuckoo Sandbox is used in the work to generate reports for static and dynamic analysis from executing sample files in a virtual environment. Their framework includes a novel feature extraction module. Weka framework is used to build and test different Machine Learning models on a dataset of 220 samples of both malicious and benign files. J48, SMO and Random Forest yielded 100%, 99% and 97% detection rate respectively and 100%, 91% and 66.67% respectively for classification. The dataset used in the experiments is too sparse which may create bias.

A data Mining framework that uses automatically discovered patterns from data to detect new malicious binaries is proposed in (Schultz et al. 2001). They try to solve the problem of hand generated heuristics used by antivirus software which are usually costly and ineffective as virus signatures evolve. In contrast to the normal virus scanner technologies that have a virus signature detector and a classifier to detect new viruses based upon the detected signatures, the proposed framework trained data mining algorithms which improve on the reduction in false positive rates.

Table 1: Summary from Literature Review

Ref	Main Idea	Contribution	Techniques	Data	Performance	Remarks
Sharif et al. 2008	Eureka-framework for enabling static Internet malware binaries analysis	Course-grained execution tracker applying a heuristic based and a binary n-gram statistical trigger to estimate when to stop the malicious process image	API Resolution Techniques IDA-Pro disassembler De-obfuscation	Corpus of 1291 malware instances,479 malicious executables from spam traps, 435 malicious executables - honeynet.	97.7% Spam Malware Corpus unpacking 93.3% honey net Malware corpus unpacking Unpacking of 90 binaries/hr	Automated classification of Malware
Lengyel et al. 2014	DRAKVUF-dynamic MA system	Improves stealth by enforcing scalability, fidelity, stealth and isolation conserving resources	Hardware virtualization extensions and the Xen hypervisor	1000 samples from shadow server	Memory saving of 62.4%	Automated classification of Malware
(Ucci, Aniello, and Baldoni 2019)	A survey on MA through machine learning techniques	Novel concept of MA economics, malware anti-analysis techniques etc	A qualitative analysis	Processing one million malware per day	86% accuracy using 3 as minimum n-grams size	Tuning strategies to balance metrics such as accuracy and cost in designing MA environment.
Schultz et al. 2001	A data Mining framework for automatically detecting new malicious binaries	Method for detecting previously undetectable malicious executables	Naive Bayes, Multimodal-Naive Bayes, RIPPER Standard statistical cross-validation	Dataset of 4,266 programs 3,265 malicious binaries and 1,001 clean programs	Multi-Naive Bayes yielded highest Detection rate 97.76%	Extention of learning algorithms to make use of byte-sequences
(Sethi et al. 2017)	A framework for detecting and classifying Malware	Intelligent framework for dynamic and static Malware analysis of malware samples based on similarity	J48, SMO and Random Forest Cuckoo Sandbox for Malware Analysis	220 Samples of malicious and benign files	100%, 99% and 97% detection rate and 100%, 91% and 66.67% classification respectively	Dataset of 220 samples needs to be expanded

Most of the researches in literature were focusing on dynamic Malware Analysis which involves execution of the malicious code to understand the Malware. Table 1 gives a brief summary of some of the researches. The success of

Malware Analysis techniques depends upon how fast the Malware can be detected. Machine Learning techniques have become popularly applied in techniques for detecting Malware with very high performance.

Table 2: Malware Analysis in brief

Types of Malware Virus Worm Trojan Horse Backdoor Spyware Adware Rootkits		**Types of Malware Analysis** Static Analysis Dynamic Analysis Hybrid	
	**Tools for Analysis** Sandboxes Sniffers Reverse code Engineers Disassemblers Debuggers Decompilers Network analyzers		**Malware Defense** Firewall Antivirus Software Secure Web Gateways Sandboxing
Malware Attacks/Acquisition Phishing emails Social Engineering Infected attachments Infected websites malicious browser plugin USB infection		**Incident Response** Isolation of compromised unit Preparation, Identification, Containment, Eradication Recovery, Lessons learned.	

3. Methods

This section discusses the two common techniques for Malware Analysis that is Sandboxing and Machine Learning. The Cuckoo Sandbox as a case study, will be explored to understand what information can be realized regarding the malware behaviour and attributes. Machine learning has become very popular in aiding Malware detection while reducing false positives.

3.1 Cuckoo Sandbox

Focus is given to the Cuckoo Sandbox. Cuckoo, developed by Claudio "nex" Guarnieri is an Open source Sandboxing tool that is used for dynamic Malware Analysis (Oktavianto and Muhardianto 2013). It implements a virtual Malware Analysis Lab which helps to analyse suspicious files without risking having malware replicating and affecting other parts of the system. Cuckoo automatically executes files, analyses them and collects comprehensive results about the behaviour of the malware as it runs in an isolated virtual environment. A file with malware can be submitted into the Cuckoo virtual environment for analysis. After submission, Cuckoo creates several subfolders for the analysis including *memory.dmp*, file containing the full memory of the analysis machine and is useful in Memory forensic analysis. In the subfolder, there is also the *dump.pcap*, which is the network dump file which can be further analysed with a network packet tracer like Wireshark in the case of Cuckoo. Additional tools come in handy for various processes and these may also be combined. For instance, an Advanced Persistent Attack (APT) is analysed using Cuckoo sandbox, Volatility and another tool Yara for a more enhanced analysis.

Table 3: Cuckoo Sandbox elements

Item	Description
Setup	• Management software in the host machine • Several virtual machines for analysis
Types of files	• DLL files, PDF, URLs, PHP scripts • Microsoft Office documents • Windows executables
Types of results	• win32 API calls performed by all processes spawned by the malware • Files being manipulated by the malware during its execution • Memory dumps of the malware processes and full memory dumps of the machines • Network traffic trace • Screenshots taken during the execution of the malware
Processing Modules	• AnalysisInfo which shows basic information such as timestamps, Version of Cuckoo • BehaviorAnalysis using behavioral logs to performs for trasnformations and interpretations • Debug module for errors and the analysis.log • Dropped module details of files dropped by the malware and Cuckoo • NetworkAnalysis to extract network information, eg DNS traffic, , IP addresses, etc • StaticAnalysis module performs some static analysis • Strings module extracting strings from analyzer binary • TargetInfo information such as hashes, on the output file • VirusTotal gives AntiVirus signatures of the output file

Table 4: Results of Analysis of executable file

VirusTotal

38/42 (collapse)

Antivirus	Result
nProtect	Win32.Sality.E
CAT-QuickHeal	None
K7AntiVirus	Virus
TheHacker	W32/Sality(rp).I
VirusBuster	Win32.Sality.L
NOD32	Win32/Sality.NAE
F-Prot	W32/Sality.K
Symantec	W32.HLLP.Sality.O
Norman	W32/Sality.N
ByteHero	None
TrendMicro-HouseCall	PE_SALITY.AE
Avast	Win32:Sality-U
eSafe	Win32.Sality.gen
ClamAV	W32.Sality.N
Kaspersky	Virus.Win32.Sality.I

In the figure below, an executable file was identified to have a virus known as Sality. Static analysis performed using Cuckoo shows that some libraries are imported form KERNEL32.DLL which can then add an entry to the registry for instance:

HKEY_LOCAL_MACHINE\Software\ Microsoft\Windows\CurrentVersion\Run

This defines programs that can run at startup thus the virus will use this to try and maintain access to the victim's machine this way. The malware can imitate legitimate software. Its activities are shown in the processes section. Malware executable files can be viewed by means of a disassembler application such as Radare, a reverse engineering tool. Besides disassembling, it is also widely used in debugging, analyzing, and manipulating binary files. Bokken, a frontend application can also be used to support the process. It shows the flowgraph of the binaries after disassembling. Hexdump tab and file info tab can be viewed in Bokken. At this point dynamic analysis can be done with Cuckoo Sandbox to give details of the behaviour of the Malware, in terms of what it does in the system and what changes it

makes. Cuckoo uses various reporting formats, such as human-readable format, MAEC (Malware Attribute Enumeration and Characterization) format and can also export data reports to other formats. Once reports have been produced from a sandboxing tool such as Cuckoo in a human readable format in natural language, it becomes easier to augment the results by applying Machine Learning techniques to the process to improve on the quality of Malware Analysis for further instances.

3.2 Machine Learning

As agreed by many researchers to be the definition attributed to the pioneer Arthur Samuel, Machine Learning is the "field of study that gives computers the ability to learn without being explicitly programmed" (Lee et .al 2017; Muñoz 2012). Machine Learning aids in detection of Malware by applying techniques that improve performance by reducing false positives (Gavriluţ et al. 2009). A typical Malware detection algorithm is shown in Figure 2. This is supervised learning applied to detection of Malware, a classification problem. Examples of benign and malicious executables are passed to train a Machine Learning Algorithm to produce a predictive model to use in the second leg to detect new instances of unknown executables. This can now be categorized as either benign or malicious with the aim of achieving lowest false positive rate. Malware detection techniques are either signature based or anomaly based. This applies to signature-based techniques for Malware detection which depend upon the known signatures of the Malware. If a know signature is detected then the executable is labeled as malicious.

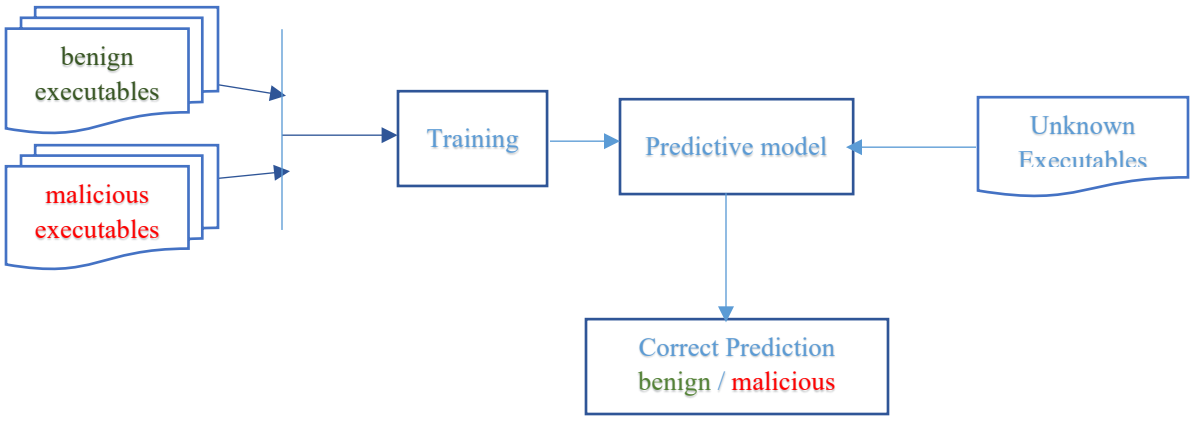

Figure 2: Supervised Machine Learning Algorithm for Malware Detection

4 A framework for Hybrid Malware Analysis using Supervised and Unsupervised Machine Learning

Static Malware Analysis works well with known malware signatures. However, the challenge comes in the case of Zero-day Malware detection where there are no known signatures. Anomaly based methods become more potent as they depend upon analysing and profiling the behaviour observed as anomalous which is malicious or normal with statistical or unsupervised learning methods. A hybrid Malware Analysis framework shown in Figure 3 which incorporates both dynamic and static analysis applying a stacked model for Unsupervised and Supervised Machine Learning is proposed.

Packed/ encrypted executables
Malware authors use techniques such as wrappers, encryption and compression for obfuscation to hide Malware code within executable files so that it evades detection by antivirus software and suppress any reverse engineering attempt by the Malware analysis. In the first part of the framework executable binaries are presented for analysis in any or a combination of the forms highlighted above. Both dynamic and static analysis is difficult to perform under the circumstances.

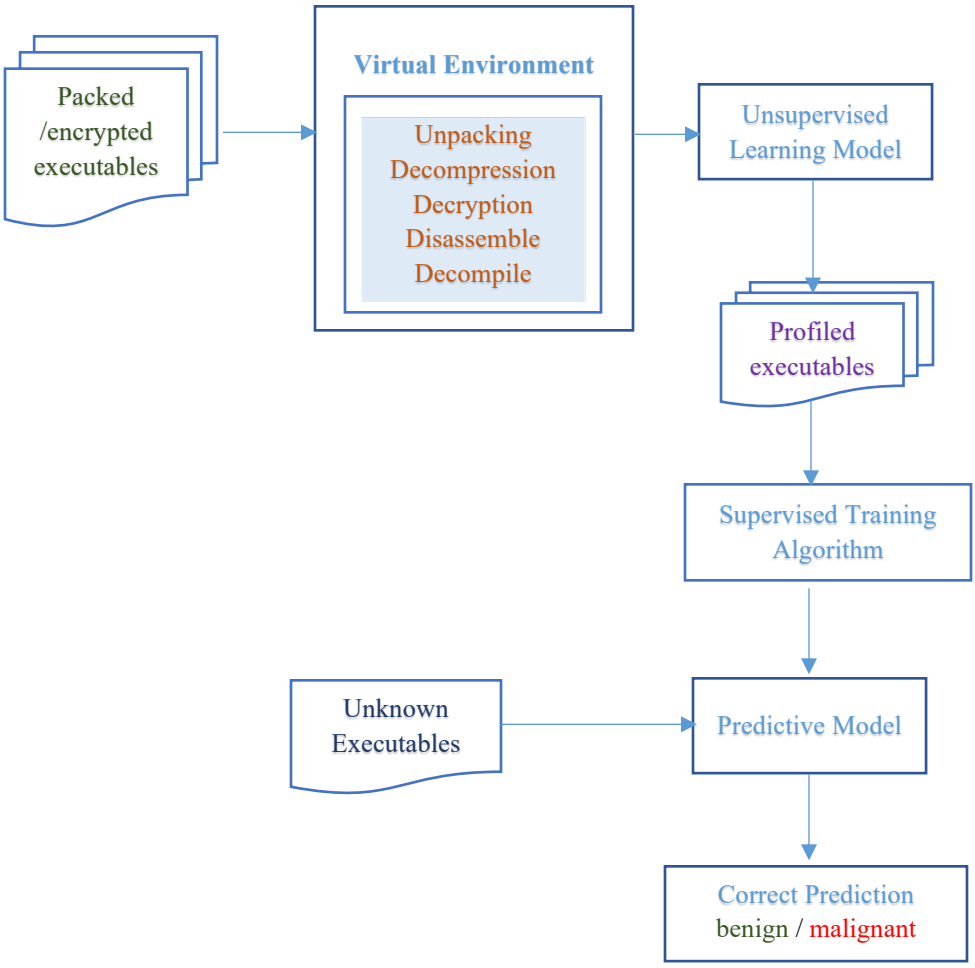

Figure 3: Hybrid framework for Malware Analysis

Virtual Environment
The packed executables have to go through the unpacking or de-obfuscation process. This decompresses the executables and also decrypts in the case of encrypted executables. Depending on how the malware has been embedded in the executable, other processes that will happen within the Virtual Environment include disassembling and decoupling to expose the code and enable dynamic analysis.

Unsupervised learning model
An unsupervised Learning algorithm is then applied to profile the behaviour of the Malware. This process applies clustering methods which are able to determine behaviour that is anomalous and that which is normal. It then categorises the binaries according to the identified patterns of behaviour. The output are the profiled executables.

Supervised Training Algorithm
A supervised training is applied with the profiled executables as the training data. This is part of the static analysis process which involves analysing the profiled executable files without executing them. A supervised Machine Learning model is applied to learn the profiles which are anomalous or normal. Thus, it extracts useful information from the binary to classify it as either benign or malignant.

Predictive Model
Now new instances of unpacked executables can then be classified as either benign or malicious by feeding it into the predictive model. The model is expected to discover the patterns exposed by the executable binaries from what it has learnt in the training with the profiled executables.

5. Results and Discussion

The work presented recommends a method catering for both dynamic and static Malware analysis of executable binaries using Machine Learning. The hybrid framework ideally implements a stacked unsupervised and supervised Machine Learning model which seeks to profile the files based on behaviour that is deemed as normal and that which is anomalous in the presence of Malware. Unsupervised Learning algorithm supports dynamic analysis as executable binaries are unpacked making it possible to create profiles for the files. This is important as Malware behaviour can be identified instead of dependency upon signatures alone which is not adequate to detect Zero-day Malware instances. The output then passes on to the supervised Learning algorithm which supports static analysis whereby the profiled files become training input to build a predictive model to aid in the detection of Malware. This combination of the two types of Malware analysis is expected to yield better results towards protection of information systems. This may help to detect stealth malware attacks and reduce the risk of extensive damage to the systems.

5.1 Proposed Improvements and Validation

The proposed hybrid framework is the earliest of efforts on the roadmap by the researchers to contribute towards this important area of Cyber Security. The current work will culminate in the implementation and testing of supervised and unsupervised algorithms on a Malware dataset. In addition to simple Machine Learning methods, Ensemble Machine Learning and Deep Learning methods shall also be tested which are expected to yield more superior results than previous methods highlighted in literature.

6. Conclusion

The research work revealed that, as techniques for introducing Malware in Information Systems and evading detection continue to evolve at a fast pace, more effort is being applied towards improving defense mechanisms and safeguards focused on preserving the security of the systems. More stealth detection techniques are being developed to ensure the Malware analysis does not leave a trace of the analysis performed in a virtual environment it does not give the Malware author an advantage. Dynamic and static Malware Analysis techniques are commonly applied with Hybrid techniques yielding better results by harnessing the strengths of the two types of analysis. Malware authors aim to bypass the defenses by applying techniques such as obfuscation, anti-debugging, anti-disassembling, sandbox evasion, antivirus evasion etc. Sandboxing is key in Malware analysis so as to prevent risks of executing infected files in the system. A hybrid framework for Unsupervised and Supervised Machine Learning was proposed. This framework enables dynamic analysis to be performed in a virtual environment Implementation and validation shall be done in the future works.

References

Baker, Kurt. 2020. “Malware Analysis.” https://www.crowdstrike.com/epp-101/malware-analysis/ (October 15, 2020).

Bashari Rad, Babak, Maslin Masrom, and Suhaimi Ibrahim. 2012. “Camouflage In Malware: From Encryption To Metamorphism.” *International Journal Of Computer Science And Network Security (IJCSNS)* 12(8): 74–83. http://paper.ijcsns.org/07_book/201208/20120813.pdf.

Bhojani, Nirav. 2014. “Malware Analysis.”

Chumachenko, Kateryna. 2017. *Machine Learning Methods for Malware Detection and Classification*. https://core.ac.uk/download/pdf/80994982.pdf (October 15, 2020).

Distler, Dennis. 2007. “Maware Analysis: An Introduction.” *SANS Institute*.

Gavriluţ, Dragoş, Mihai Cimpoeşu, Dan Anton, and Liviu Ciortuz. 2009. “Malware Detection Using Machine Learning.” In *Proceedings of the International Multiconference on Computer Science and Information Technology, IMCSIT ’09*, , 735–41.

Kendall, Kris, and Chad Mcmillan. 2007. *Practical Malware Analysis*.

Lee, Aaron, Paul Taylor, and Jayashree Kalpathy-Cramer. 2017. “Machine Learning Has Arrived!” *Ophthalmology* 124(12): 1726–28. http://dx.doi.org/10.1016/j.ophtha.2017.08.046 (November 18, 2020).

Lengyel, Tamas K. et al. 2014. “Scalability, Fidelity and Stealth in the DRAKVUF Dynamic Malware Analysis System.” *ACM International Conference Proceeding Series* 2014-Decem(December): 386–95.

Min, Byungho, and Vijay Varadharajan. 2014. “Feature-Distributed Malware Attack: Risk and Defence.” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in*

Bioinformatics) 8713 LNCS(PART 2): 457–74.
Muñoz, A. 2012. "Machine Learning and Optimization."
Oktavianto, Digit, and Iqbal Muhardianto. 2013. *Cuckoo Malware Analysis Analyze Malware Using Cuckoo Sandbox*. Packt Publishing.
Rautmare, Chinmay. 2020. "Visa Alert: POS Malware Attacks Persist - BankInfoSecurity." https://www.bankinfosecurity.com/visa-alert-pos-malware-attacks-persist-a-15126 (October 15, 2020).
Schultz, Matthew G., Eleazar Eskin, Erez Zadok, and Salvatore J. Stolfo. 2001. "Data Mining Methods for Detection of New Malicious Executables." *Proceedings of the IEEE Computer Society Symposium on Research in Security and Privacy*: 38–49.
Sethi, Kamalakanta, Shankar Chaudhary, Bata Tripathy, and Padmalochan Bera. 2017. "A Novel Malware Analysis for Malware Detection and Classification Using Machine Learning Algorithms." In , 107–13.
Sharif, Monirul et al. 2008. "Eureka: A Framework for Enabling Static Malware Analysis." *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 5283 LNCS: 481–500.
Ucci, Daniele, Leonardo Aniello, and Roberto Baldoni. 2019. "Survey of Machine Learning Techniques for Malware Analysis." *Computers and Security* 81: 123–47. https://doi.org/10.1016/j.cose.2018.11.001.

Biography

Prudence Kadebu is the head of Computer Engineering at the University of Zimbabwe. She is a Software Engineer experienced in team leading, Software Engineering, lecturing at various levels, Research and Development, Curriculum Development and Review, Security, Software Quality Assurance, and is an Artificial Intelligence and Machine Learning enthusiast. She has more than 10 years University experience. Prudence is passionate about empowering women and girls in STEM, career guidance, extending digital literacy to the less privileged communities and representing women in various platforms. She is also involved in projects on Solar PV aimed at capacitating schools to use ICT equipment donated by the government in the remote parts of Zimbabwe. Prudence advocates for promoting knowledge transfer between Universities and Industry to foster economic growth. She is also a professional member of Zimbabwe Institute of Engineers.

Kudakwashe Zvarevashe is the head of Analytics and Informatics at the University of Zimbabwe. Kudakwashe holds an MTech in Information Technology from Jawaharlal Nehru Technological University (JNTUH, India) and a BSc in Information Systems from Midlands State University (MSU, Zimbabwe). He has more than 9 years University experience in Teaching, Research and Administration. His research interests are in Cyber Security, Computer Vision, Natural Language Processing, Cloud Computing and Big Data. He is also a professional member of Zimbabwe Institute of Engineers.

Addlight Mukwazvure is a lecturer in Computer Engineering Department at the University of Zimbabwe. She holds an MTech in Computer Science from Jawaharlal Nehru Technological University (JNTUH, India) and a BSc in Computer Science from National University of Science and Technology (NUST, Zimbabwe). He has more than 9 years University experience in Teaching, Research and student mentorship. Her research interests are in Cyber Security, wireless Sensor Networks, Cloud Computing and Big Data. She is also a professional member of Zimbabwe Institute of Engineers.

Fungai N Mukora is a member of the Computer Engineering Department at the University of Zimbabwe.. She holds a Master of Science Degree in Computer Science from the University of Zimbabwe and is a candidate for the PhD Information Systems with the UKZN. Fungai teaches Hardware Engineering, with a keen interest in Logic Design and Circuits, Number Systems and Digital Computer Fundamentals. Her research areas include Digital Transformation, ICT adoption with emphasis on marginalized groups and communities.

Innocent Mapanga is a member of the Zimbabwe Information and Communication Technology a division of the Zimbabwe Institute of Engineers. He is an IT infrastructure Security Engineer who holds **a** MTech Computer Science & Engineering from, DTU, India, a BSc (Hons) Computer Science from Bindura University of Science Education (BUSE, Zimbabwe), PhD candidate. He is a seasoned IT infrastructure and Security professional with experience in designing and implementing solutions in high availability environments. He is well versed in High End Servers and Storage and has broad Knowledge in Identity and Access Management, Identity Governance Lifecycle, Security Intelligence Platforms, SOAR, SOC, Incidents Response Platforms, Security Frameworks, Vulnerability and Penetration Testing and adept at delivering strong risk management practices. Innocent also has exceptional

knowledge of desktop, server and internet levels of network security and in incident response and data breach investigations, strategic information security services, and threat intelligence analytics.

Tatenda T Gotora is a lecturer in Computer Science at the Midlands State University in Zimbabwe. He holds an MTech in Software Engineering from Jawaharlal Nehru Technological University (JNTUH, India) and a BSc in Computer Science from Midlands State University (MSU, Zimbabwe). He has more than 9 years University experience in Teaching, Research and student mentorship. His research interests are in Cyber Security, Computer Networks and Telecommunication, IOT, Cloud Computing and Big Data.