Unveiling the Dynamic Landscape of Malware Sandboxing: A Comprehensive Review

Elhaam Debas[1], Norah Alhumam[2], Khaled Riad[3]
College of Computer Science and Information Technology,
King Faisal University, Al Hassa 31982, Saudi Arabia[1,2]
Computer Science Department-College of Computer Sciences & Information Technology,
King Faisal University, Al-Ahsa 31982, Saudi Arabia[3]
Mathematics Department-Faculty of Science, Zagazig University, Zagazig 44519, Egypt[3]

***Abstract*—In contemporary times, the landscape of malware analysis has advanced into an era of sophisticated threat detection. Today's malware sandboxes conduct rudimentary analyses and have evolved to incorporate cutting-edge artificial intelligence and machine learning capabilities. These advancements empower them to discern subtle anomalies and recognize emerging threats with a heightened level of accuracy. Moreover, malware sandboxes have adeptly adapted to counteract evasion tactics, creating a more realistic and challenging environment for malicious entities attempting to detect and evade analysis. This paper delves into the maturation of malware sandbox technology, tracing its progression from basic analysis to the intricate realm of advanced threat hunting. At the core of this evolution is the instrumental role played by malware sandboxes in providing a secure and dynamic environment for the in-depth examination of malicious code, contributing significantly to the ongoing battle against evolving cyber threats. In addressing the ongoing challenges of evasive malware detection, the focus lies on advancing detection mechanisms, leveraging machine learning models, and evolving malware sandboxes to create adaptive environments. Future efforts should prioritize the creation of comprehensive datasets, distinguish between legitimate and malicious evasion techniques, enhance detection of unknown tactics, optimize execution environments, and enable adaptability to zero-day malware through efficient learning mechanisms, thereby fortifying cybersecurity defences against emerging threats.**

***Keywords*—Malware analysis; threat hunting; security operations; machine learning; cutting-edge AI; sandboxing**

Abbreviations The following abbreviations are used in this review:

SLR	Systematic Literature Review
PRISMA	Preferred Reporting Items for Systematic Reviews and Meta-Analyses
QCQP	Quadratically Constrained Quadratic Program
HCP	Honeypot-based Collaborative Protection
IoT	Internet of Things
CERTS	Computer Emergency Response Teams
UPX	Ultimate Packer for Executables
Process	Monitor Procmon
UBER	User Behavior Emulator
SCADA	Supervisory Control And Data Acquisition
ICS	Industrial Control Systems
UI	User Interface
SVM	Support Vector Machines
DT	Decision Trees
CNN	Convolutional Neural Networks

I. Introduction

Malware sandbox evaluation involves the use of controlled environments, known as sandboxes, where malware samples can be executed and analyzed safely. These sandboxes provide a secure and isolated space where the malware's activities can be closely observed and monitored without posing any risk to real computer systems and networks [1]. During the evaluation process, security experts closely monitor various aspects of the malware's behavior. This includes analyzing its network communications, such as the domains it connects to, the protocols it uses, and the data it exchanges. By examining these network interactions, security professionals can identify any suspicious or malicious activities, such as attempts to communicate with known command-and-control servers or transfer sensitive data. The sandbox evaluation also focuses on understanding the malware's system interactions. This involves studying how the malware interacts with the host system's files, processes, and registry entries. By analyzing these interactions, security experts can identify any attempts made by the malware to modify system settings, exploit vulnerabilities, or compromise the integrity of the host system.

Another important aspect of malware sandbox evaluation is observing the malware's evasion techniques. Malware often employs various tactics to avoid detection by security tools and antivirus software. By running the malware in a sandbox, security professionals can closely monitor its attempts to evade detection, such as using encryption, obfuscation, or anti-analysis techniques [2]. This knowledge helps in refining detection methods and developing countermeasures to effectively identify and mitigate similar threats in the future. The data gathered from sandbox evaluations is carefully examined to gain deeper insights into the malware's operation and communication patterns. Security experts analyze this data to understand the malware's capabilities, goals, and potential effects on a system. This information is crucial in determining the malware's objective, which could range from data theft and unauthorized system access to launching further attacks.

Furthermore, the insights gained from malware sandbox evaluation contribute to the development of efficient detection and preventive systems. By understanding the behaviour and techniques employed by malware, security professionals can create more effective defence mechanisms. This includes enhancing threat detection tools, improving response strategies, and developing mitigation techniques to protect against similar dangers in the future [3]. By staying up to date on the

newest malware behaviours and capabilities, security experts can proactively safeguard computer systems and networks. This proactive approach involves continuous research and learning to adapt sandbox evaluation techniques to the evolving landscape of cyber threats. By staying connected with security communities and sharing information, security professionals can collaborate to develop stronger defence mechanisms and respond effectively to emerging malware behaviours.

In summary, malware sandbox evaluation is a crucial procedure in cybersecurity. It allows security professionals to closely monitor and analyze the behaviour of malware in a controlled environment, enabling them to understand its capabilities, identify potential risks, and develop effective defence strategies [4]. By staying informed about the latest malware behaviour and continuously improving evaluation techniques, security experts can proactively protect computer systems and networks, creating a safer digital environment for individuals and organizations.

This paper answers the following questions:

- What are the different types of malware?
- What are the types of malware sandboxing techniques?
- What are the challenges and limitations in malware detection?

The paper aims to underscore the crucial role of malware sandboxes in offering a secure and dynamic environment for thorough analysis of malicious code. It contributes significantly to combating evolving cyber threats, particularly addressing the challenges of evasive malware detection. The focus is on advancing detection mechanisms, leveraging machine learning models, and evolving malware sandboxes to create adaptive environments. It is suggested that future efforts prioritize the creation of comprehensive datasets, distinguish between legitimate and malicious evasion techniques, enhance detection of unknown tactics, optimize execution environments, and enable adaptability to zero-day malware through efficient learning mechanisms, thereby fortifying cybersecurity defences against emerging threats. The motivation behind this review paper is to offer a comprehensive understanding of the current state of malware sandboxing technology and its potential for future development. The contribution lies in providing insights into the evolution of malware sandboxing technology, its current state, and prospects. This paper aims to provide valuable insights for researchers, practitioners, and policymakers in the cybersecurity field.

The rest of the paper is organized as follows. In Section II, the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) flow diagram is presented for the selection of research papers related to the study. The diagram depicted below illustrates the systematic approach employed to identify relevant literature for analysis. Section III presents an overview of the Malware Sandbox. Section IV delves into the systematic literature review on Malware Sandbox Evaluation, discussing existing research and findings in this field. In Section V, future directions are summarized, and ideas for further exploration and improvement in malware sandbox evaluation are proposed. Finally, Section V concludes this study by summarizing the key findings and emphasizing the importance of ongoing research and advancements in this area to combat the ever-evolving landscape of cyber threats.

II. Research Methodology

A Systematic Literature Review (SLR) was conducted following established guidelines, which serve as a valuable tool to ensure a structured data collection process that progresses through three key stages [5]. During the identification stage, comprehensive searches were conducted in well-known academic databases, including Google Scholar, the Saudi Digital Library, and ScienceDirect. The following search terms were used: 'Malware Sandbox Evolution' or 'Advanced Threat Hunting' or 'Malware Analysis' and 'Threat Intelligence' or 'Cybersecurity' or 'Security Operations' or 'Malware Detection'. The search scope was limited to peer-reviewed articles published between 2018 and 2023. Inclusion criteria were studies that explored topics related to the evolution of malware sandboxes, advanced threat-hunting techniques, malware analysis, and their intersections with threat intelligence, cybersecurity, security operations, and malware detection.

A pool of 28 articles was identified and selected for this literature review using the PRISMA methodology, as depicted in Fig. 1. This figure illustrates the systematic approach employed. The identification stage marks the initial collection of articles for review. During this phase, a significant number of records were excluded due to various reasons, such as duplicates and ineligibility, as determined by Zotero, an automation tool. Subsequently, the screening stage involved a meticulous review of 3008 articles based on their titles and abstracts, resulting in the exclusion of 2255 articles that did not closely align with the criteria. During the eligibility step, articles meeting the predefined criteria were included. Finally, in the inclusion stage, the final set of 28 articles for the systematic review was selected, with 149 articles excluded due to reasons such as language barriers (e.g., Russian, Chinese), limited access to records, or being outside the defined time frame. This process resulted in the final inclusion of 28 articles.

III. Malware Sandbox Overview

A. VirtualBox and Sandbox

In the computer world, a sandbox and a virtual box have different functions. VirtualBox is not inherently a sandbox in the traditional cybersecurity sense. VirtualBox is a virtualization platform that lets you make and run virtual machines on a host system, see Fig. 2(A). While it shares some similarities with sandbox environments, its primary purpose is to enable the operation of numerous operating systems on a single physical device rather than serving as a dedicated security sandbox [6]. A security sandbox typically refers to an isolated and controlled environment where untrusted or potentially malicious code can be executed and analyzed without threatening the actual system, see Fig. 2(B). Sandboxes are commonly used in cybersecurity for malware analysis, software testing, and providing a secure space for running untrusted applications. However, VirtualBox can be used as part of a security testing or research environment. For example, you might use VirtualBox to set up isolated virtual machines for malware analysis or to test software behaviour in different operating system environments [7]. VirtualBox helps create

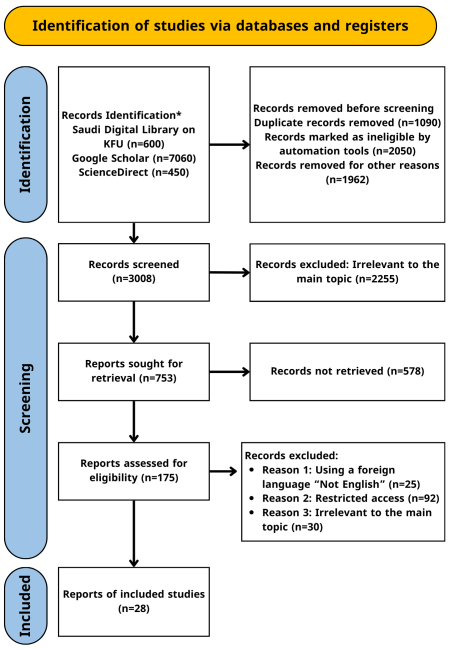

Fig. 1. Research methodology using PRISMA.

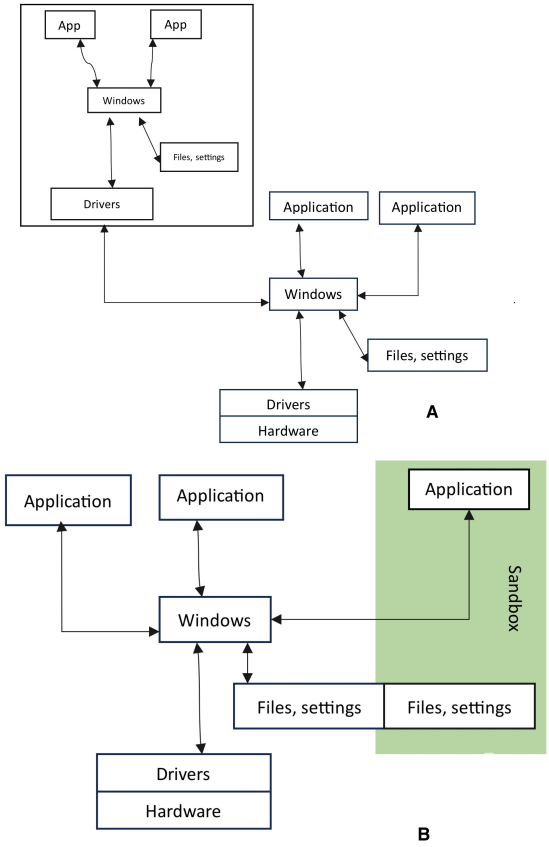

Fig. 2. VirtualBox(A) and sandbox(B) conceptual view.

controlled environments for specific purposes in such cases, but it is not a dedicated security sandbox solution [8]. If your goal is specifically to set up a security sandbox, you might want to consider specialized sandboxing solutions designed for security testing and analysis.

B. Techniques for Analyzing Malware

1) Static Analysis: Static analysis entails scrutinizing the structure and code of malware without executing it, providing vital insights into its potential impact. Standard static analysis methods include: Disassembling: Translation of malware's binary code into assembly language for understanding its functionality. Decompiling: Reverse engineering compiled code into a high-level programming language to unveil the malware's purpose. Debugging: Analysis of code in a debugging environment to pinpoint vulnerabilities and potential attack vectors.

2) Dynamic Analysis: Dynamic malware analysis observes malware behavior in a controlled environment like a virtual machine. Executing the malware in isolation allows for monitoring its activity, understanding its capabilities, and assessing potential impacts. This technique helps identify functions like spreading mechanisms.

3) Hybrid Analysis: Hybrid analysis integrates the strengths of both static and dynamic approaches. It begins with static analysis, extracting information such as embedded files and code obfuscation. Subsequently, dynamic analysis in a controlled environment, like a sandbox, helps observe the malware's behavior and uncover malicious activities not evident during static analysis. These comprehensive malware analysis techniques and tools see Table I, whether static, dynamic, or hybrid, are indispensable for cybersecurity professionals in comprehending, mitigating, and responding to ever-evolving cyber threats [9].

IV. Related Work

In this section, a comprehensive overview of significant research findings and insights on malware and sandboxes is provided. Various methodologies and approaches that researchers have employed to investigate the potential benefits, challenges, and applications of malware and sandboxes are discussed in Tables VI and VII.

Elhanashi et al. [1] Unveiled a novel anomaly-based intrusion detection system using machine learning on a challenging dataset. By employing feature selection and stacked autoencoders. Using three classifiers GaussianNB, Multi-layer, and Random forest achieved a remarkable accuracy equal to 88%, 99.3%, and 99.6% respectively, which outperformed existing methodology. This approach discovered the way for robust and efficient cyber defence against diverse attacks. This research opens doors for further exploration, inviting an investigation into advanced techniques like convolutional neural networks and dataset-specific parameter optimization.

Sethi et al. [8] introduced a novel malware analysis framework utilizing machine learning for detection and classification. The two-level classifier distinguishes between benign and malicious files, employing Cuckoo Sandbox to generate static and dynamic analysis reports in a virtual environment. Cuckoo Sandbox is an open-source automated malware analysis system, that explains its functioning in a virtual environment to monitor and generate reports on program behavior. The framework incorporates a feature extraction module based

TABLE I. WINDOWS MALWARE STATIC AND DYNAMIC ANALYSIS TOOLS

Type of Tool	Tool Name	Description
Static [10]	BinText	A mechanism for extracting binary data to text that outputs resource strings, Unicode, and ASCII text in simple plain text.
	TrID	uses binary signatures to identify file types without the need for set rules.
	Ultimate Packer for Executables (UPX)	The UCL data compression algorithm is used in this freeware and open-source executable packer.
	XORSearch	An open-source program that uses brute force to look for strings encoded with XOR, ROL, ROT, or SHIFT in a file.
	Exeinfo PE	Verifies .exe files by giving the precise size and malware entry point information.
Dynamic [10]	FakeNet	creates the illusion of a phony network for malware operating in a virtual machine.
	Process Monitor (Procmon)	Windows Sysinternals Freeware monitors and displays real-time file system activity.
	ProcDOT	uses the GraphViz suite to create a graph by processing the log files from Procmon and PCAP.
	Wireshark	examines various network protocols' structural analysis to show how encapsulation works.
	Process Explorer	Freeware system monitors and task managers offer Windows Task Manager's functionality for gathering data about active processes.
	RegShot	Open-source registry Using a quick snapshot of the system registry, the compare utility compares the registry after the malware has been executed.

on static, behavioural, and network analysis using Cuckoo Sandbox-generated information. Utilizing the Weka Framework, machine learning models are developed with training datasets, demonstrating high detection and classification rates across various machine learning algorithms, as evidenced by experimental results presented in the document. Also, the research paper offered a comprehensive overview of dynamic malware analysis, covering the techniques and tools involved in the process and detailed dynamic analysis, which entails executing a program in a controlled environment to observe its behaviour and detect any malicious activities. Alongside this, it provided an in-depth examination of recent malware samples, highlighted features, and elucidated how malware employs anti-analysis techniques, code obfuscation, and packers to enhance evasion and underscored the significance of dynamic malware analysis in identifying and analyzing unknown malware, encouraging further exploration in this domain.

BELEA et al. [9] documented malware analysis techniques, specifically static, dynamic, and hybrid analysis. It discusses the importance of analyzing malware to understand its behaviour and capabilities, and how this analysis can be used to develop effective countermeasures and strengthen cybersecurity defences. The document also mentioned other techniques used in malware analysis, such as reverse engineering, sandboxing, memory analysis, network analysis, and behavioural analysis. It emphasized the need for different tools and approaches to analyze the components of a PE file format, which is commonly used for distributing malware targeting Windows computers. The document concluded by stating that the choice of analytical method depends on the specific goals and expertise of the analyst involved.

UPPIN [10] identified the problem statement and categorized malware into four groups based on their architecture at the time of infection. The focus was on the dynamic analysis of Windows-based malware, utilizing automated sandboxing and reviewing relevant literature. The paper presented dynamic and static tools employed in Windows malware analysis, along with a detailed description. Steps for analyzing malware in a secure environment were outlined, using the LockerGoga ransomware as a specific example. The network's performance during the infection was documented, and a method based on virtual time control mechanics was suggested. This method involved the use of a modified Xen hypervisor to accelerate the sandbox's operation. The paper concluded by underscoring the importance of maintaining accessible, usable, and malware-free data and records in a system. A list of various malware mitigation strategies was provided, emphasizing the necessity for robust and effective mitigation approaches. The authors suggested that the techniques presented in their work would significantly contribute to cyber-cleaning efforts and enhance the effectiveness of information preservation policies against malware.

Kamal et al. [11] documented a user-friendly model for ransomware analysis using sandboxing. It discusses the challenges of analyzing ransomware and the difficulty of interpreting the results generated by sandbox environments. The goal of the suggested model was to offer a simple user experience for uploading ransomware files for examination and producing reports that are brief enough for average computer users to understand. Built on the Cuckoo sandbox environment, the model has been assessed through a user survey, resulting in 92% positive feedback regarding its usability.

Yong et al. [12] documented a study conducted on the practice of malware analysis. It included interviews with participants who work in the field of malware analysis and provided insights into their daily job tasks, experience, and the tools and techniques they use in their analysis process. The study also explored topics such as malware sources, analysis workflow, dynamic analysis system configuration, and the evolution of the analysis process over time. Malware analysis practitioners identified six critical decisions when configuring their dynamic analysis systems. These choices encompass considerations related to the implementation approach, selection of a virtual analysis platform, setup of the analysis environment, network communication management, determination of execution time parameters, and adopting techniques to counter evasive tactics employed by certain malware strains. Participants carefully navigate these decisions to ensure the efficacy and robustness of their dynamic analysis systems in comprehensively understanding and countering evolving malware threats.

Sikdar et al. [13] documented a game theoretic model of malware protection using the sandbox method. The authors created methods and recommendations to raise the standard for sandbox analysis. In a two-player game, where the anti-malware commits to a strategy of creating sandbox environments and the malware reacts by choosing to either attack or hide malicious activity based on the environment it senses, they analyzed the strategic interaction between developers of malware and anti-malware. The authors discussed, the conditions for the anti-malware to protect all its machines and identified conditions under which an optimal anti-malware strategy can be computed efficiently. It also provided a Quadratically

Constrained Quadratic Program (QCQP) based optimization framework to compute the optimal anti-malware strategy. Additionally, the document identified a natural and easy-to-compute strategy for the anti-malware, which achieves utility close to the optimal utility in equilibrium.

Brodschelm & Gelderie [14] addressed the challenges of sandboxing on Linux desktops in its initial section, highlighting issues such as the diverse range of software and configurations, the need for user-friendliness, and the absence of a widely accepted solution. They proposed a container-based architecture to tackle these challenges, aiming to further isolate individual applications using namespaces, UIDs, and GIDs. They provided sandbox profiles with example applications and implemented a proof-of-concept. To assess the usability of their method, the authors conducted a poll with 20 participants, revealing that the concept of sandboxing was generally well-received and easy to implement. They also examined the security implications of their approach and found that it effectively isolated applications, thereby reducing the system's attack surface. In conclusion, the authors emphasized the potential of their approach as an initial step in incrementally strengthening the standard Linux desktop. They discussed future research directions, including the long-term evaluation of application stability, access control for the D-Bus session bus, and network access isolation.

Chen et al. [15] presented a method for automatically extracting features of malware from host logs. The method is tested using the WannaCry ransomware and normal activities. The results showed that the method can accurately identify features of the malware even when a majority of the logs contain non-malicious activity. The method is also robust to variations in the number of normal activity logs. Additionally, the method can identify features of polymorphic versions of the WannaCry malware. The results demonstrated the potential for automating malware analysis and pattern generation.

Tan et al. [16] presented ColdPress, an extensible malware analysis platform that automates the process of malware threat intelligence gathering. It combined state-of-the-art tools and concepts into a modular system that aids analysts in extracting information from malware samples. The platform is user-friendly and can be extended with user-defined modules. ColdPress has been evaluated with real-world malware samples and has demonstrated efficiency, performance, and usefulness to security analysts. The platform is containerized and can be easily deployed on different operating systems. Plans for ColdPress include adding more external modules and output formats.

Al-Marghilani [17] offered a thorough examination of several IoT malware evasion strategies, including virtual machine-based tactics, code obfuscation, polymorphism, and metamorphism. The difficulties in identifying and stopping IoT malware are also covered, including the intricacy of IoT systems, the absence of standards, and the requirement for immediate detection and action. The necessity of trust-based schemes—which depend on reputation-based systems to identify and stop malware attacks—is emphasized in the article. It also covered the usage of graph-based techniques, which used behaviour analysis and network architecture to detect and stop malware attacks, as well as Honeypot-based Collaborative Protection (HCP). The legal and regulatory difficulties in safeguarding Internet of Things (IoT) systems are also covered in the study, along with the necessity for IoT authorities and Computer Emergency Response Teams (CERTS) guidelines. To facilitate the deployment of a sophisticated analysis environment, the author emphasized the significance of integrating the malware analysis process with environment configuration and offered suggestions for resolving the legal and regulatory issues related to enhancing the dynamic malware analysis procedure and safeguarding IoT systems.

Liu et al. [18] proposed a system called User Behavior Emulator (UBER) designed to enhance malware analysis sandboxes by generating realistic system artefacts based on automatically derived user profile models. UBER aimed to prevent sandbox detection by malware leveraging system fingerprinting. The architecture comprised four elements: computer usage collector, user profile generator, artefact generator, and update scheduler. The collector gathers user system data, and the generator creates user behaviour profiles. Next, in an execution environment, the artefact generator replicates realistic system artefacts. The malware analysis framework's emulated environment is routinely copied by the update scheduler to create the sandbox. UBER modelled user behaviour from raw usage data to maintain authenticity, offering a secure emulation process transparent to malware. Regular cleaning and removal of UBER components precede cloning to prevent its use as a sandbox detection indicator. This ensures a continuous supply of authentic system artefacts for effective malware analysis.

Xie et al. [19] proposed a technique to enhance the protection of the Linux sandbox against malware sensitive to environmental factors. They distinguished a physical machine, a virtual machine, and a sandbox based on the first six characteristics of the Linux environment, including wear and tear, hardware, software, networks, user behaviour, and system configuration. The authors developed a tool named EnvFaker to collect these features from the operating environment, as illustrated in Fig. 3. EnvFaker examined each feature, and if any item triggered the rule, it contributed to the statistical data of that feature, potentially indicating the presence of a sandbox. The differences in features between physical machines, virtual machines, and sandboxes. EnvFaker's attributes were compared across different settings, such as sandboxes, virtual machines, and physical computers. The experiment utilized three popular virtual machine platforms and three well-known open-source sandboxes (Cuckoo, Limon, and Lisa), all running on Ubuntu 18.04. The results demonstrated that the feature data collected by the detection tool was distinguishable. For instance, the secure log, message log, HTTP access log, and MySQL log of the used machine exhibited rapid growth, with counts significantly higher than those of the new machine. Process counts and TCP connection counts also slightly exceeded those of the new machine. Comparing physical machines with virtual machines, significant differences were observed in sensitive processes, attributed to virtual machines deploying daemon processes for platform control convenience. Hardware strings also vary due to unique configurations in virtual machines. The authors concluded that EnvFaker effectively strengthened the Linux sandbox against environmental-sensitive malware, efficiently detecting discrepancies between physical machines, virtual machines, and sandboxes. EnvFaker was highlighted as a lightweight, user-friendly, and more capable tool compared to other well-known sandboxes in the

market.

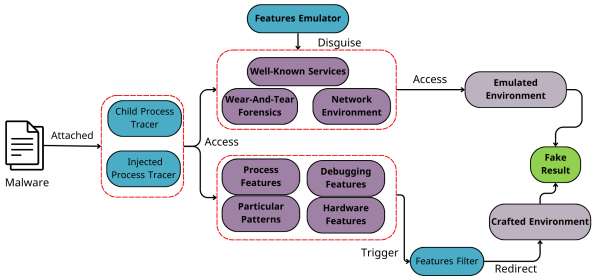

Fig. 3. Architecture of EnvFaker.

Naseer et al. [20] addressed the challenges associated with identifying malware and proposed potential solutions. They discussed the significance of malware detection in the contemporary digital environment and provided a detailed examination of various types of malware, including viruses, worms, and Trojan horses, along with the methods through which they can infect a system. They delved into the difficulties inherent in malware detection, including the need for real-time detection, the utilization of encryption and obfuscation techniques, and the increasing complexity of malware. It highlighted the limitations of conventional signature-based detection methods and underscored the necessity for more advanced approaches such as behavioural analysis and machine learning. Various malware detection techniques were explored, encompassing hybrid methods, PAM clustering, and machine learning-based approaches. The paper presented recommendations for further research and conducted a comprehensive analysis of each technique, outlining their respective advantages and disadvantages. Notably, the paper discussed various machine learning algorithms, including decision trees, support vector machines, and neural networks, and highlighted the effectiveness of machine learning-based techniques in identifying Android malware. The authors also covered the critical role of feature engineering and feature selection in enhancing the precision of machine learning-based methods.

Gazzan and Sheldon [21] conducted a comprehensive review of the literature addressing ransomware attacks on Supervisory Control And Data Acquisition (SCADA) and Industrial Control Systems (ICS). They examined the organizational and technical facets of the ransomware issue, talking about the difficulties in predictive modelling and highlighting the need for situational awareness in identifying and averting ransomware attacks. The authors identified distinctive features of ICS and SCADA systems that make them susceptible to ransomware attacks, including outdated and proprietary software, a lack of security protocols, and the potential for physical damage to critical infrastructure. They proposed a situational-based framework for ransomware prediction, combining operational and behavioural aspects of malware attacks. The suggested framework for handling ransomware incidents and situational awareness aimed to integrate managerial and organizational policies vertically, with a horizontal incorporation of the human element. The framework comprised three essential components: stakeholders (cybersecurity team, management team, and end users), inputs (SCADA design, cybersecurity policy playbooks, threat intelligence, and operational data), and outputs (perception, comprehension, and projection). The framework involved gathering incident-related data from the SCADA environment (perception), synthesizing incident components, determining the severity of cybersecurity objectives (comprehension), and projecting potential ransomware incident scenarios for planning the proper response (projection) to gather data related to situational awareness about ransomware attacks. Due to the framework's adaptability to operational and behavioural changes in ransomware and target systems, it could. The framework made use of managerial and organizational data as well as details from the ransomware process to predict future attacks by analyzing the malware's and the system's behaviour. In summary, the study offered insightful information about how ICS and SCADA systems are susceptible to ransomware attacks and suggested countermeasures for early detection and avoidance.

Yamany et al. [22] the experimental work conducted to investigate the behaviour of the SALAM ransomware was detailed, employing both static and dynamic analysis techniques. The authors utilized reverse engineering to identify intriguing strings, imports, and network activities associated with the ransomware. Through their analysis, they discovered that the SALAM ransomware encrypts files on infected machines using a variation of the Salsa20 encryption algorithm. The researchers also examined the ransomware's ability to propagate across a network and devised a decryption script to recover encrypted files. The SALAM ransomware, for encrypting all files on the compromised computer, generated a random key. Leveraging the ransomware's encryption key, the authors successfully created a decryption script capable of unlocking encrypted files without requiring payment of the ransom. The paper highlighted the importance of combining static and dynamic analysis techniques for the detection and analysis of malware. It also compared various types of ransomware and malware analysis approaches, delineating their respective advantages and disadvantages, as illustrated in Table II. Additionally, the authors underscored the necessity of proactive measures that businesses can adopt to defend themselves against ransomware attacks. These measures include implementing robust security protocols, regularly backing up data, and training staff on recognizing and avoiding phishing scams. In summary, the paper provided a comprehensive examination of the SALAM ransomware's behaviour and the challenges associated with decrypting it. It also offered valuable insights into the increasing sophistication of ransomware attacks and the critical importance of taking preventive actions.

Fasna and Swamy [23] described sandboxes and their operation. They defined sandboxes as virtualized environments simulating live systems, ensuring that the executable under test operates similarly to the actual environment. The paper explained how sandbox systems reduce the risk of compromising live systems by monitoring suspicious executable files in a controlled environment. It also covered various types of sandboxes, including appliance and cloud sandboxes. Cloud sandboxes, hosted in the cloud and accessible from any location, were contrasted with appliance sandboxes, installed onsite to offer greater control over the sandbox environment. The paper discussed the concept of evasion concerning sandboxes, elucidating how attackers could use it to bypass sandboxing. It

TABLE II. MALWARE ANALYSIS APPROACHES

Malware Analysis Type	Advantages	Disadvantages	Tools and Technologies
Static Analysis	It requires little kernel overhead and can be completed in a brief run-time.	The accuracy of malware detection is also less in static analysis.	Virustotal, Google, PE Explorer, CEF Explorer, and Resource Hacker.
Dynamic Analysis	Discovers and verifies vulnerabilities that occur during run-time.	a large amount of kernel overhead that may cause the system to lag while it is analyzed.	Wireshark, Process Monitor, Process Explorer, IDA Pro, OllyDbg.
Hybrid Analysis	Because it can detect malicious malware and reduce false negatives, it is more accurate than any other analysis type.	kernel overhead and cause systems to lag when being analyzed.	Ghidra, Windbg, gdb, Java Decompiler.
Sandboxing	Users can run files or programs in an isolated testing environment without affecting the application.	Making the testing environment resemble the actual production environment requires a certain set of skills.	Cuckoo Sandbox, AnyRun Sandbox, Joe Sandbox.

outlined the limitations of sandboxes, including their inability to detect all types of malware and susceptibility to circumvention through sophisticated obfuscation techniques. In summary, the paper presented a comprehensive analysis of sandboxes and their importance in protecting organizations against malicious software.

Edukulla. [24] explained that conventional web browsers and email apps are used to check downloaded files for malware to protect users from potential risks. The limitations, however, appeared when the downloaded file was larger than what was allowed for scanning, or when the malware signature was missing from worldwide databases of malware that was known to exist. To overcome these constraints, the authors proposed utilizing a sandbox environment to isolate files downloaded during web browsing, protecting against the potential dangers of opening unscanned malicious files. The sandbox environment could be implemented on the user's device or within a cloud platform. Scanning methods involved deep content inspection and signature matching against known malware, as illustrated in Fig. 4. The paper also discussed incorporating suitable User Interface (UI) mechanisms to enhance the outlined techniques, allowing the web browser to indicate a file's known malware status. For instance, download links for files known to contain malware could be marked with an alert, such as a red check mark, while links for safe files could be marked with a green check mark. In summary, by sandboxing downloaded files and conducting malware checks, the paper provided a comprehensive method to safeguard users against potential cyberattacks when downloading files from the internet.

Iqbal et al. [25] discussed the use of sandboxing techniques and tools such as Sandboxie and Symantec Workspace Virtualization in digital forensic investigations. It explored how these tools can automate the process of finding digital forensic artefacts in a Windows system. They provided a background on sandboxing and the tools used, described the research methodology, and presented the results and comparative analysis of the tools. The paper concluded with the value of sandboxing in

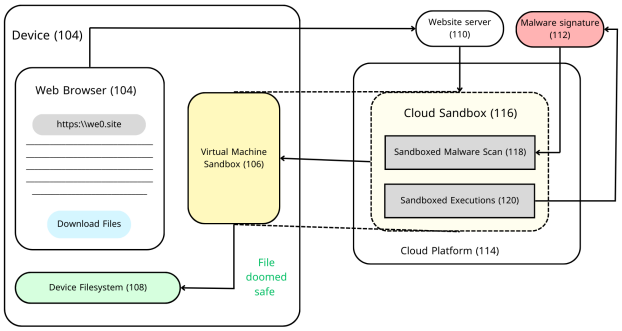

Fig. 4. Sandboxed inspection of the downloaded file to check for malware.

digital forensic investigations and suggestions for future work.

Yokoyama et al. [26] described a method for utilizing the Windows-based program SandPrint to exfiltrate malware's sandbox features. The program analyzed and published sandbox properties, collecting data on installed (or emulated) hardware, network settings, and precise OS details. Over two weeks, the authors submitted SandPrint to 20 malware analysis services, resulting in 66 analysis reports from 11 of these services. Employing unsupervised learning processes, they determined the features of 76 sandboxes by grouping the SandPrint reports and their distinct features. Furthermore, the authors used the SandPrint data to train an automated classifier capable of distinguishing between a user system and a sandbox. The tool aimed to provide sandbox operators with information on how to deploy more covert analysis systems and protect their systems against malware intrusions. They demonstrated the identification of malware security appliances using traits gleaned from public sandboxes, even in the absence of prior knowledge about the inner workings of the appliance's sandbox. Additionally, the paper offered insights for sandbox operators on implementing more covert analysis systems and incorporating a responsible disclosure procedure for alerting organizations to create sandboxes and/or appliances.

Namanya et al. [27] presented a summary of the malware landscape, providing background data for a planned investigation into creating malware detection methods. They defined malware, discussed its evolution over time, and described how malware had become more sophisticated and harder to detect. Attackers were noted to employ various techniques to evade detection and compromise systems. Current malware incidents, such as the WannaCrypt0r ransomware attack in 2017 and the Sony Pictures hack in 2014, were also discussed. The necessity of efficient malware detection and protection techniques was stressed, with an explanation of how these attacks impact both individuals and enterprises. The paper provided an overview of various methods of malware analysis, including hybrid, dynamic, and static analysis. It delved into the evasion strategies employed by malware, such as anti-debugging, anti-virtualization, and code obfuscation. The conclusion emphasized the crucial role of developing efficient malware detection frameworks to counter the growing threat of cybercrime. The paper highlighted the importance of a multi-layered approach to cybersecurity, involving firewalls, intrusion detection systems, antivirus software, and other security measures. Table III

summarizes the types of malware that are commonly known, including viruses, worms, Trojan horses, ransomware, adware, spyware, and rootkits which answer research question 1.

Talukder [28] provided a comprehensive overview of various malware types, including viruses, worms, Trojan horses, and ransomware. The paper extensively covered the tools and techniques employed for malware detection and analysis. Malware, identified as one of the most significant security risks on the internet, exhibited a consistent yearly increase in detections, with a notable spike in the middle of the 2010s, see Fig. 5. This graph underscored the escalating threat posed by malware, emphasizing the critical need for effective methods and tools in its identification and analysis. The author highlighted the importance of clearly classifying and differentiating between different types of malware. Various approaches to malware analysis, such as static, dynamic, and hybrid analysis, were discussed. The paper delved into different kinds of malware analysis tools available, covering areas like malware detection, memory forensics, packet analysis, scanners/sandboxes, reverse engineering, debugging, and website analysis. It provided a comprehensive inventory of tools accessible for analyzing each type of malware, categorizing them based on specific domains and methodologies. In summary, the article offered an in-depth exploration of malware detection and analysis techniques, providing a solid understanding of domain-specific analysis. It stands as a valuable resource for anyone interested in the field of malware analysis and detection.

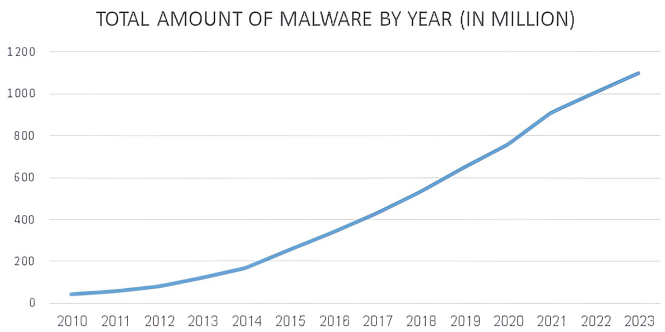

Fig. 5. Total number of malware detected by year (in millions) [29].

Kaur and Bindal [30] focused on dynamic malware analysis, aimed to provide a general overview of the characteristics of recent malware and discuss the methods and resources utilized in this field, with a particular emphasis on the Cuckoo sandbox running on Windows XP (SP3). The paper began by highlighting the sheer volume of malware samples received by anti-malware companies daily, emphasizing the importance of automatically analyzing these samples. Dynamic malware analysis, as explained in the paper, involves running a program in a controlled environment and generating a report that describes the behaviour of the program. They detailed the various methods and tools employed in dynamic malware analysis, focusing on the Cuckoo sandbox—an automated malware analysis system available as an open-source download. The authors explained how the Cuckoo sandbox operates and how it can be utilized to examine malware behaviour. They provided a comprehensive overview of the common characteristics of contemporary malware, including code obfuscation, rootkit functionality, and anti-debugging techniques. The paper clarified how these characteristics can be identified and analyzed through the application of dynamic malware analysis techniques. In conclusion, the paper offered insightful information about the general characteristics of contemporary malware and the methods and resources employed in dynamic malware analysis. It suggested the need for further research in this area and the development of improved methods for examining samples of unknown malware.

Küchler et al. [31] suggested that the study aimed to find the optimal time for executing a malware sample in a sandbox to collect sufficient data for classification without wasting resources or jeopardizing the experiments. The paper presented a large-scale study on how the execution time affects the amount and quality of collected events, such as system calls and code coverage. It also discussed implementing a machine learning-based malware detection method and its application to data collected over different time windows. The paper mentioned using 32 different sandboxes for their analysis, and the operating system used is the 32-bit version of Windows 7. The authors concluded that most malware samples either run for less than two minutes or more than ten minutes in a sandbox. However, most of the behavior is observed during the first two minutes of execution, yielding higher accuracy for their machine learning classifier. They recommended that two minutes is generally sufficient for analyzing fresh malware samples in a sandbox environment.

Denham et al. [32] discussed the threat of ransomware, a type of malware that encrypts data on a device and demands payment for decryption, the specific analysis of two ransomware samples: Wannacry and Cryptolocker. The authors aimed to identify and understand ransomware's obfuscation and propagation techniques within a sandbox environment to develop mitigation methods. It covered topics such as asymmetric encryption and cryptocurrency in ransomware attacks. The authors employed a dual approach of dynamic and static analysis within a sandbox environment, utilizing Oracle's VirtualBox.It was chosen for its open-source nature, high customizability, and support for snapshots, which are helpful for malware sandboxing.

Akhtar and Feng [33] emphasized the effectiveness of machine learning algorithms such as Support Vector Machines (SVM), Decision Trees (DT), and Convolutional Neural Networks (CNN) are effective malware detectors with low false positive rates. The results indicated that SVM achieved an accuracy of 96.41%, while DT achieved 99%, and CNN achieved 98.76%. The paper also mentioned the cyber kill chain, devised by Lockheed Martin, outlines the stages of a cyber attack, providing a strategic framework for preventing and mitigating intrusions see Fig. 6. The chain consists of seven stages: Reconnaissance, where attackers gather information; Weaponization, involving the creation of malicious tools; Delivery, the transport of malware to the target; Exploitation, the active use of vulnerabilities; Installation, establishing a foothold on the compromised system; Command and Control, enabling communication with a remote server; and finally, Actions on Objectives, where attackers achieve their goals. To prevent cyber intrusions, organizations implement security measures at each stage. These measures encompass threat intelligence, email and web filtering, vulnerability management,

TABLE III. COMMON MALWARE TYPES

Type of Malware	Description	Propagation	Delivery	Targets	Notable Characteristics
Virus	Self-replicating malware that spreads through infected files or scripts.	Email, downloads, websites.	Requires user interaction.	Files, applications, OS.	Destructive or data-stealing.
Worm	Self-propagating malware that spreads through network vulnerabilities.	Network transmissions, emails, websites.	Rapidly infects multiple systems.	Networked computers, servers.	No user interaction is required.
Trojan	Deceptive malware disguised as legitimate software.	Email, downloads, websites.	Deceives users for installation.	User systems, data.	Unauthorized access, data theft.
Ransomware	Encrypts files and demands payment for decryption.	Email, downloads, websites.	Monetarily motivated.	Individuals, businesses.	Highly disruptive.
Adware	Displays unwanted ads, collect user data.	Software bundles, downloads, websites.	Generates ad revenue.	User data for targeted ads.	Slows down systems.
Spyware	Spies on users, and captures sensitive data.	Downloads, websites, bundled with other malware.	Covert data ex-filtration.	Keystrokes, login credentials.	Data theft focus.
Rootkit	Hides presence, allows unauthorized access.	Often part of other malware.	Difficult to detect, maintains persistence.	Data theft, system control.	Backdoor access.

endpoint protection, firewalls, intrusion detection systems, security awareness training, and incident response planning. Organizations can enhance their overall cybersecurity resilience by addressing the various stages of the Cyber Kill Chain.

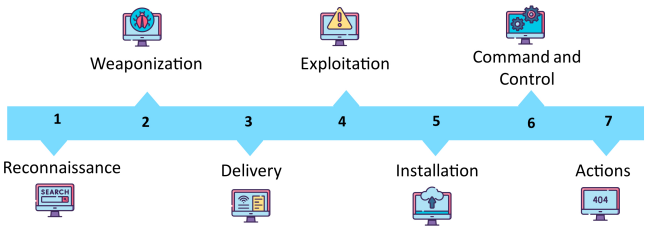

Fig. 6. Cyber kill chain.

Ijaz et al. [34] significantly contributed to the critical domain of malware detection in internet security, along with the pressing need for robust defence mechanisms against the escalating threat landscape of malware. A key focus of the research was on the analysis of executable binaries, constituting 47.80% of malware. Notably, the authors employed a classification approach, identifying malware categories such as Virus, Trojan Horse, Adware, Worm, and Backdoor. Also, they very complicated explored both static and dynamic features for comprehensive malware analysis, extracting over 2300 features dynamically and 92 features statically from binary files using PEFILE. The efficacy of the Cuckoo sandbox in dynamic malware analysis was highlighted, showcasing its accuracy and customizability. The examination spans static features drawn from a substantial dataset of 39000 malicious binaries and 10000 benign files, alongside the dynamic analysis of 800 benign files and 2200 malware files within the Cuckoo Sandbox. They outlined the limitations associated with dynamic malware analysis, addressing challenges related to controlled network behaviour, the original tactics employed by malware, and the complexities of analyzing packed malware with the added small difference of detecting virtualized environments. The study results show that the accuracy of static malware analysis is 99.36%, which is higher than the effectiveness of dynamic analysis. The paper not only provided valuable insights into the complexity of malware analysis but also suggested the advancement of detection methods through the integration of static and dynamic analyses with machine learning techniques, also proposed future directions aimed at overcoming dynamic analysis limitations and establishing an undetectable controlled environment for more effective malware analysis.

Ilić et al. [35] conducted a comparative study by systematically evaluating the performance of the Cuckoo and Drakvuf sandboxes across multiple critical features related to isolated program execution. Installation and setup complexity, scalability, reporting capabilities, execution time, evasion prevention, variety of analyses, integration with other tools, customization options, automated sample submission and API usage, signatures, and visualization were all taken into account during the assessment. The findings revealed that Cuckoo generally exhibits superior performance over Drakvuf, particularly in aspects such as documentation, installation ease, and widespread adoption by diverse organizations. Despite this, the authors underscored the importance of selecting a sandbox based on expected malware behaviour and highlighted Drakvuf's potential superiority in handling evasive and "fileless" malware scenarios. This valuable insight offered practical guidance to the professional community, aiding in a nuanced understanding of the strengths and weaknesses inherent in these sandboxes for malware analysis. The research contributes significantly to the ongoing efforts to enhance cybersecurity measures and practices by providing a comprehensive evaluation of the two sandboxes and their suitability for specific use cases. Additionally, they specifically documented a pilot comparative analysis focused on assessing the effectiveness and informative value of the reports generated by Cuckoo and Drakvuf in analyzing malicious programs. The study emphasized Drakvuf's status as an actively maintained and configurable solution, providing further depth to the evaluation of different features outlined in the paper.

V. CHALLENGES AND LIMITATIONS IN MALWARE DETECTION

A. Evasive Malware Detection

Evasive malware detection encounters challenges due to the increasing sophistication of evasion techniques, the rapid evolution of malware, the adaptive and dynamic nature of evasive malware, zero-day malware and emerging variants, limited availability of comprehensive datasets, high resource and time complexity in detection, and integration and compatibility issues with security systems. Additionally, there are difficulties in distinguishing legitimate vs. malicious evasion techniques,

recognizing unknown evasion techniques, optimizing execution environments, adapting to zero-day malware, and creating comprehensive behaviour datasets. On the limitations side, false positives and negatives in detection, lack of explainability in machine learning models, privacy concerns in sharing malware samples, attribution challenges, compatibility issues with legacy systems, limited scalability of current solutions, and the absence of standardization in evaluation metrics pose constraints [36], [37]. Table IV presented the most common challenges and limitations in evasive malware detection.

B. Real-Time Malware Analysis of IoT Devices

Analyzing IoT devices in real-time is tricky due to their varied and ever-changing features, the many types of malware they can encounter, the need for quick analysis, and the limited resources on these devices. There are also challenges like making the analysis work well across different IoT setups, understanding the complex behaviour of IoT malware, and keeping up with new threats. Existing tools for studying IoT malware have their limits too. They can struggle with things like handling many devices, adapting to different setups, and understanding the tricky behaviour of IoT malware. Privacy is also a concern. All these factors make it hard to effectively use existing tools for studying IoT malware [38].

Malware detection also has its difficulties. Malware creators use tricks to hide their code and make it tough to detect. Traditional methods might not catch these tricks, and advanced malware can disguise itself well. Machine learning, a potential solution, has its problems, like needing a lot of good data. Setting up a safe space (sandbox) for IoT devices to run and test programs also has its issues, like needing special tools and the risk of thinking a harmless program is dangerous. To address these challenges, experts recommend employing a combination of methods for malware detection, continuously monitoring emerging techniques, and continually enhancing the efficacy of tools to remain proactive against evolving threats.

C. Malware Detection and Analysis

Challenges in malware detection and analysis include the sophistication of evolving malware techniques, rapid evolution and variability of malicious code, concealed and polymorphic malware, detection of zero-day exploits, increasing scale and complexity of cyber threats, obfuscation and anti-analysis techniques employed by malware, and the dynamic and adaptive nature of modern malware. These challenges coexist with inherent uncertainties in identifying unknown threats, resource-intensive analysis, difficulty in differentiating between malicious and legitimate activity, limited effectiveness against polymorphic and encrypted malware, challenges in timely updates, lack of standardization, and privacy concerns with ethical implications in data analysis [1], [39].

D. Ransomware and IoT Malware Analysis

In ransomware analysis, challenges arise from the complex nature of ransomware, polymorphic behaviour, evasion techniques, and dynamic execution. Additionally, designing a comprehensive automation environment, addressing diverse characteristics and functionalities of IoT malware, adapting to the dynamic behaviour of IoT malware, ensuring adaptability to evolving threats, and handling the intricacies of automation poses challenges. Limitations include dataset diversity, dependence on sandboxing, time and resource constraints, and adaptability to new variants in ransomware, while IoT malware analysis faces challenges in achieving complete automation [40], [41].

E. Machine Learning for Malware Detection

Machine learning for malware detection encounters adversarial attacks, where threat actors deliberately employ obfuscation techniques to evade detection, posing a significant challenge for machine learning models. Imbalanced datasets, characterized by a disproportionate number of samples in different classes, can lead to biased models and impact the overall performance of detection systems. Feature engineering, a critical aspect of machine learning, becomes complex in the context of malware detection due to the need to identify discriminative features from intricate and evolving malware samples. The dynamic and polymorphic nature of malware further complicates detection, as models must adapt to new variants and their evolving characteristics, while also generalizing across these variants. Overfitting, lack of transparency leading to interpretability issues, resource intensiveness, and the absence of causality understanding present additional limitations. Furthermore, concept drift, where the statistical properties of data change over time, adds to the complexity of maintaining accurate and reliable detection models. These multifaceted challenges and limitations underscore the imperative for continuous research and innovation to develop machine-learning models that can effectively address the intricacies of malware detection [42].

F. IoT Malware Evasion Techniques

Challenges in IoT malware evasion techniques involve increasing sophistication of evasion techniques, rapid evolution of malware in the IoT environment, dynamic and adaptive nature of evasive malware in IoT devices, variability and proliferation of IoT architectures, limited availability of comprehensive datasets specific to IoT malware, resource and processing constraints in IoT devices, and interoperability challenges in integrating evasive malware detection with IoT security systems. These challenges coexist with difficulties in distinguishing legitimate IoT device behaviour, recognizing emerging and unknown evasion tactics, practical implementation issues in optimizing execution environments, efficient adaptability to zero-day IoT malware, and challenges in prioritizing and creating comprehensive datasets specifically tailored for IoT malware [43].

G. Industrial Control Systems

In Industrial Control Systems (ICS), challenges include identifying subtle early indicators of ransomware attacks, adapting detection mechanisms to unique characteristics and protocols of ICS environments, addressing increasing complexity and sophistication of ransomware attack techniques, overcoming limitations in real-time monitoring and analysis of ICS network traffic, and ensuring compatibility and integration of detection solutions with diverse ICS architectures [44].

H. Behavioral Analysis

An additional reference addressing challenges in behavioural analysis, anomaly detection, and the interpretation of security alerts highlighted issues like over-reliance on static features, scalability challenges, context-aware detection difficulties, resource intensiveness, evolving tactics of malicious actors, and ethical and privacy concerns [45].

TABLE IV. MOST COMMON CHALLENGES AND LIMITATIONS IN EVASIVE MALWARE DETECTION [46]

Challenges in Evasive Malware Detection	Limitations
Increasing Sophistication of Evasion Techniques	Difficulty in Distinguishing Legitimate vs. Malicious Evasion Techniques
Rapid Evolution of Malware	Detection and Recognition of Unknown Evasion Techniques
Adaptive and Dynamic Nature of Evasive Malware	Optimizing Execution Environments for Practical Implementation
Zero-Day Malware and Emerging Variants	Efficient Adaptability to Zero-Day Malware Through Learning Mechanisms, Including Resource and Time Constraints
Limited Availability of Comprehensive Datasets	Challenges in Prioritizing and Creating Comprehensive Evasive Behavior Datasets
High Resource and Time Complexity in Detection	Balancing Complexity in Multiple Execution Environments
Integration and Compatibility with Security Systems	Implementation Challenges in Adapting Detection Mechanisms to Existing Security Infrastructure

VI. FUTURE EXTENSION

In this section, new directions for the future of malware analysis are proposed. These directions are envisioned to shape the field and contribute to significant advancements. The ongoing fight against complex malware is still a major concern in cybersecurity. Predicting future developments in evasive malware detection and malware sandbox development poses both excitement and challenges. To keep up with increasingly complex evasion strategies in the future, the focus will be on improving detection procedures. Exploring the machine learning models holds massive potential for enhancing the agility and accuracy of malware detection systems. In addition, research on the development of malware sandboxes will remain crucial, with a focus on building settings that can adapt to real-world situations. Continued efforts will be directed towards fortifying cybersecurity defenses against emerging evasive malware threats, ensuring their resilience and efficacy. This proactive strategy is necessary to address the static and dynamic landscapes of cybersecurity threats.

A. Evasive Behavior Dataset Creation

Prioritizing the development of a comprehensive dataset that accurately represents evasive behaviours is strongly recommended. Such a resource will significantly enhance researchers' ability to devise more robust solutions for detecting evasive malware. To make an evasive behaviour dataset, first, record different situations where objects exhibit evasive manoeuvres in real life using cameras or other sensors. Then, mark these instances in the recordings by specifying what objects are involved, when it happens, and what kind of avoidance is occurring. Also, include scenes where no evasive actions take place to help train the model in what's normal. Check the data carefully to make sure it's accurate, and be mindful of privacy by blurring sensitive details. Split the dataset into different parts for training and testing, and write down how you collected everything. If you share the dataset, do it responsibly. Keep improving the dataset as you learn more about what the model needs to understand [47].

B. Distinguishing between Legitimate and Malicious or Unknown Evasion Techniques

Addressing the challenge of distinguishing between evasion techniques used in legitimate behaviour and those employed for malicious purposes is essential. Developing accurate classification methods is crucial for effective detection. It involves using smart systems that learn normal behaviour patterns and recognize anomalies, employing known signatures of malicious tactics, and implementing rules and dynamic analysis. By considering the supervised and unsupervised methods through machine learning, these systems can effectively identify and respond to potential threats. Regular updates, human oversight, and integration of threat intelligence contribute to a comprehensive approach to stay ahead of evolving evasion techniques [48].

C. Optimizing Execution Environments

Tackling the challenge of utilizing multiple execution environments in evasive malware detection without introducing high complexity in terms of time and resources is crucial. Streamlining this process is essential for practical implementation. To optimize execution environments for evasive malware detection, start by clearly identifying the different platforms relevant to your system. Conduct thorough testing across diverse environments to ensure the effectiveness of detection algorithms, addressing challenges and ensuring adaptability [49]. Develop adaptive algorithms that can dynamically adjust to various execution contexts, and implement parallel processing techniques to handle multiple environments simultaneously, reducing detection time. Document optimized configurations, algorithms, and deployment strategies for each platform to facilitate effective maintenance and updates. By focusing on these key steps, you can efficiently manage multiple execution environments, making a balance between practical implementation and considerations of time and resources [50].

D. Zero-Day Malware Adaptability

Developing and implementing efficient updating learning mechanisms to adaptively learn new behaviours, particularly in the context of zero-day malware and emerging variants, is suggested. Deep learning and unsupervised machine learning can play a crucial role in this adaptation. Detecting and addressing the adaptability of Zero-Day malware involves several key steps. First, understand these threats' dynamic nature by analyzing historical instances and identifying common evasion tactics. Secondly, explore adaptive algorithms that can quickly evolve to recognize new, unseen malware patterns. Investigate the vulnerabilities and weaknesses exploited by Zero-Day malware to enhance preemptive defences. Thirdly, implement real-time monitoring and analysis to swiftly identify anomalous behaviours indicative of Zero-Day threats. Collaborate with threat intelligence communities to stay informed about emerging trends. Additionally, regularly update security protocols and leverage machine learning to predict potential adaptation

strategies. Finally, consider incorporating deception techniques and honeypots to divert and confuse evolving malware. By highlighting these crucial areas for future work, researchers can contribute significantly to overcoming existing challenges in evasive malware detection and advancing the development of more effective and adaptive solutions [51].

VII. Discussion

The discussion encompasses an analysis of the challenges and limitations in malware detection, insights into future directions, and the significance of malware sandboxing in cybersecurity. The challenges outlined shed light on the multifaceted nature of malware detection. From the increasing sophistication of evasion techniques to the rapid evolution of malware and the scarcity of comprehensive datasets, detecting and analyzing malicious software pose significant hurdles. Moreover, the dynamic nature of evasive malware, the emergence of zero-day exploits, and the resource-intensive nature of detection further complicate the task. These challenges are exacerbated by limitations such as false positives and negatives, lack of explainability in machine learning models, and compatibility issues with legacy systems.

Understanding these challenges is crucial for advancing malware detection and analysis techniques. Recognizing the need for innovative approaches, such as machine learning models and behavioural analysis, can help overcome the limitations of traditional detection methods. Moreover, prioritizing the creation of comprehensive datasets and enhancing compatibility with existing security systems can improve the efficacy of malware detection solutions. Additionally, addressing privacy concerns and ensuring transparency in detection methodologies are essential for building trust in the cybersecurity community.

The proposed future directions underscore the importance of continuous innovation in malware analysis. Leveraging machine learning models holds promise for enhancing detection accuracy and agility, while the development of malware sandboxes remains crucial for creating secure environments for analysis. Emphasizing the creation of evasive behaviour datasets, distinguishing between legitimate and malicious evasion techniques, optimizing execution environments, and adapting to zero-day malware are key areas for future research and development. By addressing these challenges and embracing emerging technologies, the cybersecurity community can stay ahead of evolving threats and safeguard digital ecosystems effectively.

Malware sandboxing emerges as a linchpin of cybersecurity in the discussion. By providing controlled environments for malware analysis, sandboxes enable security experts to dissect and understand the behaviour of malicious software without compromising the integrity of the host system. The comparative analysis of various malware sandboxes highlights their diverse features and capabilities, offering insights into their effectiveness in detecting and analyzing malware. Moreover, the literature survey underscores the importance of sandboxes in facilitating dynamic analysis, detecting ransomware attacks, and leveraging machine learning algorithms for malware detection and classification.

In conclusion, the discussion underscores the intricate challenges and promising avenues in malware detection and analysis. By addressing these challenges and embracing innovative approaches, the cybersecurity community can fortify defences against evolving threats and safeguard digital environments effectively. Malware sandboxing remains a cornerstone of cybersecurity, offering a secure space for thorough analysis and empowering security professionals to stay ahead of malicious actors. Moving forward, collaboration, research, and continuous innovation are essential for advancing malware detection and analysis techniques and ensuring the resilience of digital ecosystems against cyber threats.

VIII. Conclusion

In the ever-advancing landscape of cybersecurity, the evolution of malware sandbox technology stands out as a critical defence against sophisticated threats. Modern sandboxes, infused with artificial intelligence and adaptive features, create realistic environments challenging for malware to evade.

Malware sandbox evaluation, conducted in controlled environments, proves instrumental in understanding and mitigating malicious threats. Security experts gain crucial insights by closely monitoring network communications, system interactions, and evasion techniques. This knowledge enhances detection methods and fuels the development of robust defence strategies.

The impact of sandbox evaluation extends beyond immediate threat identification, empowering security professionals to improve tools and strategies proactively. Collaboration within security communities remains vital, ensuring collective strength against emerging malware behaviours.

In essence, malware sandbox evaluation is a linchpin of cybersecurity, offering a secure space for thorough analysis and equipping experts to safeguard digital environments effectively. This proactive approach, coupled with ongoing research and collaboration, fortifies defences against the dynamic nature of modern cyber threats.

An analysis of the related work is presented. Table V summarizes the characteristics of the related sandboxes and compares them, addressing research question 2. The table includes the following characteristics:

- Malware Sandbox: The name of the malware sandbox.
- Description: The description of the malware sandbox.
- Analysis Capabilities: If Assess the sandbox's ability to analyze code without executing it or during execution.
- OS: The operating system the malware sandbox supports.
- Signature-Based: If the malware sandbox relies on signature-based detection.
- Detection Techniques: The techniques used to detect malware in the malware sandbox.
- Licensing Model: If the malware sandbox has an open-source or commercial license.

TABLE V. MALWARE SANDBOX

Malware Sandbox	Description	Analysis Capabilities	OS	Signature Based	Detection Techniques	Licensing Model
Cuckoo Sandbox [1], [18], [52], [8], [25], [34].	A malicious code investigation tool that examines malware in detail and provides comprehensive results based on the series of tests made by it during the execution of the malicious code sample.	Dynamic and Static analysis.	Windows, Linux, and macOS.	NO	A combination of behavioural and static analysis techniques to detect malware.	Open-Source
Limon Sandbox [18].	An open-source sandbox designed for dynamic malware analysis. It focuses on analyzing malware behaviour during runtime to understand its impact on a system.	Dynamic analysis	Linux	YES	A combination of heuristics and behavioural analysis techniques	Open-Source
Lisa Sandbox [18].	A powerful virtual environment that allows researchers, analysts, and security professionals to examine and analyze potentially harmful files safely. It provides a secure environment to execute and observe the behaviour of files without risking the host system's integrity.	Dynamic and Static analysis.	Windows, Linux, and macOS	YES	A combination of behaviour-based analysis, signature-based detection, machine learning algorithms, heuristics, and anomaly detection.	Free versions with limited features and offer commercial licenses
Joe Sandbox [8], [53].	A fully automated malware analysis system that provides deep analysis and agile sandboxing capabilities. It supports all types of file formats, including Android apps, and generates reports in XML, JSON, HTML, PDF, etc.	Dynamic analysis.	Windows, Linux, and macOS	NO	A combination of behavioral and static analysis techniques	A commercial licenses
AnyRun Sandbox [8].	A cloud-based sandboxing platform that allows users to analyze malware behaviour in real-time	Dynamic and Static analysis.	Windows, Linux, and macOS	YES	Behavioral analysis techniques	A commercial licenses
VMRay Analyzer [54], [55].	An agent-less dynamic behaviour analysis tool for malware. It is embedded in the hypervisor to monitor the behaviour of malware and overcome the problem in traditional sandboxes.	Static and Dynamic analysis techniques	Windows, Linux, and macOS	YES	A combination of signature-based detection and behavioral analysis	A commercial licenses
Malwr [53].	An online platform and community-driven malware analysis service that allows users to submit and analyze suspicious files in a controlled environment and give a very detailed report in html/xml format.	Dynamic analysis	Windows, Linux, and macOS	NO	A combination of behavioral and static analysis techniques	Open-Source
Threat Expert [53].	an online malware analysis system that provides a simple user interface for analyzing malware samples by submitting them. It generates a detailed report on the malware, including the time stamp of the malware, the type of packers used by the malware author, and the level of security.	Dynamic analysis	Windows	NO	A combination of behavioral and static analysis techniques	A commercial licenses
Drakvuf sandboxe [35].	Controlled environments created for executing and observing potentially malicious code. These sandboxes aim to provide a secure and isolated space where malware samples can be executed, allowing analysts to study their behaviour without risking damage to the actual operating environment.	Dynamic analysis	Windows	NO	behavior analysis techniques	Open-source

ACKNOWLEDGMENT

This work was supported by the Deanship of Scientific Research, Vice Presidency for Graduate Studies and Scientific Research, King Faisal University, Saudi Arabia [Grant No. 6128].

REFERENCES

[1] Elhanashi, A., Gasmi, K., Begni, A., Dini, P., Zheng, Q., & Saponara, S. (2022, September). Machine Learning Techniques for Anomaly-Based Detection System on CSE-CIC-IDS2018 Dataset. *In International Conference on Applications in Electronics Pervading Industry, Environment and Society*, **(2022)**,*(pp. 131-140)*. Cham: Springer Nature Switzerland.

[2] Yokoyama, A., Ishii, K., Tanabe, R., Papa, Y., Yoshioka, K., Matsumoto, T., ... & Rossow, C. Sandprint: Fingerprinting malware sandboxes to provide intelligence for sandbox evasion. *In Research in Attacks, Intrusions, and Defenses: 19th International Symposium, RAID 2016, Paris, France, September 19-21, 2016, Proceedings 19 (pp. 165-187)*. Springer International Publishing, **2016**.

[3] Faruk, M. J. H., Shahriar, H., Valero, M., Barsha, F. L., Sobhan, S., Khan, M. A., ... & Wu, F. (2021, December). Malware detection and prevention using artificial intelligence techniques. *In 2021 IEEE International Conference on Big Data (Big Data)*, **2021**, *(pp. 5369-5377)*. IEEE.

[4] Malware Sanboxes Available Online :https://www.vmray.com/glossary/malware-sandbox/ (accessed on 30 Nov 2023).

[5] Page MJ, McKenzie JE, Bossuyt PM, Boutron I, Hoffmann TC, Mulrow CD, et al, "The PRISMA 2020 Statement: An Updated Guideline for Reporting Systematic Reviews", *BMJ*, **2021**.

[6] What's the Difference Between a Sandbox and a Virtual Machine? Available online: https://askleo.com/whats-the-difference-between-a-sandbox-and-a-virtual-machine/ (accessed on 15 Nov 2023).

[7] Afianian, A., Niksefat, S., Sadeghiyan, B., & Baptiste, D. Malware dynamic analysis evasion techniques: A survey. *ACM Computing Surveys (CSUR)*, **2019**, *52(6), 1-28*.

[8] Sethi, K., Chaudhary, S. K., Tripathy, B. K., & Bera, P. A novel malware analysis framework for malware detection and classification using machine learning approach. *In Proceedings of the 19th international conference on distributed computing and networking*, **2018**, *(pp. 1-4)*.

[9] BELEA, A. R. Methods for Detecting Malware Using Static, Dynamic and Hybrid Analysis. *In Proceedings of the International Conference on Cybersecurity and Cybercrime-2023* **2023** *(pp. 258-5)*. Asociatia Romana pentru Asigurarea Securitatii Informatiei.

[10] UPPIN, C. Dynamic Analysis of a Window-Based Malware Using Automated sandboxing. *UPPIN, C. Dynamic Analysis of a Window-Based Malware Using Automated sandboxing.*, **2019**.

[11] Kamal, A., Derbali, M., Jan, S., Bangash, J. I., Khan, F. Q., Jerbi, H., ... & Ahmad, G. (2021). A User-friendly Model for Ransomware Analysis Using Sandboxing. *Computers, Materials & Continua*. **2021**, *67(3)*.

TABLE VI. THE LITERATURE SURVEY-1: OVERVIEW ON MALWARE SANDBOX

Authors	Publ.	Topic	Major Findings
Elhanashi et al. [1]	2022	Anomaly-based detection using ML	Discovered the way for robust and efficient cyber defence against diverse attacks using three different classifiers, outperforming existing methods with 99.6% accuracy.
Kamal et al. [11]	2021	User-friendly model for ransomware analysis using sandboxing	Developed a user-friendly ransomware analysis sandbox model called RASTA that leverages Cuckoo Sandbox and other tools.
Yong et al. [12]	2021	Practice of malware analysis	Provided insights into the various stages and techniques involved in malware analysis, including data collection, triage, static and dynamic analysis, and reporting.
Sikdar et al. [13]	2022	Anti-Malware Sandbox Games	Proposed the concept of anti-malware sandbox games, where analysts play games to train and improve AI models for malware detection in sandboxes.
Brodschelm & Gelderie [14]	2022	User-friendly application sandboxing for Linux desktops	Developed a user-friendly application sandboxing solution for Linux desktops called AppArmor, which leverages mandatory access control to restrict application behaviour.
Chen et al. [15]	2017	Automated behavioural analysis of malware	Presented an automated behavioural analysis approach for malware detection using Wannacry ransomware as a case study.
Tan et al. [16]	2021	Coldpress: Extensible malware analysis platform	Developed Coldpress, an extensible platform for malware analysis and threat intelligence gathering that combines dynamic analysis with network traffic monitoring.
Al-Marghilani [17]	2021	Comprehensive analysis of IoT malware evasion techniques	Analyzed various techniques used by IoT malware to evade detection in sandboxes, such as time-based execution, host fingerprinting, and API hooking.
UPPIN [10]	2019	Dynamic analysis of Windows malware using automated sandboxing	Analyzed a Windows malware sample using Cuckoo Sandbox and other tools to understand its behaviour and functionality.
Liu et al. [18]	2022	Enhancing malware analysis sandboxes with emulated user behavior	Proposed incorporating emulated user behaviour into malware analysis sandboxes to improve the detection of evasive malware.
Xie et al. [19]	2021	Envfaker: Reinforcing Linux sandbox against environmental-sensitive malware	Developed Envfaker, a method to reinforce Linux sandboxes against evasion techniques used by environment-sensitive malware.
Naseer et al. [20]	2021	Malware detection: Issues and challenges	Discussed various challenges in malware detection, including the increasing sophistication of malware, the use of obfuscation techniques, and the need for real-time detection.
BELEA et al. [9]	2023	Methods for detecting malware using static, dynamic, and hybrid analysis	Compared the effectiveness of static, dynamic, and hybrid malware analysis methods, highlighting the advantages and disadvantages of each approach.
Gazzan & Sheldon [21]	2023	Early detection and prediction of ransomware attacks against industrial control systems	Explored opportunities for early detection and prediction of ransomware attacks on industrial control systems using a combination of network traffic analysis and machine learning techniques.
Yamany et al. [22]	2021	SALAM Ransomware Behavior Analysis Challenges and Decryption	Analyzed the unique challenges in analyzing SALAM ransomware due to its use of encryption and anti-analysis techniques. Proposed potential decryption techniques.
Fasna & Swamy [23]	2022	Sandbox: A Secured Testing Framework for Applications	Developed a secured testing framework for applications using sandboxes to isolate and analyze their behaviour, preventing potential vulnerabilities from affecting the host system.
Edukulla [24]	2020	Sandboxing files downloaded via a web browser	Proposed techniques for sandboxing files downloaded from web browsers to mitigate the risk of malware infections.
Iqbal et al. [25]	2015	Sandboxing: Aid in digital forensic research	Demonstrated the use of sandboxes in digital forensic investigations to analyze malware behaviour and extract evidence for legal proceedings.

TABLE VII. THE LITERATURE SURVEY-2: OVERVIEW ON MALWARE SANDBOX

Authors	Publ.	Topic	Major Findings
Yokoyama et al. [26]	2016	Sandprint: Fingerprinting malware sandboxes to provide intelligence for sandbox evasion	Developed Sandprint, a tool for fingerprinting malware sandboxes to understand how malware tries to evade detection in sandbox environments.
Namanya et al. [27]	2018	The world of malware: An overview	Provided a comprehensive overview of the world of malware, including its history, types, motivations, and attack vectors.
Talukder [28]	2020	Tools and techniques for malware detection and analysis	Presented a survey of various tools and techniques used for malware detection and analysis, including static analysis, dynamic analysis, and machine learning-based approaches.
Kaur & Bindal [30]	2016	A complete dynamic malware analysis	Discussed the process of dynamic malware analysis, including steps such as execution in a controlled environment, behaviour monitoring, and analysis of results.
Sethi et al. [8]	2018	A novel malware analysis framework for malware detection and classification using machine learning approach	Developed a machine learning-based malware analysis framework for detecting and classifying malware based on features extracted from static and dynamic analysis.
Küchler et al. [31]	2021	Does Every Second Count? Time-based Evolution of Malware Behavior in Sandboxes	Analyzed how malware behaviour can evolve overtime in sandboxes, potentially affecting detection accuracy.
Denham et al. [32]	2022	Ransomware and malware sandboxing	Discussed the use of sandboxes for analyzing ransomware and other malware, highlighting their effectiveness in detecting malicious behaviour.
Akhtar & Feng [33]	2022	Malware Analysis and Detection Using Machine Learning Algorithms	Explored the use of machine learning algorithms for malware analysis and detection, demonstrating their potential in identifying malware variants and zero-day attacks.
Ijaz et al. [34]	2019	Static and dynamic malware analysis using machine learning	Compared the performance of static and dynamic malware analysis techniques using machine learning, finding that a combination of both approaches can improve detection accuracy.
Ilić et al. [35]	2022	A pilot comparative analysis of the Cuckoo and Drakvuf sandboxes: An end-user perspective	Conducted a comparative analysis of the Cuckoo and Drakvuf sandboxes, evaluating their features, performance, and ease of use from an end-user perspective.

[12] Yong Wong, M., Landen, M., Antonakakis, M., Blough, D. M., Redmiles, E. M., & Ahamad, M. (2021, November). An inside look into the practice of malware analysis. *In Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security* **2021** *(pp. 3053-3069)*.

[13] Sikdar, S., Ruan, S., Han, Q., Pitimanaaree, P., Blackthorne, J., Yener, B., & Xia, L. (2022). Anti-Malware Sandbox Games. *arXiv preprint arXiv:2202.13520*, **2022**.

[14] Brodschelm, L., & Gelderie, M. Application Sandboxing for Linux Desktops: A User-friendly Approach, Department of Electrical Engineering and Computer Science, Aalen University of Applied Sciences, **2022**.

[15] Chen, Q., & Bridges, R. A. Automated behavioral analysis of malware: A case study of wannacry ransomware. *In 2017 16th IEEE International Conference on machine learning and applications (ICMLA)* **2017***(pp. 454-460)*, IEEE.

[16] Tan, H., Chandramohan, M., Cifuentes, C., Bai, G., & Ko, R. K. Coldpress: An extensible malware analysis platform for threat intelligence. *arXiv preprint arXiv:2103.07012* **2021**.

[17] Al-Marghilani, A. Comprehensive Analysis of IoT Malware Evasion Techniques. *Engineering, Technology & Applied Science Research*,

2021 *11(4), 7495-7500*.

[18] Liu, S., Feng, P., Wang, S., Sun, K., & Cao, J. Enhancing malware analysis sandboxes with emulated user behavior. *Computers & Security*, **2022** *115, 1013*.

[19] Xie, C., Guo, Y., Shi, S., Sheng, Y., Chen, X., Li, C., & Wen, W. Envfaker: A method to reinforce linux sandbox based on tracer, filter and emulator against environmental-sensitive malware. *In 2021 IEEE 20th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, **2021** *(pp. 667-677). IEEE*.

[20] Naseer, M., Rusdi, J. F., Shanono, N. M., Salam, S., Muslim, Z. B., Abu, N. A., & Abadi, I. Malware detection: issues and challenges. *In Journal of Physics: Conference Series (Vol. 1807, No. 1, p. 012011). IOP Publishing.*, **2021**.

[21] Gazzan, M., & Sheldon, F. T. Opportunities for Early Detection and Prediction of Ransomware Attacks against Industrial Control Systems. *Future Internet*, **2023**, *15(4), 144*.

[22] Yamany, B. E. M., & Azer, M. A. SALAM Ransomware Behavior Analysis Challenges and Decryption. *In 2021 Tenth International Conference on Intelligent Computing and Information Systems (ICICIS)*, **2021** *(pp. 273-277). IEEE*.

[23] Fasna, V., & Swamy, R. Sandbox: A Secured Testing Framework for Applications, *Journal of Technology & Engineering Sciences*, **2022**.

[24] Edukulla, S. K. Sandboxing Files Downloaded Via A Web Browser. *Technical Disclosure Commons*, **2020**.

[25] Iqbal, A., Alobaidli, H., Guimaraes, M., & Popov, O. Sandboxing: aid in digital forensic research. *In Proceedings of the 2015 Information Security Curriculum Development Conference*, **2015** *(pp. 1-5)*.

[26] Yokoyama, A., Ishii, K., Tanabe, R., Papa, Y., Yoshioka, K., Matsumoto, T., ... & Rossow, C. Sandprint: Fingerprinting malware sandboxes to provide intelligence for sandbox evasion. *In Research in Attacks, Intrusions, and Defenses: 19th International Symposium, RAID 2016, Paris, France, September 19-21, 2016, Proceedings 19 (pp. 165-187)*, **2016**, Springer International Publishing.

[27] Namanya, A. P., Cullen, A., Awan, I. U., & Disso, J. P. (2018, August). The world of malware: An overview. *In 2018 IEEE 6th international conference on future internet of things and cloud (FiCloud)*, **2018** *(pp. 420-427)*. IEEE.

[28] Talukder, S. Tools and techniques for malware detection and analysis.*arXiv preprint arXiv:2002.06819*, **2020**.

[29] AV-TEST. (n.d.). Home. AV-TEST. Retrieved January 15, 2023, from https://www.av-test.org/en/

[30] Kaur, N., Bindal, A. K., & PhD, A. A complete dynamic malware analysis. *International Journal of Computer Applications*, **2016**, *135(4), 20-25*.

[31] Küchler, A., Mantovani, A., Han, Y., Bilge, L., & Balzarotti, D. (2021, February). Does Every Second Count? Time-based Evolution of Malware Behavior in Sandboxes. *In NDSS*, **2021**.

[32] Denham, B., & Thompson, D. R. Ransomware and malware sandboxing. *In 2022 IEEE 13th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, **2022**, *(pp. 0173-0179). IEEE*.

[33] Akhtar, M. S., & Feng, T. Malware Analysis and Detection Using Machine Learning Algorithms.*Symmetry*, **(2022)**, *14(11)*, 2304.

[34] Ijaz, M., Durad, M. H., & Ismail, M. Static and dynamic malware analysis using machine learning. *In 2019 16th International bhurban conference on applied sciences and technology (IBCAST)*, (**2019**, January),(pp. 687-691). IEEE.

[35] Ilić, S. Ž., Gnjatović, M. J., Popović, B. M., & Maček, N. D. A pilot comparative analysis of the Cuckoo and Drakvuf sandboxes: An end-user perspective. *Vojnotehnički glasnik/Military Technical Courier*,(**2022**),*70(2)*, 372-392.

[36] Le, H. V., & Ngo, Q. D. V-sandbox for dynamic analysis IoT botnet. *IEEE Access*, **2020**, *8, 145768-145786*.

[37] Aboaoja, F. A., Zainal, A., Ghaleb, F. A., Al-rimy, B. A. S., Eisa, T. A. E., & Elnour, A. A. H. Malware detection issues, challenges, and future directions: A survey. *Applied Sciences* ,**(2022)**, *12(17)*, 8482.

[38] Kachare, G. P., Choudhary, G., Shandilya, S. K., & Sihag, V. Sandbox Environment for Real Time Malware Analysis of IoT Devices. *In International Conference on Computing Science, Communication and Security*, **2022**, *(pp. 169-183)*. Cham: Springer International Publishing.

[39] Suraneni, N. Malware Detection and Analysis, *Culminating Experience Projects*, **2022**.

[40] Kamal, A., Derbali, M., Jan, S., Bangash, J. I., Khan, F. Q., Jerbi, H., ... & Ahmad, G. (2021). A User-friendly Model for Ransomware Analysis Using Sandboxing. *Computers, Materials & Continua*, **2021**, *67(3)*.

[41] Lee, S., Jeon, H., & Park, G. (2021). Design of automation environment for analyzing various IoT malware. *Tehnički vjesnik*, **2021**, *28(3)*, 827-835.

[42] Gibert, D., Mateu, C., & Planes, J. (2020). The rise of machine learning for detection and classification of malware: Research developments, trends and challenges. *Journal of Network and Computer Applications*, **2020**, *153*, 1025.

[43] Al-Marghilani, A. (2021). Comprehensive Analysis of IoT Malware Evasion Techniques.*Engineering, Technology & Applied Science Research*, **2021**, *11(4)*, 7495-7500.

[44] Gazzan, M., & Sheldon, F. T. (2023). Opportunities for Early Detection and Prediction of Ransomware Attacks against Industrial Control Systems. *Future Internet*, **2023**, *15(4)*, 144.

[45] Jeffrey, N., Tan, Q., & Villar, J. R. (2023). A Review of Anomaly Detection Strategies to Detect Threats to Cyber-Physical Systems. *Electronics*, **2023**, *12(15)*, 3283.

[46] Pilli, E. S., Joshi, R. C., & Niyogi, R. (2010). Network forensic frameworks: Survey and research challenges. *digital investigation*, **2010**, *7(1-2)*, 14-27.

[47] Khraisat, A., & Alazab, A. (2021). A critical review of intrusion detection systems in the internet of things: techniques, deployment strategy, validation strategy, attacks, public datasets and challenges. Cybersecurity,**(2021)**, 4, 1-27.

[48] Zhang, B., Xiao, W., Xiao, X., Sangaiah, A. K., Zhang, W., & Zhang, J. Ransomware classification using patch-based CNN and self-attention network on embedded N-grams of opcodes. *Future Generation Computer Systems*, **(2020)**,110, 708-720.

[49] Demetrio, L., Biggio, B., Lagorio, G., Roli, F., & Armando, A. Functionality-preserving black-box optimization of adversarial windows malware. IEEE Transactions on *Information Forensics and Security* ,**(2021)**,16, 3469-3478.

[50] Moser, A., Kruegel, C., & Kirda, E. (2007, May). Exploring multiple execution paths for malware analysis. *In 2007 IEEE Symposium on Security and Privacy (SP'07)* , **(2007)** , (pp. 231-245). IEEE.

[51] Guo, Y. A review of Machine Learning-based zero-day attack detection: Challenges and future directions.*Computer Communications*,**(2023)**,198, 175-185.

[52] Jadhav, A., Vidyarthi, D., & Hemavathy, M. Evolution of evasive malwares: A survey. *In 2016 International Conference on Computational Techniques in Information and Communication Technologies (ICCTICT)*, **2016** *(pp. 641-646)*. IEEE.

[53] Jamalpur, S., Navya, Y. S., Raja, P., Tagore, G., & Rao, G. R. K. (2018, April). Dynamic malware analysis using cuckoo sandbox. *In 2018 Second international conference on inventive communication and computational technologies (ICICCT) (pp. 1056-1060). IEEE*,**2018**.

[54] Ali, M., Shiaeles, S., Papadaki, M., & Ghita, B. V. (2018, October). Agent-based vs agent-less sandbox for dynamic behavioral analysis. *In 2018 Global Information Infrastructure and Networking Symposium (GIIS) (pp. 1-5). IEEE*, **2018**.

[55] Botacin, M., Ceschin, F., Sun, R., Oliveira, D., & Grégio, A. (2021). Challenges and pitfalls in malware research. *Computers & Security*, **2021**, *106*, 102287.