EECS 994 Capstone

The effectiveness of current security systems in detecting and mitigating malware and APTs.

16/08/2024

Adeyemi Fagbade
adeyemi.fagbade@ndus.edu

Abstract:

The cybersecurity landscape has evolved rapidly, with traditional file-based malware giving way to sophisticated fileless attacks and advanced persistent threats (APTs). Fileless malware, which operates without leaving traditional file traces, leverages operating system tools and trusted applications to execute malicious activities, thereby evading conventional signature-based detection methods. This stealthy approach allows fileless attacks to evade signature-based detection systems effectively. Similarly, APTs are targeted, multistage attacks designed to gain persistent access to systems while remaining undetected. A comprehensive analysis of existing literature and instances of security system failure reveals a variety of detection techniques and highlights the limitations of current systems in effectively identifying and mitigating these threats. Both fileless malware and APTs present significant challenges for current security systems due to their inherent evasiveness and ability to circumvent conventional defense mechanisms. Existing research efforts have focused on analyzing the characteristics, techniques, and life cycles of these threats to develop more effective detection and mitigation strategies. The tabled approaches range from comprehensive reviews of fileless attack vectors to proposing novel detection models leveraging machine learning. However, several key challenges remain. Fileless malware's lack of persistent artifacts and APTs' multi-stage nature make forensic analysis and trace detection extremely difficult. To enhance protection against emerging threats, future work must prioritize developing proactive, adaptive security systems capable of early detection and automated response. Multidisciplinary efforts integrating technical solutions with policy frameworks, industry collaboration, and continued research into emerging adversarial techniques will be crucial for maintaining cyber resilience for security systems.

1. Introduction

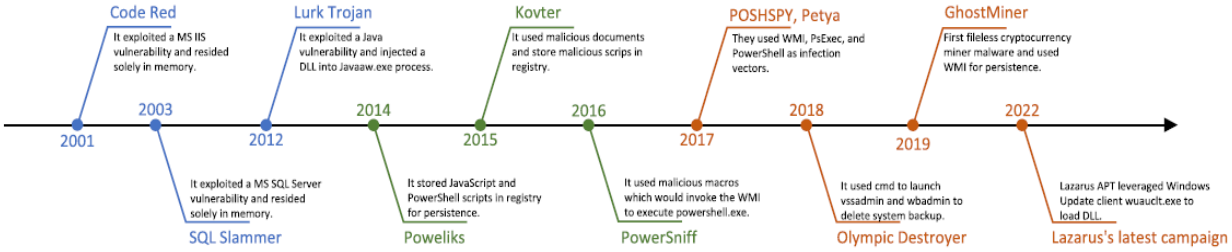
The cybersecurity landscape has evolved significantly over the past decade, with increasingly sophisticated threats like fileless malware and advanced persistent threats (APTs) challenging traditional security systems (see Figure 1). These emerging threats exploit legitimate operating system tools and trusted applications, making detection and mitigation difficult. Fileless malware, in particular, operates without leaving conventional file traces and exploits standard operating system tools to execute malicious activities. Similarly, APTs, characterized by their multistage and targeted nature, further exacerbate the difficulty of detection and mitigation. The inadequacy of the usual signature-based methods against these sophisticated attacks necessitates a comprehensive examination of current security systems and the development of innovative mitigation strategies.

Research in this domain has progressed from early anomaly detection techniques to more advanced approaches utilizing machine learning, data mining, and big data analytics. Early research, such as the work by Idika and Mathur [1], emphasized the importance of anomaly detection techniques for identifying previously unknown malware. They proposed a framework combining machine learning algorithms with static and dynamic analysis to enhance malware detection capabilities. However, the complexity of attacks has evolved beyond early security designs. CrowdStrike [2-4] reports that eight out of ten common attack vectors resulting in successful data breaches employed dynamic fileless attack methods. In fact, an estimate from the Ponemon Institute indicates that fileless attacks have a success rate roughly ten times higher than traditional attack methods. Barr-Smith et al. [5] highlighted that 26.26% of APT activities utilize fileless attack methods, particularly the sub-technique known as Living-off-the-Land.

As the complexity and proliferation of malware and APTs grew, researchers explored more advanced approaches. In 2010, Siddiqui et al. [6] investigated the use of data mining techniques for malware detection, highlighting the

effectiveness of feature selection and classification algorithms in identifying malicious code patterns. Rieck et al. [7] introduced a framework called Drebin, which employed static analysis and machine learning to detect Android malware, achieving high detection rates.
However, the increasing sophistication of APTs have undoubtedly prompted further research towards specialized detection and mitigation strategies. In 2013, Virvilis and Gritzalis [8] conducted a comprehensive survey of existing APT detection techniques, emphasizing the need for multi-layered defenses and the integration of multi-stages detection methods. They highlighted the importance of combining signature-based, anomaly-based, and behavioral analysis approaches to effectively combat malware and APTs. As a result, recent research has focused on leveraging advanced machine learning techniques and big data analytics to enhance the effectiveness of security systems. Kwon et al. [9] proposed a deep learning-based approach for detecting and classifying malware, demonstrating high accuracy and the ability to identify previously unseen malware variants. Berman et al [10]. explored the use of graph neural networks for detecting malicious activity in network traffic, showing promising results in identifying complex attack patterns and reducing false positives.

Despite these advancements, current security systems still struggle to keep pace with rapidly evolving threats. Aghakhani et al. [11] identified limitations in existing approaches, particularly when confronted with adversarial attacks. The shift towards behavior-based detection methods and integrated machine learning techniques has shown promise but requires further refinement and validation. Sudhakar and Kumar's 2021 work [12] provided a comprehensive survey on fileless malware, exploring its reliance on security system memory rather than traditional storage mediums. They also identified significant gaps in existing detection mechanisms and proposed also a shift towards behavior-based detection methods. In addition, Borana et al. [13] developed an assistive tool designed to detect fileless malware, using heuristic and behavior-based detection techniques. Liu et al.'s [14] comprehensive survey categorized detection methods into signature-based, heuristic, and behavior-based and advocated for the integration of machine learning models with behavior-based detection to enhance detection capabilities. However, they also noted the need for continuous update and training of these models to keep pace with the evolving threat landscape.



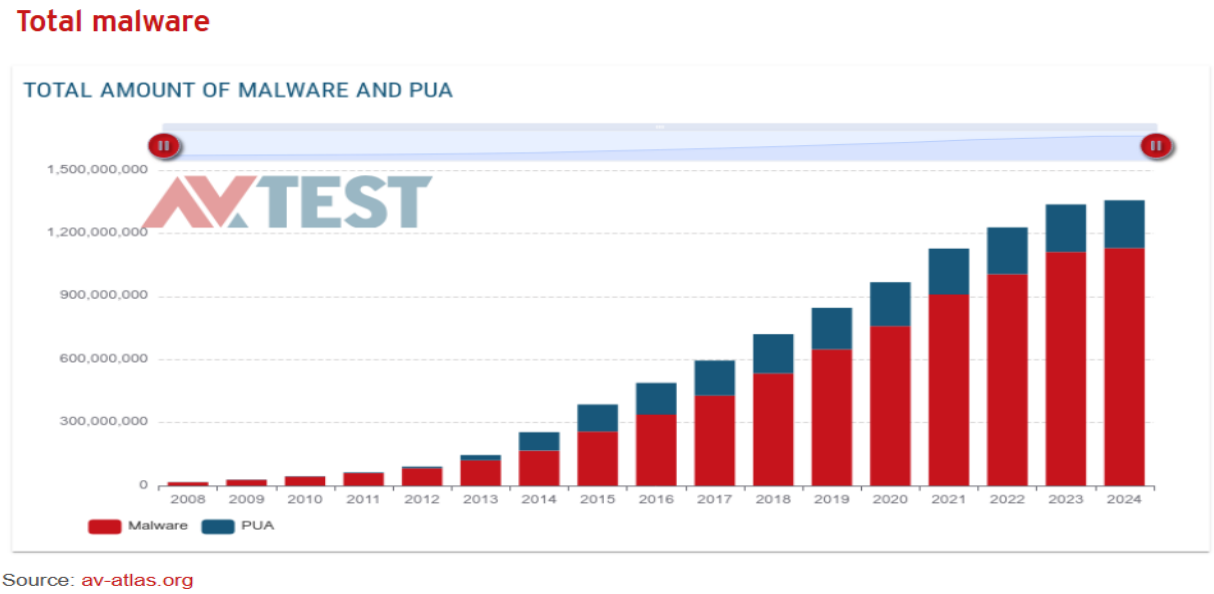


Figure 1: Notable evolution of classic fileless attack cases and trends [14].

Lately Andy Greenberg's account [15] of the NotPetya cyberattack underscored the inadequacies of traditional security measures and the need for a multifaceted approach to cybersecurity. Han et al. [16] later introduced UNICORN, a runtime provenance-based detection system for combating APTs, which significantly improved the detection of stealthy APT activities. By leveraging detailed provenance data, UNICORN framework aimed to detect and block APTs before they could even establish a foothold. However, they acknowledged the challenge of managing the vast amounts of data generated by provenance tracking and the need for efficient data processing algorithms. In another perspective, Hossain et al. [17] focused on forensic analysis techniques to trace and understand the behaviors of APTs and fileless malware, and hence developed a dependence-preserving data compaction method for more scalable forensic analysis and facilitating quicker identification of malicious activities and their origins. Afianian et al.'s survey [18] on malware dynamic analysis evasion techniques provided insights into methods employed by malware to avoid detection. Typical strategies include obfuscation, encryption, and environmental awareness, which allow malware to operate undetected during dynamic analysis. They asserted that to mitigate these evasion techniques, security systems must incorporate advanced analysis tools capable of emulating diverse environments and behaviors. They emphasized the importance of continuous research and development to stay ahead of adversaries' evolving tactics. Hence, the remainder of this term's report will delve deeper into the intricacies of fileless malware and APTs, providing a comprehensive analysis of their persistent techniques and evolving evasion strategies. We will explore current detection and mitigation approaches, evaluating their effectiveness and limitations. The report will also examine recent case studies of fileless malware attacks, highlighting detection potentials and areas for improvement. Recent cases of fileless malware and their detection potentials will be discussed, providing insights into the effectiveness of current security systems and proposing innovative solutions to enhance their capabilities in combating these evolving threats. Hence, this analysis aims to enhance cybersecurity solutions, focusing on the Windows platform (see Figure 2). We address three key research questions:

- M1: Are fileless attacks truly file-free?
- M2: What challenges do current detection methods face in identifying fileless attacks?
- M3: How can these challenges be addressed to improve detection for security systems?

By exploring these questions, we seek to deepen our understanding of fileless attack mechanisms and detection strategies. Our contributions include:

- A comprehensive review of fileless attack concepts and technological developments.
- An examination of malware evasion and propagation techniques used for target identification and stealth.
- An analysis of fileless malware behavior and persistence mechanisms.

This research aims to provide insights that can lead to more effective cybersecurity measures against fileless attacks.

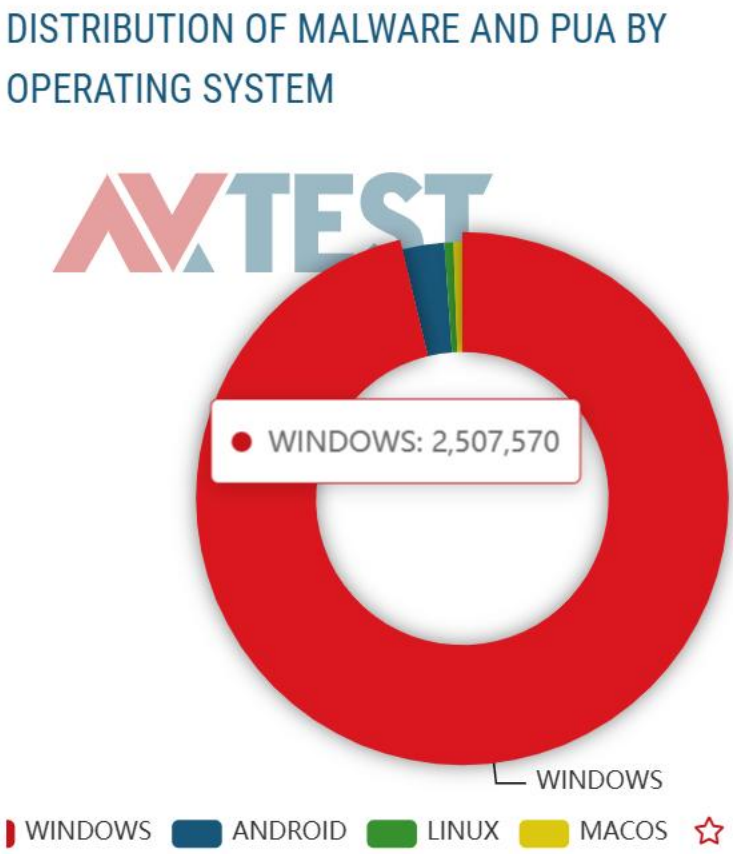


Figure 2: The figure illustrates the total stock of malware and PUA collected by AV-TEST, categorized by the frequency of occurrence per operating system. Extract from AV-ATLAS - Malware & PUA

1.1 Background of fileless malware and APTs:

Fileless malware differs from traditional file-based attacks by exploiting legitimate system processes and built-in tools rather than using distinct malicious executables. These trusted components, often referred to as 'Living off the Land Binaries (LOLBins)' [19] are leveraged to execute attacks and maintain stealth. Table 1 provides a detailed comparison between traditional file-based malware and fileless malware, highlighting their key differences in approach and execution.

Techniques	Traditional file-based malware	Fileless malware
Source code	Yes	No
Malicious file	Yes	No
Malicious process	Yes	No (Uses trusted OS processes)
Complexity	Moderate	Very high
Detection complexity	Moderate	Very high
Persistence	Medium	Low
File Types	• Executable files • Script embedded in a format that executes scripts (PDF, Word, Excel etc.,)	• JavaScrpt • WMI • PowerShell • Flash • WScript/ CScript
Targets	Executable file with single targeted OS/ patch level combination	Can target many different OS/ path level combinations
Obfuscation methods	• Encrypt file • Archive file • Executable file disguised as another type of file • Executable file embedded in another file	• Encoding • Escaped ASCII/ Unicode values • String splitting • Encryption • Randomization • Data obfuscation • Logic structure obfuscation • White space
Anti-virus detection	Possible with known signature	Not possible
Sandboxes detection	Physically availability of file	Not possible
Behavior-based heuristics and unsupervised machine learning	File-based malware shows abnormal behavior in the system after compromising the targeted host. Hence, these systems are designed to detect such behavior.	Fileless attacks are designed to behave like a benign process in the system, so they may not alarm as an anomaly. Hence, very difficult to detect.

Table 1: Comparison between file-based malware and fileless malware [12].

1.2 Definition of fileless malware:

Malware, short for malicious software, refers to harmful programs designed to target digital devices. Traditional malware typically creates executable files that damage systems or steal information, often communicating with command and control (C&C) servers through randomly generated IP or URL addresses or processes. Fileless malware attacks, however, represent a more sophisticated threat. These attacks don't rely on downloading malicious files or writing content to the disk. Instead, they exploit vulnerable applications to inject malicious code directly into main memory or leverage trusted applications and native administrative tools (like Microsoft Office, PowerShell, or Windows Management Instrumentation) to run scripts and load harmful code into volatile memory. Hence, the concept of fileless attacks has evolved over time, with varying definitions across the cybersecurity industry:

- Some security vendors describe fileless malware as infections using legitimate programs without leaving traces, Trellix [20].
- Others define it as malicious activities using built-in system tools without installing code on the target system, CrowdStrike [3].
- Some companies have used terms like "bodiless malware" or "non-malware attacks" to describe this phenomenon.

A comprehensive definition of a fileless attack could be:

An attack that doesn't rely on executable files for malicious functionality. The payload is executed directly in memory through various techniques, leveraging system memory, services, legitimate binaries, and trusted applications to carry out the attack, Side et al [14]. This characterization makes fileless attacks particularly

challenging to detect and remove, as they don't leave traditional malware traces on the system, highlighting the need for advanced cybersecurity measures. In particular, the execution is implicit and methodological.

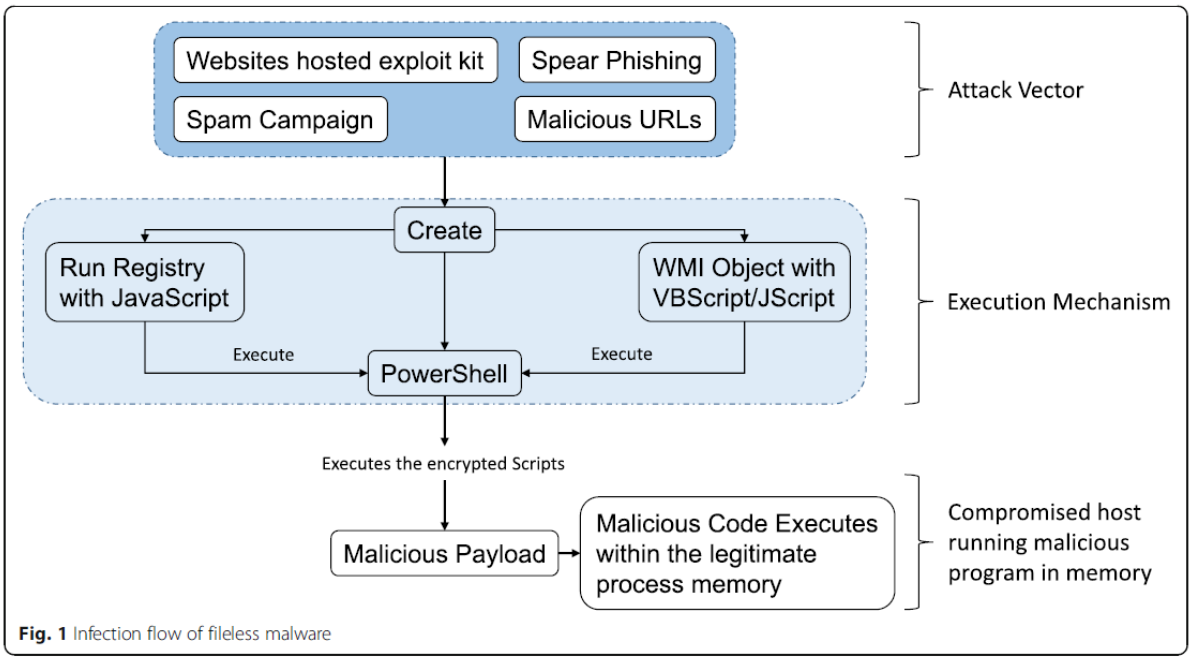


Figure 3: Infection flow chart for fileless malware [12].

1.3 Execution Of Fileless Malware:

In recent years, cybercriminals have increasingly adopted sophisticated techniques to evade traditional antivirus (AV) solutions. One such approach involves leveraging legitimate Windows applications, particularly Windows Management Instrumentation (WMI) and PowerShell, to execute malicious code without leaving easily detectable traces on the file system [21-22]. This method, known as fileless malware, has become a significant concern in the cybersecurity community. The life cycle of fileless malware typically unfolds in three distinct phases:

- **Attack Vector**: This initial stage involves the methods through which attackers target their victims. Common vectors include phishing emails, compromised websites, or exploitation of software vulnerabilities, Korolov [23].
- **Execution Mechanism**: Once the attack vector succeeds, the malware initiates its execution process. This may involve creating registry entries for persistence, establishing WMI objects with VBScript or JScript, or invoking PowerShell instances. These actions are designed to maintain a foothold in the system while minimizing detectability, Pontiroli & Martinez [24].
- **Payload Delivery**: In this phase, PowerShell or other native Windows tools are utilized to execute the malicious payload directly in the memory of legitimate processes. This approach circumvents traditional file-based detection methods, as no malicious files are written to the disk, Gorelik & Moshailov, [24].

The infection process, illustrated in Figure 3, highlights the sophisticated strategies employed by fileless malware to compromise systems without leaving a detectable footprint on the file system.

1.3.1 PowerShell:

PowerShell, a powerful scripting language and command-line shell developed by Microsoft, offers a wide array of features that can be exploited by malicious actors. Its versatility and deep integration with the Windows operating system make it an attractive tool for cybercriminals seeking to evade detection by antivirus (AV) solutions, establish persistence on compromised systems, or conduct covert surveillance operations, Pontiroli & Martinez [24]. One of the key advantages that attackers leverage is the fact that many PowerShell modules are inherently trusted by the Windows operating system. This trust allows malicious scripts to operate with a degree of legitimacy, potentially bypassing security measures that might otherwise flag suspicious activity, Gorelik & Moshailov [25]. Advanced evasion techniques have been developed to further exploit PowerShell's capabilities. These methods often involve the dynamic loading of scripts directly into system memory, a process that leaves no trace on the file system. The ability to execute code entirely in memory, without writing to disk, represents a paradigm shift in malware operations. It allows attackers to maintain a presence on compromised systems while minimizing their observable footprint, making detection and forensic analysis considerably more difficult for security professionals.

1.3.2 Windows Management Instrumentation

Windows Management Instrumentation (WMI) has been a persistent and powerful feature of the Windows operating system since its introduction in Windows NT 4.0 and Windows 95 [12]. Its extensive capabilities have made it a valuable tool for system administrators, but also a potent weapon in the arsenal of cybercriminals. WMI's versatility allows it to be leveraged across multiple stages of an attack lifecycle, including initial reconnaissance, detection evasion, code execution, lateral movement within networks, covert data storage, and maintaining persistence on compromised systems. These features enable attackers to create sophisticated backdoors that operate entirely in memory, without leaving telltale files on the target system, Graeber [21]. Furthermore, WMI can be used to execute malicious JavaScript or VBScript directly in memory, potentially bypassing traditional antivirus solutions that rely primarily on file-based detection methods. As a result, WMI remains a significant concern for cybersecurity professionals, requiring constant vigilance and the development of advanced detection and prevention strategies.

2. Methodology:

This study aims to assess the effectiveness of current security systems in detecting and mitigating fileless malware and advanced persistent threats (APTs). To address this, we conducted a comprehensive review and comparative analysis of existing literature and research publications. The research questions guiding this study are: (1) Are current security systems effectively designed to detect and withstand fileless malware and APTs? (2) What are the challenges of existing detection methods in identifying fileless attacks? Given the complexity and specialized nature of these questions, secondary data from existing scholarly articles, industry reports, and case studies (of fileless malware samples) were deemed appropriate. Hence, a significant amount of dataset, largely from windows operating systems, with various examples to provide resources for comparable investigations were taken from https://ilkerkara.karatekin.edu.tr/RequestDataset.html. These sources provide extensive insights and empirical evidence from various security system implementations and their performance against malware and APTs.

2.1 Theoretical (Exploratory)Approach:

In the initial phase of this research, we conducted a systematic literature review (SLR) and a logical exploration of cyber-attack on existing security systems, delving into their emergent behavior when faced with various forms of traditional and fileless malware attacks, as well as examining their underlying assumptions. Subsequently, we will collect and analyze relevant literature on existing cyber systems to gain a comprehensive understanding of the current state of security systems. The SLR approach is particularly appropriate for this research as it allows for a comprehensive analysis of existing literature, synthesizing findings from multiple studies to provide a holistic view of the current state of security systems [25]. This method is crucial in identifying gaps in current knowledge and understanding the challenges faced by existing detection methods.

2.2 Observational (Descriptive) Research:

Case Studies: Examine documented instances of fileless malware and APTs to understand how current systems respond or behave in the incident response process to generate and overcome the attack. The data collection process involves systematically searching relevant databases using specific keywords related to fileless malware, APTs, and security system effectiveness. Articles were then filtered based on the selection criteria, and relevant data were extracted and organized for analysis. Tools such as EndNote were used for reference management, and NVivo & excel will be employed for qualitative and quantitative(malware) data analysis respectively. This process facilitated a structured and comprehensive review of the existing literature, allowing for the identification of common themes, challenges, and gaps in the current detection methods for fileless malware and APTs. The use of recent and relevant literature ensured that the findings are up-to-date and reflective of the current state of cybersecurity. This methodology not only addresses the research questions comprehensively but also provides valuable insights for future research and development in the field.

For malware analysis of the provided fileless attack cases, one can adopt dynamics or static analysis, which involves collecting information about harmful software before its execution. This technique of analysis may be used to determine whether a suspicious file includes malware and employs a variety of properties derived from the raw bytes of portable executable files (PEs) such as texts, opcodes, API calls, byte arrays, and control flowcharts Gibert et. al. [29]. Similarly, pattern recognition analysis may also be used to find anomalies in malicious events. A malicious cluster can help to categorize the attack patterns and attack reconstruction to uncover the motivation and intention of the attacker.

2.3 Taxonomy of fileless malware

The deployment and propagation of fileless malware attacks can be categorized into three distinct types, based on the system hierarchy they leverage, as shown in Figure 4a[14]. These types are:

1. Memory-based
2. Registry-based
3. Living off the Land (LotL)-based

Additionally, LotL-based attacks can be further subdivided into: a) Binary-based b) Script-based. For the preview, we would only discuss the commonly seen type-memory-based.

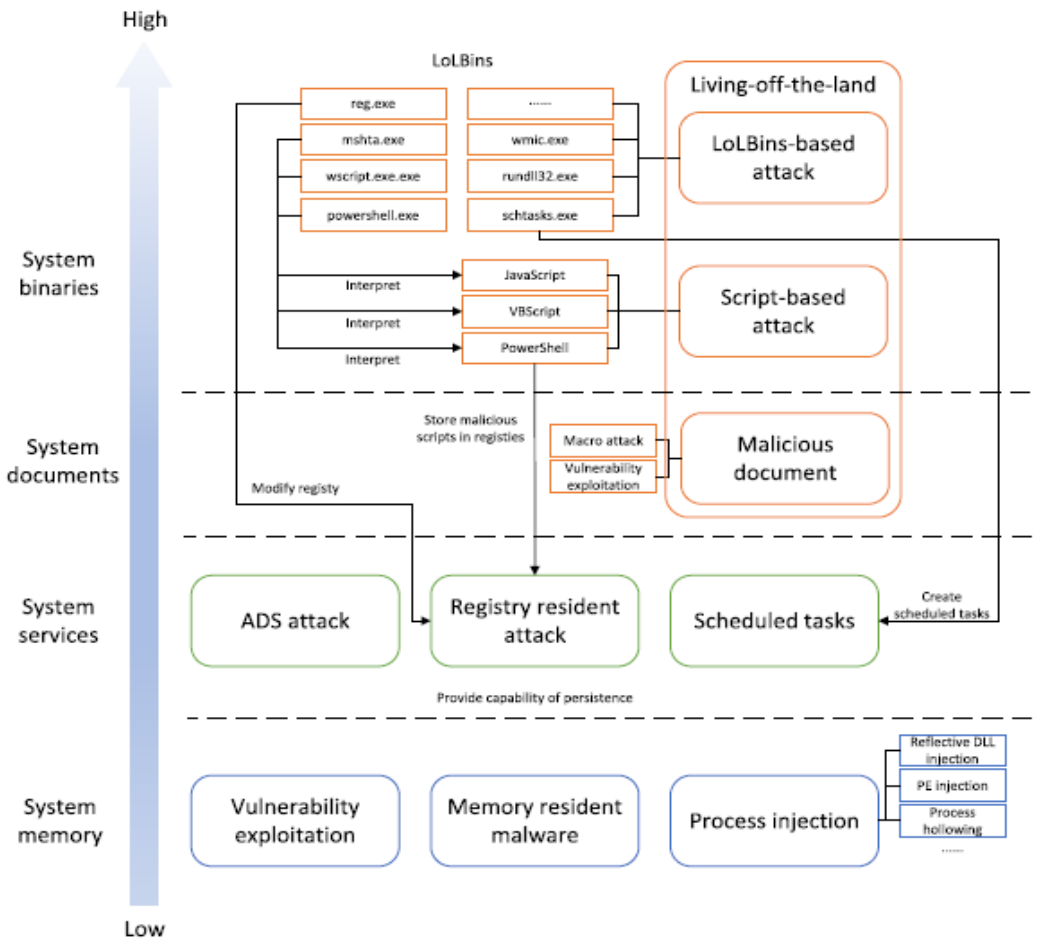


Figure 4a: Hierarchies of system leverage in fileless malware attacks[14].

2.3.1 Memory-based fileless attack

Memory-based fileless attacks represent a sophisticated cyber threat where malicious code operates exclusively within a system's RAM, leaving no trace on the hard drive. This approach allows threat actors to evade traditional security measures that rely on file-based detection.

In a typical scenario, attackers exploit system vulnerabilities to gain initial access to the target environment. Once inside, they deploy and execute malicious payloads directly in memory, circumventing the need to write data to disk. This method significantly reduces the attack's detectability, as many security solutions focus on scanning files stored on hard drives. Several techniques can be employed to achieve memory-based fileless attacks:

- Process Injection: Attackers inject malicious code into legitimate running processes, leveraging their existing permissions and masking the malicious activity.
- PowerShell Exploitation: Threat actors utilize PowerShell's powerful scripting capabilities to execute malicious commands directly in memory.
- Reflective DLL Loading: This technique involves loading a DLL into a process's memory without registering it with the system, making it harder to detect.
- In-Memory Code Execution: Attackers may use programming languages that support dynamic code execution, such as Python or JavaScript, to run malicious code directly in memory.
- Living off the Land Binaries (LOLBins): Attackers misuse legitimate system tools and binaries to execute malicious actions, blending in with normal system operations.

2.3.1.1 Vulnerability exploitation

Vulnerability exploitation is a critical vector for fileless attacks, with zero-day exploits representing the most severe threat. Zero-day attacks target undisclosed vulnerabilities, leaving defenders with minimal time to react and protect systems. These attacks are particularly dangerous because traditional signature-based antivirus solutions are ineffective against them.

Attackers often leverage vulnerabilities during the initial stages of an intrusion. Common vulnerability types include:

- Buffer overflows: Occur when data exceeds a program's allocated buffer size, potentially overwriting adjacent memory.
- Format string vulnerabilities: Arise from improper handling of user-supplied format strings in certain functions.
- Integer overflows: Result from arithmetic operations producing a value too large for the intended data type.
- Use-after-free vulnerabilities: Involve accessing memory after it has been freed, potentially leading to arbitrary code execution.

To illustrate, consider a buffer overflow attack: An attacker intentionally provides input exceeding a program's buffer capacity. This overwrites adjacent memory, potentially replacing critical data such as return addresses. When the function returns, execution flow is redirected to attacker-controlled code, enabling arbitrary execution.

Vulnerability exploitation often involves crafting specific inputs that cause programs to mishandle data, leading to unintended behavior. This can result in:

1. Privilege escalation: Gaining higher-level permissions than intended.
2. Information disclosure: Unauthorized access to sensitive data.
3. Denial of service: Disrupting normal system operations.
4. Remote code execution (RCE): Running arbitrary code on the target system.

2.3.1.2 Memory resident malware

Memory-resident malware represents a sophisticated class of threats that operate entirely within a system's RAM, avoiding interaction with the file system. This characteristic makes such malware particularly challenging to detect using conventional antivirus software or security tools, which often rely on file-based scanning methods [12].

Several notable examples of memory-resident malware have emerged over the years, demonstrating the evolution of this threat. Code Red, an early memory-resident worm exploited a vulnerability in Microsoft's IIS web server. It propagated through TCP/IP connections on port 80, injecting itself directly into the memory of vulnerable systems. The worm used a buffer overflow technique, sending a string of repeated 'N' characters followed by a malicious payload. Another pioneering memory-resident worm, SQL Slammer was remarkably compact at just 376 bytes. This allowed it to fit within a single network packet, facilitating rapid propagation. The worm targeted a vulnerability in Microsoft's SQL server, sending malformed requests to UDP port 1434. Infected systems would then broadcast the malicious code to random IP addresses, potentially leading to distributed denial-of-service (DDoS) attacks. Recently Duqu 2.0 has emerged. In 2015, Duqu 2.0 represented a significant advancement in memory-resident malware. It utilized a zero-day vulnerability for initial infection and remained exclusively in memory. Duqu 2.0 was modular, with capabilities ranging from a basic remote backdoor (about 500KB) to a comprehensive network espionage platform (up to 18MB). Its modular design allowed for the dynamic loading and execution of various espionage-focused components, all operating within system memory.

The operational strategy of memory-resident malware typically involves exploiting vulnerabilities for initial access, followed by continuous operation within RAM. However, this approach has a significant limitation: the lack of persistence. Since the malware resides only in volatile memory, a system restart will effectively eliminate it.

2.4 Detection techniques for fileless malware& mitigation:

Malware detection methods are crucial for protecting computer systems against data loss and vulnerabilities caused by malicious software. These methods include signature-based, behavior-based, model-based, and heuristic-based detection, as well as newer techniques utilizing deep learning, cloud computing, mobile devices, and Internet of Things technologies.

While signature-based detection is effective for known viruses, it struggles with unknown malware. In fact, fileless malware poses a particular challenge due to the absence of files to evaluate. Hence, the development of comprehensive secure systems remains difficult due to the rapid evolution of malware and the need for ongoing research [30].

Various types of malware analysis and detection techniques (or tools) exist as shown in Figure 4-5 and Table 2, each with its own strengths and limitations. The effectiveness of these methods varies depending on the nature of the threat and the specific context of the system being protected.

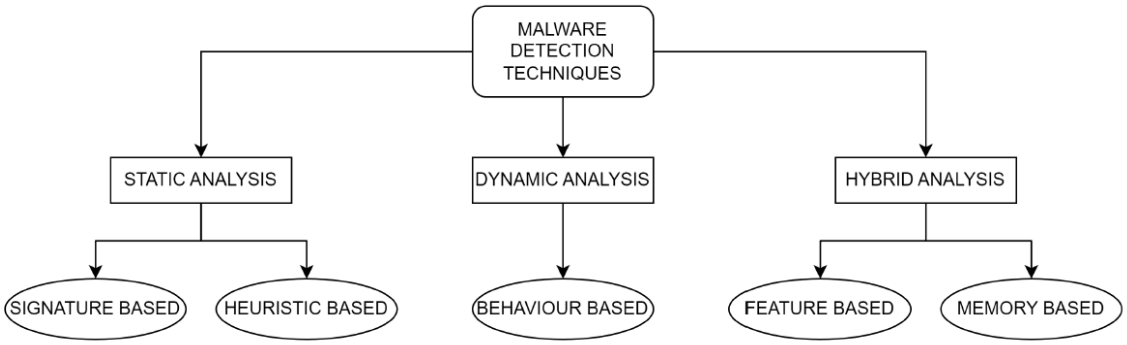


Figure 4: Malware Detection Techniques.

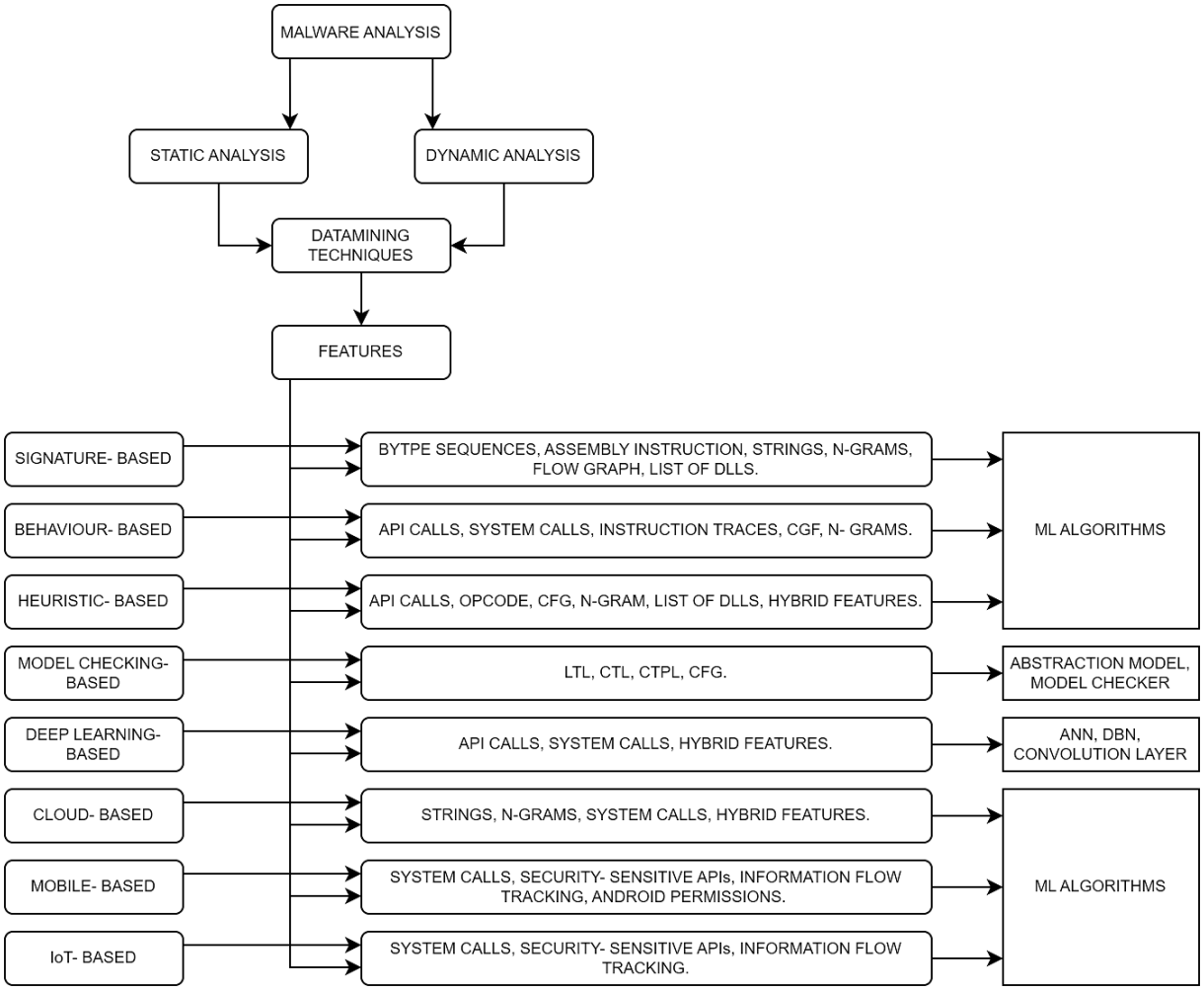


Figure 5: Malware Analysis schema [30].

Tool	Focus	Features	Limitations
GRR	Incident Response, Live Forensics	Remote Investigation, Scalable, Automation	Learning Curve, Configuration
Wireshark	Network Protocol Analysis	Packet-level Analysis, Wide Protocol Support	Networking Expertise, Limited Malware Coverage
VirusTotal	Web-based Malware Analysis	Aggregate Results, Web Interface, Community-driven	Relies on Signatures, Limited to Files and URLs
Suricata	Network IDS/IPS	High Performance, Multi-threaded, Rule-based	Configuration Expertise, Network-focused
Snort	Network IDS/IPS	Rule-based Detection, Regular Updates	Regular Rule Updates, Network-focused
Sophos Intercept X	Endpoint Protection	Real-time Protection, Exploit Prevention, Centralized Management	Endpoint-specific, Limited Network Coverage

Table 2: Malware Detection Tools and Comparison.

3 Comparative analysis

The evolving landscape of cyber threats, particularly the spread of fileless malware, presents ongoing challenges to digital security. Thus far, many research have been conducted on detection and analysis of fileless malware, but

more still need to be done since security system does not recognize unusual behaviors such as suspicious login activities, odd working hours, or abnormal network traffic etc. Hence, organizations must develop and implement effective strategies to counter these risks. This section evaluates current research on fileless malware and APT detection and mitigation by examining selected research studies. The review focuses on approaches and methodologies used, key analyses and conclusions, identified limitations and suggestions for future research. By assessing these aspects, we aim to provide a comprehensive overview of the current state of malware defense research and highlight areas for potential improvement.

3.1 Related work:

Idika et al., [1], provided a comprehensive survey of the array of techniques deployed for malware detection covering both traditional and emerging methods such as static and dynamics analysis method. The authors traversed signature-based, behavior-based, and hybrid approaches, evaluating their merits and limitations. They also delved into the utilization of machine learning techniques, like decision trees, neural networks, and support vector machines, for malware detection. The resulting publication includes the assessment of the strengths and weaknesses of each examined technique, considering factors like detection accuracy, performance overhead, and adaptability to new threats. The survey concluded by highlighting the ever-evolving nature of malware, advocating for perpetual advancement in detection techniques and the incorporation of a multi-layered approach, combining various detection methods to create a robust malware defense system. However, their report has insufficient coverage of emerging threats such as fileless malware and APT. Hence, future work should incorporate the emerging threats such as fileless malware into the detection techniques.

Similarly, Talukder et al. [31] conducted an extensive review of malware detection and analysis techniques. Their study began by explaining the importance of effective malware detection and outlining the associated challenges. The authors then examined various methods, including static analysis techniques, dynamic analysis approaches, signature-based detection, behavior-based identification, sandboxing, memory analysis, and machine learning applications in malware detection. After evaluating these methods, the researchers concluded that a comprehensive approach combining multiple techniques is necessary for effective malware detection and analysis. They emphasized that no single method is sufficient to address the complex and evolving nature of malware threats. The work provides valuable insights into the current state of malware detection research and highlights the need for integrated strategies in cybersecurity.

In 2017, Bulazel and Yener [26] focus on the challenges posed by general class of malware, including wild and fileless malware that can evade automated dynamic analysis systems. Importantly, the paper examined the dynamic analysis techniques, to detect malware and the evasion strategies employed by malware developers to bypass these techniques. Emphasis is placed on sandboxing and behavior monitoring to detect malware. They emphasize the strategies malware uses to evade these techniques. The report analysis includes a categorization of evasion techniques, such as environment detection, timing-based evasion, and anti-debugging methods. The authors also assess the prevalence and effectiveness of these techniques in real-world malware samples. They also evaluate various counter-evasion strategies, analyzing their strengths and weaknesses. Main conclusion as usual beam a search light on the evasiveness of sophisticated malwares and emphasis the need for continuous innovation in counter-evasion methods to keep pace with evolving malware capabilities. One limitation of the research is the fact that it does not review the blend and integration of static and dynamics techniques to confront the propagation of malware transmission and improve detection effectiveness.

Like the above research papers, Souri et al., [32], created a sweeping survey of malware detection methodologies that harnessed data mining techniques. The authors dissected the limitations of conventional signature-based approaches and pivoted to data mining paradigms such as machine learning and clustering. They juxtaposed the merits and demerits of these approaches, presented a taxonomy of malware detection methods, and elucidated

ongoing research challenges. This comprehensive review serves as a compass for those interested in the frontier of malware detection via data mining strategies.

Now, with the emergence of fileless malware attack, Liu et al [14] focus on the on the emerging threat of fileless malware, which operates primarily in memory and exploits legitimate system tools. The authors explore the evolution of fileless attack techniques and mechanisms, and their corresponding detection methods. The research involves a systematic review of literature on fileless malware, including academic papers, industry reports, and real-world case studies. The authors also analyzed trends in fileless attack techniques over time. The analysis includes a classification of fileless attack methods, such as exploiting PowerShell, using living-off-the-land binaries, and leveraging in-memory execution while examining the detection capacity of the standard methods like behavior analysis, memory forensics, and machine learning approaches, assessing their effectiveness and success rate against fileless threats. They were able to identify the limitations of traditional file-based detection methods and stress the need for advanced memory-based and behavior-based detection techniques to combat fileless malware effectively. However, the paper lacks practical implementation details for proposed detection methods. Perhaps, their next research should focus on the developing practical implementation strategies for the proposed detection methods.

In a similar research trend, Ilker Kara [33] examined fileless malware threats and the use of memory forensics for detection and analysis. The author conducted a literature review on fileless malware and memory forensics techniques such as code injection, process hollowing, and reflective DLL loading and then evaluated case studies and performed empirical analyses of real-world fileless malware samples. Significant remarks of the analysis include the high level of false-positive behavior detection of the traditional techniques of detection. The author infers that harmful code elements might be identified using applications like Yara, which identifies malware without downloading it. The paper concludes that memory forensics is a crucial tool in combating fileless malware but faces significant challenges due to the volatility of evidence and the complexity of distinguishing between legitimate and malicious in-memory activities. The author identifies key research challenges and proposes potential directions for future work in this area. Graeber [34] utilized Windows Management Instrumentation (WMI) and PowerShell to detect Windows-based fileless malware, mirroring the approach taken by Kara [33]. The use of PowerShell allowed the authors to demonstrate an effective method for identifying fileless malware. Due to the ability of PowerShell scripts to access data from various sources like the file system or registry, they can be dynamically presented in memory [26], thereby thwarting attackers from persisting or spying on the target system.

Additionally, it is important to highlight the unique techniques discussed in various papers on fileless malware detection, such as those mentioned by Graeber [34]. Rivera & Inocencio [35], for instance, recommended using program tools to detect memory and script-based fileless malware attacks. They discovered a novel method for identifying fileless malware and demonstrated that achieving 100% detection accuracy while maintaining scalability is challenging. By employing the approach described by Graeber [34], they aimed to address this issue through the development of a memory analysis model for Windows-based fileless malware attacks, achieving a high success rate with this strategy.

In a mini assessment of the mentioned papers, one can observed that the selected papers provide a comprehensive overview of current challenges in malware detection and analysis, with a particular focus on emerging threats like fileless malware and advanced evasion techniques. They collectively highlight the limitations of traditional detection methods and emphasize the need for more sophisticated approaches, including memory forensics, behavior analysis, and machine learning techniques. Hence, future research work derived from these publications include:

- developing more resilient dynamic analysis techniques that can withstand advanced evasion methods.
- improving memory forensics tools for better detection and analysis of fileless malware.
- exploring the application of advanced AI and machine learning techniques in malware detection.
- investigating new methods for real-time detection of fileless malware in enterprise environments.

- developing comprehensive frameworks that combine multiple detection techniques to create more effective security systems.

Similarly, the observed limitation (or research constraint) as identified in the selected publications include

- lack of sufficient empirical data on the real-world effectiveness of proposed detection techniques.
- limited discussion of the practical implementation challenges for advanced detection methods in enterprise environments.
- insufficient exploration of the legal and ethical implications of advanced malware detection and analysis techniques.
- lack of consideration for the human factor in malware detection and response processes.
- and lastly, absence of quantitative comparisons between different detection techniques in terms of performance and accuracy.

These deficiencies could be addressed in future research to provide a more comprehensive understanding of effective malware detection and mitigation strategies towards better security systems. However, the above comparative analysis underscores the complexity of detecting and mitigating malware and APTs, particularly fileless malware. While current security systems have made strides in improving detection, continuous research and development are necessary to address the evolving threat landscape. Hence, future work should focus on integrating various detection techniques, leveraging advanced technologies like AI and machine learning, and enhancing forensic tools to provide comprehensive security solutions.

3.2 Selected types and cases of malware attacks and security systems responses

In this section, we survey cases of fileless attacks against deployed analytic and security systems. Note that as with rest of this survey, we only cover academic offensive and defensive works, so discussed statistics may not be representative of the threat landscape observed in industry. Offensive actors creating malware are likely to be more aggressive in conducting fingerprint discovery efforts than academic researchers. Defensive assessments of malware evasion are likely better understood by industry antivirus and endpoint protection companies which face evasive malware daily.

3.2.1 Kovter & Poweliks

Kovter and Poweliks represent advanced fileless malware that have evolved significantly since their initial appearances [38-39]. These threats primarily use registry keys for persistence, making them challenging to detect and remove using traditional antivirus solutions.

Kovter, originally a click-fraud Trojan, has developed into a sophisticated fileless threat. It typically infects systems through malicious email attachments or exploit kits. Once executed, it uses JavaScript and Windows tools like Mshta to inject malicious code into the registry. Kovter variants may employ various Windows processes (such as WScript, CMD, MSHTA, and PowerShell) to execute their payloads. Some versions are fully fileless, while others may write interpreted scripts to disk.

Poweliks, first identified in 2014, pioneered innovative registry-based persistence methods. Modern Kovter variants have adopted many of Poweliks' techniques to enhance stealth and evade detection. Both malware families utilize similar tactics, including registry manipulation, local code injection, and the use of legitimate Windows processes to execute malicious code. While sometimes they may download additional payloads, they often maintain a fully fileless presence focused on click-fraud or other malicious activities. In mid-2022, a significant Poweliks attack targeted healthcare institutions. The malware was delivered through malicious email attachments disguised as documents from trusted sources. Upon execution, the malware used PowerShell commands to write its code into the registry, ensuring it would run every time the system started. Poweliks then established communication with command-and-control servers to receive further instructions and download additional payloads, including

ransomware. This form of attack traditionally disrupted operations by encrypting critical data and demanding ransom payments, demonstrating the severe impact of fileless malware on critical infrastructure.

3.2.2 SMB - EternalBlue – DoublePulsar / WannaCry

The SMB exploit, famously used in the WannaCry ransomware outbreak [40], has also been employed in lesser-known fileless malware variants. This exploit demonstrates a sophisticated fileless attack chain:

- It begins with a network exploit
- Directly injects shellcode into the kernel (using DoublePulsar)
- Injects code into usermode, typically through a legitimate Windows process like lsass.exe

Up to this point, the attack remains entirely fileless. In WannaCry's case, the injected code in lsass.exe then downloaded and installed a malicious executable as a service, leveraging lsass's system privileges. However, prior to WannaCry, the same SMB exploit was used for direct credential theft from lsass.exe without downloading any files, showcasing a fully fileless attack.
Key fileless techniques employed in these attacks include:

1. CreateRemoteThread: Injecting a new thread into the usermode lsass.exe process directly from the kernel.
2. Asynchronous Procedure Calls (APC): Some variants use existing threads in an alertable state to execute malicious shellcode.
3. Network-to-Kernel Code Injection: Malicious shellcode is injected directly into the kernel via SMB packets without writing files to disk.

3.2.3 DNSMessenger & Meterpreter Injection

DNSMessenger and Meterpreter are fully fileless attacks used by many advanced groups. DNSMessenger is basically a fileless method for delivering malicious code by utilizing the DNS network protocol. Malicious commands are delivered directly into the memory of the running process. Meterpreter is also a method of delivering malicious code through network directly into the memory of the running process, although usually utilizing TCP or HTTP protocols. While Meterpreter is widely used by many pentesting frameworks, both methods have been adopted by advanced groups like FIN7 and others. Methods utilized by those attacks are:

• Reflective loading – Meterpreter injection usually works by injecting a DLL code into the process directly from network and then remapping the DLL inside the process. The method requires the shellcode to identify all the required functions in the process and remap those addresses into the injected DLL.

• Local injection – In the case of the DNS messenger, a shellcode is injected into the running process by the same running process after receiving it from the DNS TXT records (using VirtualAlloc). This makes for stealthier execution and evasion from behavior detection solutions.

3.2.4 Packers

Although packing is a legitimate way to compress executable, essentially, it's in-memory self-modifying code that alters the memory state of the process. The same technique is utilized by many malware families for signature re-creation and more importantly – behavior detection evasion. Overall packing can also be used as a method for code injection by rewriting the existing executable and recreating its code after decryption and remapping of the new functionality. Malware likes to hide their real API and functionality with encryption of the functions and execution of a position independent code (shellcode). The same code does not use much of the declared API and performs reflective loading of new malicious DLLs. We identify this technique as fileless because the result is running malicious code that was created purely in memory, without writing to the disk. Many known malware heavily utilize packing and local code injection techniques to evade static analysis, including Locky, Cerber, LockyBot, Andromeda and others.

3.3 Recent Cases:

- **The Polish TicTacToe Dropper Malware:**

The "TicTacToe dropper" is a sophisticated malware delivery system that emerged in 2023, targeting victims through phishing emails containing .iso file attachments. Its name derives from a Polish language string ("Kolko_i_krzyzyk") [41] found in its code. This dropper employs a multi-stage unpacking process to evade detection:

- An initial executable (e.g., 'ALco.exe') launches the attack.
- It then sequentially unpacks and executes obfuscated .NET PE DLL files ('Hadval.dll', 'cruiser.dll', and 'Farinell2.dll') directly in memory.

This memory-only execution makes the malware particularly difficult for traditional antivirus software to detect. The final payload varies, including remote access tools (RATs) and information stealers such as Leonem, AgentTesla, and LokiBot. The TicTacToe dropper campaign has targeted a diverse range of victims across multiple sectors, suggesting that the attackers have broad objectives. Its versatility and stealth make it a significant threat in the cybersecurity landscape.

- **Lazarus Group's AppleJeus Campaign:**

The Lazarus Group, a threat actor associated with North Korea, orchestrated a sophisticated malware campaign known as AppleJeus from 2020 to 2023. This operation primarily targeted cryptocurrency exchanges and financial institutions, employing a blend of social engineering tactics and fileless malware techniques to circumvent traditional security measures. The malware's modus operandi involved leveraging legitimate Windows tools, particularly PowerShell, to execute its payload predominantly within system memory, thereby minimizing its detectable footprint. Kaspersky Lab's research indicated that the campaign's impact extended across organizations in more than 17 countries [42]. In response to this threat, the cybersecurity community developed specialized detection mechanisms, including specific indicators of compromise and tailored rules to identify AppleJeus activity. Concurrently, financial institutions bolstered their defensive posture by implementing enhanced security protocols, such as more robust network segmentation and stricter controls on PowerShell usage.
The U.S. Cybersecurity and Infrastructure Security Agency (CISA) played a crucial role in disseminating information about this threat [43]. They issued alerts and guidance emphasizing the critical nature of multi-factor authentication implementation and regular security audits as key defensive measures. CISA's official stance underscored the persistent threat posed by North Korean state-sponsored cyber actors to the cryptocurrency and broader financial services sectors.

- **Nodersok/Divergent Fileless Malware:**

Microsoft Security Intelligence reported on a fileless malware campaign called Nodersok (also known as Divergent), which utilized legitimate tools like Node.js and WinDivert to create a proxy network for malicious activities [44]. This malware infected thousands of machines, primarily in the United States and Europe. Nodersok operated entirely in memory, making it challenging for traditional antivirus solutions to detect. Microsoft researchers noted, "The campaign is particularly interesting not only because it uses advanced fileless techniques, but also because it relies on an elusive network infrastructure that causes the attack to fly under the radar" [44]. In response, Microsoft enhanced its Windows Defender ATP with new behavior-based detection capabilities. The cybersecurity community developed advanced memory forensics techniques to identify and analyze such threats. Organizations were advised to implement application whitelisting and restrict the execution of unnecessary scripting languages. The case highlighted the need for continuous monitoring of network traffic patterns and the importance of keeping legitimate tools and frameworks updated to prevent exploitation.

- **Egregor Ransomware:**

The ransomware known as Egregor emerged as a formidable cyber threat, distinguished by its partial utilization of fileless techniques within its infection chain. This malicious software targeted a diverse array

of global organizations, with notable victims including prominent entities such as Barnes & Noble and Ubisoft [45]. Egregor's infection methodology was multifaceted, combining phishing emails, exploit kits, and fileless execution mechanisms to deploy its payload. The ransomware's capacity to operate partially within system memory presented significant challenges for initial detection efforts. In response to this evolving threat, cybersecurity firms, including FireEye (now integrated into Mandiant), developed specialized detection rules and conducted comprehensive analyses of Egregor's techniques. Their research highlighted the ransomware's sophisticated blend of conventional and innovative evasion tactics, which complicated both detection and analysis processes [46]. Law enforcement agencies played a crucial role in countering the Egregor threat. Europol coordinated international efforts to disrupt the ransomware's infrastructure, culminating in arrests in Ukraine in February 2021. These operations targeted individuals suspected of providing Egregor as a ransomware-as-a-service offering to other cybercriminals, reportedly profiting from a percentage of the ransom payments [47]. The Egregor incidents catalyzed a widespread reassessment of ransomware preparedness among organizations globally. This reevaluation led to the implementation of enhanced security measures, including the adoption of offline backup strategies, improved network segmentation practices, and more robust email filtering systems.

- **PetitPotam NTLM Relay Attack:**

The PetitPotam exploit, discovered by security researcher Gilles Lionel, emerged as a significant fileless attack technique that profoundly impacted the cybersecurity landscape from 2021 to 2023. Although not classified as malware in the traditional sense, PetitPotam exploits a vulnerability in Microsoft's Encrypting File System Remote Protocol (MS-EFSRPC) to compel remote Windows machines to authenticate against an attacker-controlled server [48]. This sophisticated technique enables potential takeover of Windows domains, particularly those with Active Directory Certificate Services (AD CS) enabled. The fileless nature of PetitPotam, operating entirely through legitimate Windows protocols, rendered it especially concerning to cybersecurity professionals. Its ability to execute without leaving typical malware artifacts on the file system posed significant detection challenges. In response to this threat, Microsoft released security updates and guidance to address the underlying vulnerability (CVE-2021-36942)[49]. The exploit's mechanism involves an unauthenticated attacker invoking a method on the LSARPC interface, coercing the domain controller to authenticate against a specified server using NTLM (NT LAN Manager) protocol. The cybersecurity community reacted by developing specialized detection rules focused on identifying unusual NTLM authentication patterns that might indicate a PetitPotam attack in progress. Organizations were advised to implement additional security measures, including the disabling of NTLM authentication where feasible and the implementation of Extended Protection for Authentication (EPA) on AD CS servers.

- **PyLoose**

In the latter half of 2023, a novel strain of fileless malware targeting Linux systems, dubbed PyLoose, emerged as a significant threat to cloud computing environments [50]. This malware represents a notable evolution in attack methodologies, being one of the first known instances of a Python-based fileless attack. PyLoose's operational mechanism exploits the memory file descriptor (memfd) utility native to Linux systems, enabling it to load its payload directly into system memory. This sophisticated approach allows the malware to circumvent conventional security measures that primarily focus on detecting file-based threats. The primary objective of PyLoose is the misappropriation of computational resources from infected systems for cryptocurrency mining operations, specifically targeting the Monero cryptocurrency. The initial vector of compromise frequently involves exploiting publicly accessible Jupyter Notebook services with inadequate system command restrictions, a vulnerability particularly prevalent in cloud-based

infrastructures. To effectively counter threats of this nature, organizations must implement a comprehensive suite of security measures. These should include:

- Implementing stringent access control protocols
- Conducting regular monitoring for anomalous system activities
- Ensuring prompt application of the latest security patches and updates

Furthermore, the deployment of advanced threat detection solutions capable of identifying suspicious memory operations and atypical network traffic patterns associated with cryptomining activities is crucial. These measures collectively form a multi-layered defense strategy essential for protecting against sophisticated fileless threats like PyLoose in modern computing environments.

- **SocGholish**

SocGholish represents a significant evolution in fileless malware technology, garnering considerable attention within the cybersecurity community[51]. This JavaScript-based malware employs a sophisticated delivery mechanism, utilizing drive-by downloads from compromised websites to infiltrate target systems. The malware's distinguishing feature lies in its ability to execute entirely within system memory, significantly complicating detection efforts by traditional security measures.
Upon successful execution, SocGholish initiates an extensive reconnaissance phase, meticulously gathering system information. This includes detailed analysis of the Active Directory environment and an inventory of installed security products. The collected data is subsequently exfiltrated to attacker-controlled command-and-control servers, providing the adversaries with a comprehensive understanding of the compromised environment. Notably, SocGholish often serves as a precursor to the deployment of more destructive payloads, particularly ransomware, underscoring its role in multi-stage attack scenarios.
Effective mitigation strategies against SocGholish necessitate a multi-faceted approach to cybersecurity. Key components of this defense strategy include:

- ✓ Implementation of rigorous endpoint protection measures, with particular emphasis on advanced endpoint detection and response (EDR) solutions. These tools focus on identifying anomalous behavioral patterns rather than relying solely on traditional file signature-based detection methods.
- ✓ Regular and comprehensive security awareness training programs for users. These initiatives are crucial in reducing the likelihood of successful drive-by download attacks, which form the primary infection vector for SocGholish.

4. Conclusion

Fileless malware has become a prevalent threat in the cybersecurity landscape, with these techniques being widely integrated into existing malware and forming the basis for new, highly sophisticated attacks. Modern endpoint compromises are increasingly caused by fileless attacks, which have demonstrated a significantly higher success rate compared to traditional file-based malware. Attackers leverage these methods to evade security measures, achieve persistence, and initiate subsequent phases of their operations. The severity of fileless malware threats cannot be overstated, as they can potentially block or restrict users' access to their systems and data. Consequently, security systems and defenders must prioritize the detection and prevention of fileless malware attacks. These attacks have shown rapid growth since 2014 (see Figure 1) and are projected to increase further in the coming years.
Fileless malware operates by executing code directly in memory and utilizing legitimate application system resources to carry out malicious actions. By leaving no traces on disk, these attacks can circumvent traditional detection solutions. The use of PowerShell and Windows Management Instrumentation (WMI) further complicates detection efforts, as these tools can bypass signature-based systems, maintain persistence, and facilitate data exfiltration.

More so, the review of previous works indicates that while significant progress has been made in malware detection, especially with the integration of machine learning techniques and behavior-based analysis, traditional methods remain insufficient against the stealth and adaptability of fileless malware. As highlighted by Idika et al. [1] and Talukder et al. [31], a multi-layered approach combining various detection methods is essential to create a robust defense system. However, current detection strategies still face challenges, particularly in identifying and analyzing fileless malware signatures, as well as in coping with its diverse and evasive characteristics.

Liu et al. and Kara emphasize the need for advanced memory-based and behavior-based detection techniques, which are more adept at combating fileless threats. The research also points to the potential of memory forensics as a critical tool, albeit one that requires further development to overcome the volatility of evidence and the complexity of distinguishing between legitimate and malicious activities.

The current term paper aims to provide a concise comparative analysis of fileless malware functionality, examining current trends and categorizing existing detection methods targeting real-world scenarios. It is evident that fileless malware will continue to evolve and become more challenging to combat, especially with the availability of open-source tools.

Our focus is primarily on fileless attack techniques targeting the Windows platform. We explore the historical development of these attacks, from early memory-resident malware to registry-based persistence attacks, and examine the prevalent trend of Living-off-the-Land (LotL) attacks and their detection on Windows systems. Over time, research on modern security frameworks has shown promise in mitigating a wide range of fileless malware attacks. To defend against such threats, behavioral runtime analysis of the system is crucial, despite its high resource consumption.

Several limitations have been identified in current fileless malware detection strategies and analysis methods, including:

- Automated analysis methods' inability to identify fileless malware signatures
- Unclear behavior patterns of fileless malware
- Diverse and unrelated characteristics of fileless malware
- Inadequate next-generation detection for fileless malware
- Fileless malware's ability to evade analysis in virtual environments and its resilience against cloaking methods

Hence, the evolution of fileless malware necessitates continuous innovation in detection and mitigation strategies. Future research would prioritize the detection of Living-off-the-Land (LotL) attacks, which have emerged as a primary form of fileless threat [36-37]. Another avenue for investigation is analyzing the effectiveness of using malware evasion tactics against itself. For instance, implementing a program on production systems that impersonates a sandbox environment could potentially deter fileless malware execution. The efficacy of such an approach in raising the bar for future evasion attempts warrants further study.

Additionally, incorporating machine learning strategies, along with data mining and AI-based immune systems, could prove beneficial in developing more robust detection and prevention mechanisms. Further research should focus on developing specific countermeasures for each type of fileless malware attack, enhancing our ability to defend against these evolving threats.

References:

[1] Idika, N., & Mathur, A. P. (2007). A survey of malware detection techniques. Purdue University.

[2] CrowdStrike, 2017. Who needs malware? How adversaries use fileless attacks to evade your security. https://www.crowdstrike .com /resources /white -papers /who - needs -malware -how -adversaries -use -fileless -attacks -to -evade -your -security/.

[3] CrowdStrike, 2022. Fileless malware explained. https://www.crowdstrike .com / cybersecurity -101 /malware /fileless -malware/.

[4] CrowdStrike, 2023. 2023 Global Threat Report.

[5] Barr-Smith, F., Ugarte-Pedrero, X., Graziano, M., et al., 2021. Survivalism: systematic analysis of windows malware living-off-the-land. In: Proceedings of the 2021 IEEE Symposium on Security and Privacy (SP '21).

[6] Siddiqui, M., Wang, M. C., & Lee, J. (2010). Data mining methods for malware detection using instruction sequences. Artificial Intelligence Review, 33(1), 1-22.

[7] Rieck, K., Trinius, P., Willems, C., & Holz, T. (2011). Drebin: Efficient and explainable detection of Android malware in your pocket. In Proceedings of the 11th Annual Network and Distributed System Security Symposium (NDSS), San Diego, CA.

[8] Virvilis, N., & Gritzalis, D. (2013). The big four - what we did wrong in advanced persistent threat detection?. In 2013 International Conference on Availability, Reliability and Security (pp. 248-254). IEEE.

[9] Kwon, D., Kim, H., Kim, J., Suh, S. C., Kim, I., & Kim, K. J. (2019). A survey of deep learning-based network anomaly detection. Cluster Computing, 1-13.

[10] Berman, D. S., Buczak, A. L., Chavis, J. S., & Corbett, C. L. (2019). A survey of deep learning methods for cyber security. Information, 10(4), 122.

[11] Aghakhani, H., Meng, D., Wang, Y. C., Kruegel, C., & Vigna, G. (2020). When malware is packin' heat; limits of machine learning classifiers based on static analysis features. In Proceedings of the Network and Distributed System Security Symposium (NDSS).

[12] Sudhakar, G., & Kumar, S. (2021). An emerging threat Fileless malware: a survey and research challenges. Journal of Network and Computer Applications, 168, 102732.

[13] Borana, P., Sihag, V., Choudhary, G., Vardhan, M., & Singh, P. (2021). An Assistive Tool For Fileless Malware Detection. In Proceedings of the International Conference on Information and Communication Technology.

[14] Liu, S., Peng, G., Zeng, H., & Fu, J. (2021). A survey on the evolution of fileless attacks and detection techniques. Computers & Security, 105, 102270.
[15] Greenberg, A. (2022). The untold story of NotPetya, the most devastating cyberattack in history. Wired. Retrieved from https://www.wired.com/story/notpetya-cyberattack-ukraine-russia-code-crashed-the-world/

[16] Han X., Pasquier, T., Bates, A., et al. (2020). UNICORN: runtime provenance-based detector for advanced persistent threats. In Proceedings of the 2020 Network and Distributed Systems Security Symposium (NDSS '20).

[17] Hossain, M. N., Wang, J., Sekar, R., et al. (2018). Dependence-preserving data compaction for scalable forensic analysis. In Proceedings of the 27th USENIX Security Symposium (USENIX Security '18).

[18] Afianian, A., Niksefat, S., Sadeghiyan, B., & Baptiste, D. (2018). Malware dynamic analysis evasion techniques: A survey. arXiv preprint arXiv:1811.01190.

[19] Living Off The Land Binaries And Scripts - (LOLBins and LOLScripts) (2019). https://github.com/LOLBAS-Project/LOLBAS

[20] Trellix, no date. What is fileless malware? https://www.trellix .com /en -us /security - awareness /ransomware /what -is -fileless -malware .html.

[21] Graeber M (2015) Abusing windows management instrumentation (WMI) to build a persistent, asynchronous, and fileless backdoor. Black Hat, Las Vegas.

[22] Mansfield-Devine, S. (2017). Fileless attacks: compromising targets without malware. Network Security, 2017(4), 7-11.

[23] Korolov, M. (2019). What is fileless malware and how to protect against it. CSO Online. Retrieved from https://www.csoonline.com/article/3385520/what-is-fileless-malware-and-how-to-protect-against-it.html

[24] Pontiroli, S. M., & Martinez, F. R. (2015). The tao of .NET and PowerShell malware analysis. Virus Bulletin Conference.

[25] Gorelik, M. & Moshailov, R. (2019). Fileless malware: Attack trend evolution and detection techniques. In 2019 Cyber Security in Networking Conference (CSNet) (pp. 1-6). IEEE.

[26] Bulazel A, Yener B (2017) A survey on automated dynamic malware analysis evasion and counter-evasion: pc, mobile, and web. In: Proceedings of the 1st reversing and offensive-oriented trends symposium, p 2 ACM.

[27] Kitchenham, B., & Charters, S. (2007). Guidelines for performing systematic literature reviews in software engineering. Technical report, Ver. 2.3 EBSE Technical Report. EBSE.

[28] Hsieh, H. F., & Shannon, S. E. (2005). Three approaches to qualitative content analysis. Qualitative Health Research, 15(9), 1277-1288.

[29] 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, 153, 102526. https://doi.org/10.1016/j.jnca.2019.102526.

[30] Tahir, R. A study on malware and malware detection techniques. Int. J. Educ. Manag. Eng. **2018**, 8, 20.

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

[32]. Souri, A.; Hosseini, R. A state-of-the-art survey of malware detection approaches using data mining techniques. Hum.-Centric Comput. Inf. Sci. **2018**, 8, 1–22.

[33] I1ker Kara. Fileless malware threats: Recent advances, analysis approach through memory forensics and research challenges. Article in Expert Systems With Application 119133, 2022. DOI: 10.1016/j.eswa.2022.119133.

[34] Graeber, M. (2015). Abusing Windows Management Instrumentation (WMI) to Build a Persistent, Asyncronous, and Fileless Backdoor. (pp. 31–48). Las Vegas, NV, USA: Black Hat.

[35] Rivera, B.S., & Inocencio, R.U. (2015). Doing more with less: a study of fileless infection attacks. Virus Bulletin, 2–56

[36] Barr-Smith, F., Ugarte-Pedrero, X., Graziano, M., et al., 2021. Survivalism: systematic analysis of windows malware living-off-the-land. In: Proceedings of the 2021 IEEE Symposium on Security and Privacy (SP '21).

[37] Symantec, 2019. Living off the land white paper. https://docs .broadcom .com /doc /living - off -the -land - turning -your -infrastructure -against -you -en.

[38] Kim, K., Alfouzan, F.A., & Kim, H. (2021). Cyber-Attack Scoring Model Based on the Offensive Cybersecurity Framework. Applied Sciences, 11(16), 7738.

[39] Lee, G., Shim, S., Cho, B., Kim, T., & Kim, K. (2021). Fileless cyberattacks: Analysis and classification. ETRI Journal, 43(2), 332–343. https://doi.org/10.4218/etrij.2020-0086 .

[40] Jakub Kroustek (12 May 2017). "Avast reports on WanaCrypt0r 2.0 ransomware that infected NHS and Telefonica". *Avast Security News*. Avast Software, Inc. Archived from the original on 5 May 2019. Retrieved 14 May 2017.
[41] A. Gat and M. Robson, "TicTacToe Dropper," Fortinet Blog, Feb. 14, 2024. Available: https://www.fortinet.com/blog/threat-research/tictactoe-dropper. [Accessed: Feb. 28, 2024]
[42] Kaspersky Lab. (2020). "Lazarus Under the Hood."

[43] CISA. (2021). Alert (AA21-048A): AppleJeus: Analysis of North Korea's Cryptocurrency Malware.

[44] Microsoft Security Intelligence. (2019). Nodersok/Divergent malware campaign.

[45] ZDNet. (2020). Barnes & Noble informs customers about data breach.

[46] FireEye. (2020). Egregor Ransomware: An Analysis of the Latest Ryuk Variant.

[47] Europol. (2021). Egregor ransomware affiliates arrested in Ukraine.

[48] Lionel, G. (2021). PetitPotam - NTLM Relay to AD CS.

[49] Microsoft. (2021). KB5005413: Mitigating NTLM Relay Attacks on Active Directory Certificate Services (AD CS).

[50] PyLoose: New Fileless Linux Malware PyLoose Targets Cloud Workloads for Cryptomining – Bitdefender- https://www.bitdefender.com/blog/hotforsecurity/new-fileless-linux-malware-pyloose-targets-cloud-workloads-for-cryptomining/.

[51] SocGholish: Living off the Land and Fileless Malware - ReliaQuest; Fileless malware campaign roundup - Zscaler- https://www.zscaler.com/blogs/security-research/fileless-malware-campaign-roundup.