
CYFIRMA
DECODING THREATS
MALWARE
DETECTION
EVASION
TECHNIQUES

INTRODUCTION

In today's ever-evolving cybersecurity landscape, the role of malware detection solutions has reached a critical juncture in safeguarding against malicious activities, and the relentless ingenuity of threat actors demands a continuous effort to stay ahead. This blog serves as a stepping stone on the path to a deeper understanding of these evasion techniques, and their critical importance in fortifying our cybersecurity measures. It's crucial to recognize that cybercriminals often employ a strategic overlap of these techniques, creating a complex web of evasion strategies that challenge conventional security measures. Signature-based evasion tactics involve subtle code alterations to elude known signatures, while behaviour-based methods manipulate malware actions to mimic legitimate behaviour. Anti-analysis techniques impede dynamic analysis through sandbox detection and delay mechanisms. Process injection techniques seamlessly integrate malicious code into legitimate processes to escape detection, and file-less malware operates within system memory, leaving minimal file traces. Overlap in the usage of evasion techniques is a natural process based on threat actor requirements and purpose: the fusion of these evasion strategies presents a formidable challenge, necessitating the adoption of advanced threat detection and prevention solutions to effectively counter these evolving threats.

MALWARE DETECTION EVASION TECHNIQUES

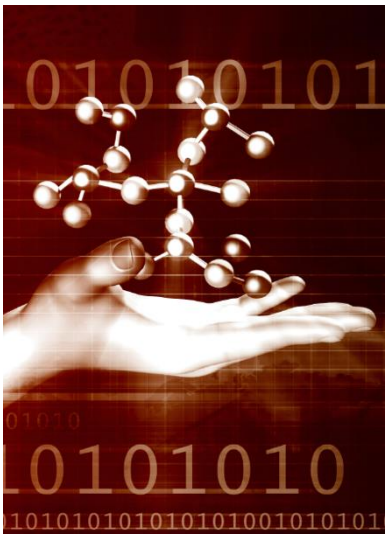
1. Signature-based Evasion Techniques

Signature-based malware evasion techniques involve altering the characteristics of malicious software to avoid detection by security solutions that rely on predefined signatures or patterns. These evasion techniques are employed by cybercriminals to bypass traditional antivirus and intrusion detection systems.

Polymorphic and Metamorphic Malware: Polymorphic malware refers to a type of malicious software that can change its code or appearance each time it infects a new system, making it difficult for traditional signature-based antivirus programs to detect and block it effectively. This evasion technique involves altering the malware's structure or encryption method, creating numerous unique variants that evade static signature-based detection, which relies on identifying specific patterns within the malware's code. On the other hand, metamorphic malware takes this concept a step further by not only changing its appearance, but also modifying its underlying code

while maintaining its functionality. This dynamic transformation complicates detection attempts, as there is no fixed signature to match against, forcing security systems to rely on more advanced behavioural analysis and heuristics to identify and combat such threats.

File Encryption: Encryption in malware involves encoding the malicious code or components to obfuscate its true purpose and evade detection by security software. This technique aims to prevent easy analysis by converting the malware's content into a scrambled format that can only be deciphered with a specific decryption key. Signature-based detection, which relies on identifying predefined patterns or signatures of known malware, becomes less effective against encrypted malware, since the signature is hidden within the encrypted content.



Packers and Crypters: Packers and crypters are techniques used in malware to evade signature-based detection. Packers are tools that compress and encrypt the malware's code, creating a new executable that requires a specific unpacking routine to be executed, before revealing the original malicious code. This obfuscation makes it difficult for traditional antivirus programs to identify the malware's signature, as it's effectively concealed within the packed executable. Crypters, on the other hand, focus on encrypting the malware's code and generating a decryption routine that can reconstitute the original malicious payload at runtime. Both techniques create dynamic and ever-changing versions of the malware, making it harder for security systems to detect and block, thereby relying more on behavioural analysis and heuristics to recognize their malicious activities.

Code Obfuscation: Code obfuscation in malware refers to the intentional manipulation of the code's structure, logic, and presentation to make it intricate and convoluted. This technique aims to hinder signature-based detection, employed by antivirus software. By transforming the code into a complex and non-standard form, the malware's recognizable patterns are obscured, making it difficult for security systems to identify its signature, based on predefined patterns of known threats.

2. Behaviour-based Evasion Techniques

Behaviour-based evasion techniques involve altering the actions and characteristics of malware to evade detection by security systems that rely on recognizing unusual or malicious behaviour. These evasion methods aim to bypass behaviour signature-based detection by focusing on dynamic analysis and anomalous activities.

Sandbox Detection: Sandbox detection in malware involves the use of behaviour-based evasion techniques to identify if the malware is running within a controlled environment, such as a virtual machine or sandbox, commonly used for analysis and detection purposes. Malware equipped with sandbox detection mechanisms can detect the presence of certain attributes or behaviours associated with such

environments, like specific file paths, registry entries, or network configurations. Upon detection, the malware may alter its behaviour, delay malicious actions, or remain dormant to evade analysis. This dynamic response aims to mislead security analysts and automated systems, delaying the discovery of its true malicious intent until it's executed in a legitimate user environment, thereby allowing the malware to avoid detection and maximize its potential impact.

Environment Checking: Environment checking in malware involves employing behaviour-based evasion techniques to assess the context in which the malware is running, allowing it to adapt its behaviour accordingly. By monitoring factors like system configurations, software installations, network settings, or even the presence of security tools, the malware can determine if it's operating within a virtualized environment or under the scrutiny of security analysts. Upon detecting signs of analysis or sandboxing, the malware can alter its actions, delay its malicious payload, or even halt its operation temporarily, thereby evading detection and analysis attempts and making it more challenging for security systems to accurately identify its true nature.

Process Hollowing: Process hollowing in malware involves a behaviour-based evasion technique, where a legitimate process is manipulated by the malicious code to replace its legitimate code with the malware's payload. This technique is used to run the malware within a seemingly normal process, making it harder to detect. The malware starts by creating a new instance of a legitimate process and then replaces its code with its own, effectively hollowing out the process. This behaviour-based approach allows the malware to execute within a trusted context, often bypassing security measures that focus on static signatures or known malicious patterns, thus evading detection, while leveraging the familiar behaviour of legitimate processes.

3. Anti-analysis Techniques

Anti-analysis techniques are strategies employed by malware authors to impede the efforts of security researchers, analysts, and automated systems attempting to analyse and understand malicious software. These techniques make it more difficult to uncover the true intent and behaviour of malware, ultimately hindering effective defense and response.

Code Obfuscation: Malware code is intentionally convoluted and obscured through techniques like encryption, packing, and code restructuring. This makes it challenging to decipher the actual functionality and purpose of the malware.

Anti-debugging Techniques: Malware employs tactics to detect whether it's being executed within a debugger or analysis environment. If detected, the malware might alter its behaviour or refuse to execute, thwarting attempts at analysis.

Anti-VM Detection: Employing techniques to identify if the malware is running within a virtual machine (VM) and altering behaviour accordingly.

4. Process Injection Techniques

Process injection techniques are strategies used by malware to insert their malicious code into legitimate processes running on a compromised system. This enables malware to evade detection, leverage the privileges of the targeted process, and execute its malicious activities under the guise of a trusted application.

DLL Injection: DLL injection in malware is a process injection technique where a malicious code injects its own Dynamic Link Library (DLL) into a legitimate process running in memory, enabling it to manipulate the process's behaviour and potentially execute malicious actions without raising suspicion. This technique involves exploiting the memory space of a target process to load the malicious DLL, allowing the malware to gain control over the process's functions, data, and resources. By leveraging the context of a trusted process, the malware can evade traditional signature-based detection and blend in with legitimate activities, making it challenging for security systems to discern the malicious behaviour from the host process's normal operations.

Code Injection: Code injection in malware refers to a process injection technique, where malicious code is inserted into the memory space of a legitimate process, altering its behaviour to carry out malicious actions. This method involves exploiting vulnerabilities in a target process's memory to inject the malicious code, effectively hijacking the process's execution flow. By leveraging the trusted context of the legitimate process, the injected code can evade detection mechanisms that rely on signature-based identification, making it difficult for security solutions to distinguish between the legitimate process and the injected malicious behaviour.

Thread Injection: Thread injection is a process injection technique employed by malware, where the malicious code creates a new thread within a legitimate process and injects its payload into that thread's execution flow. By doing so, the malware leverages the resources and privileges of the legitimate process to execute its malicious actions. This method enables the malware to avoid detection mechanisms that focus on process-level activities, as it operates within the context of an existing trusted process. Since the injected thread mimics legitimate behaviour, it becomes challenging for security systems to discern the malicious thread from the genuine operations of the host process, thus evading signature-based detection approaches.

Memory Allocation/Write Techniques: Memory allocation/write techniques are process injection methods used by malware to insert their malicious code into the memory space of a legitimate process. These techniques involve the malware dynamically allocating memory within the target process's address space and then writing its payload into that allocated memory. By manipulating the memory of a trusted process, the malware can execute its code without triggering traditional signature-based detection methods. This approach allows the malware to evade static analysis, as its code is not stored as a standalone file, making it challenging for security systems to identify and isolate the injected malicious behaviour from legitimate process activities.

5. Fileless Malware Techniques

Fileless malware refers to malicious code that operates entirely within a computer's memory, without leaving a trace on the filesystem. This evasion technique allows malware to avoid detection by traditional security solutions that focus on detecting and analysing files.

Memory-based Execution: Memory-based execution in malware refers to a fileless technique, where malicious code is directly loaded and run in a system's memory, without leaving traces of its presence on the filesystem. This approach leverages vulnerabilities or legitimate system tools to inject and execute the malicious payload in memory, bypassing traditional signature-based antivirus detection that primarily scans files. Since the malicious code doesn't rely on a persistent file, it becomes harder for security solutions to detect and mitigate, as there is no traditional file-based signature to identify. This technique emphasizes exploiting system weaknesses and operating in memory, enabling the malware to operate stealthily and avoid leaving traditional forensic traces.

Living-off-the-land Techniques (LOLbins): Living-off-the-land techniques (LOLbins) in malware involve utilizing pre-existing, trusted system tools and processes to execute malicious activities, often without the need to drop new files. This fileless approach leverages legitimate tools like PowerShell, WMI (Windows Management Instrumentation), or other command-line utilities to carry out malicious actions directly in memory or on the system. By using tools already present on the system, attackers can avoid the need to introduce new files, making it challenging for traditional file-based detection methods to identify their activities. These techniques capitalize on the inherent trustworthiness of built-in tools, making it difficult for security systems to distinguish between legitimate and malicious usage.

Registry-based Attacks: Registry-based attacks in malware involve exploiting the Windows Registry—a hierarchical database used to store configuration settings, user preferences, and system information—for malicious purposes. In the context of fileless malware techniques, attackers manipulate the registry to execute their malicious code directly from memory or through trusted system utilities, without the need to drop files onto the filesystem. By leveraging legitimate registry keys and values, attackers can achieve persistence, privilege escalation, and execution of malicious scripts without raising suspicion, as these activities often bypass traditional file-based security measures. This fileless approach capitalizes on the trusted nature of registry interactions, making it challenging for security solutions to detect and prevent these covert attacks.

Document Macros: Document macros in malware involve using scripts embedded within documents, such as Microsoft Office files, to execute malicious actions when the document is opened. This technique falls under the umbrella of fileless malware as it doesn't rely on traditional executable files; instead, it leverages the scripting capabilities of popular document formats. When a user opens a document containing macros, the script is executed, and it can download and execute malicious payloads directly in memory, evading traditional file-based detection methods. By taking advantage of the user's trust in commonly used document formats, attackers can exploit this vector to deliver and execute malware without relying on standalone executable files.

CONCLUSION

The ever-evolving landscape of cybersecurity presents an ongoing battle between threat actors refining their evasion techniques, and security professionals striving to keep up with these changes. Meanwhile, signature-based methods fall short in the face of zero-day exploits, and the resource-intensive nature of dynamic analysis poses its own challenges. The prevalence of false positives and negatives further complicates detection, while the complexity of advanced evasion techniques demands specialized expertise. Nonetheless, addressing these challenges is imperative for organizations to safeguard their systems and data from evolving threats.

To effectively combat these issues, a multi-faceted security approach is crucial. This involves integrating signature-based detection with advanced techniques such as behavioural analysis, heuristics, machine learning, and anomaly detection. Additionally, staying abreast of the latest threat intelligence and adopting proactive strategies are paramount in staying ahead of the game. By embracing these measures, organizations can better fortify their defenses against the relentless onslaught of signature-based malware evasion techniques.

CYFIRMA is an external threat landscape management platform company. We combine cyber intelligence with attack surface discovery and digital risk protection to deliver early warning, personalized, contextual, outside-in, and multi-layered insights. Our cloud-based AI and ML-powered analytics platform provides the hacker's view with deep insights into the external cyber landscape, helping clients prepare for impending attacks. CYFIRMA is headquartered in Singapore with offices across APAC, US and EMEA. The company is funded by Goldman Sachs, Zodius Capital, Z3 Partners, OurCrowd and L&T Innovations Fund.