Cybersecurity

SURVEY

Open Access

An emerging threat Fileless malware: a survey and research challenges

Sudhakar[1,2*] and Sushil Kumar[3]

Abstract

With the evolution of cybersecurity countermeasures, the threat landscape has also evolved, especially in malware from traditional file-based malware to sophisticated and multifarious fileless malware. Fileless malware does not use traditional executables to carry-out its activities. So, it does not use the file system, thereby evading signature-based detection system. The fileless malware attack is catastrophic for any enterprise because of its persistence, and power to evade any anti-virus solutions. The malware leverages the power of operating systems, trusted tools to accomplish its malicious intent. To analyze such malware, security professionals use forensic tools to trace the attacker, whereas the attacker might use anti-forensics tools to erase their traces. This survey makes a comprehensive analysis of fileless malware and their detection techniques that are available in the literature. We present a process model to handle fileless malware attacks in the incident response process. In the end, the specific research gaps present in the proposed process model are identified, and associated challenges are highlighted.

Keywords: Fileless malware, Botnet, Incident response, Memory forensics, Incident investigation, Memory resident malware, Rootkit

Introduction

Throughout the history of the malicious programs, there is one thing which had remained unchanged, the development of the malware program. Someone had to develop the code in such a manner so that no existing anti-virus (AV) software can detect its presence in the system. In 2002, the development of the malware industry had changed the entire threat landscape. This malware has the capability of residing in the system's main memory undetected making least changes in the file system. This strategy has become the non-malware or fileless malware (Patten, 2017; Kumar et al., 2019a).

When a system is detected as the compromised by some malicious program or malware, the very first thing a forensic expert will work to look for some malicious programs or software that should not be there. However, in this case, there is none because the fileless malware does not reside in the file system, it is a running program in the memory (Mansfield-Devine, 2017; Tian et al., 2019a). The attacker has been using malware for its capabilities to control the compromised systems locally or remotely. Although, the operating systems itself providing several capabilities to the attacker.

The attackers are mostly involved in exploring vulnerabilities in the legitimate software that are already installed in the machine such as flash player, web-browser, PDF viewer and Microsoft office to exploit and load a script directly into the main memory without even touching the local file systems (Pontiroli & Martinez, 2015; Rani et al., 2019). In Windows Operating Systems, two most powerful tools and. NET framework are already installed which attacker can use to exploit the vulnerability, one is WMI (Windows Management Instrumentation) (Graeber, 2015) and second is PowerShell. WMI came into the limelight of the cybersecurity community when it was discovered that it is used maliciously as a component in the suite of exploits by Stuxnet (Falliere et al., 2011; Farwell & Rohozinski, 2011). Since then, WMI has been gaining popularity amongst the attackers, because it can perform system reconnaissance, AV/VM (Virtual Machine) detection, code execution, lateral movement, persistence, and data theft. Similarly, in the case of PowerShell, it is a highly flexible system shell and scripting platform for the

* Correspondence: sudhak82_scs@jnu.ac.in
[1]School of Computer & Systems Sciences, Jawaharlal Nehru University, 110067, New Delhi, India
[2]Indian Computer Emergency Response Team, Ministry of Electronics & Information Technology, 110003, New Delhi, India
Full list of author information is available at the end of the article

attacker to provide all the features in the different stages of an intrusion. Since, it can also be used to bypass antivirus detection, maintain persistence or infiltrate data. For example: In 2016, a hacker group infiltrates into the DNC (Democratic National Committee) with fileless malware. In this incident, the PowerShell and WMI were used as the attack vectors (Report, 2016).

In recent time, malware developers have adopted the high-level language in the development of the malicious codes that have changed the malware industry. After the release of Microsoft .Net framework, it became the center of attraction for all the windows software developers and unintentionally revolutionized the malware industry (Tian et al., 2019b; Tian et al., 2019c). It gives the malware writers a new and powerful arsenal equipped with all the features to make a malware undetected and stay ahead of the antivirus software. With the use of this framework, malware creator may easily interact with the operating system and exploit vulnerabilities with the entire catalog of products with the help of the framework (Patten, 2017; Pontiroli & Martinez, 2015; Tian et al., 2019d; Bhasin et al., 2018). The attacker uses a tool like PowerShell to coordinate attacks with the help of existing toolkits such as meterpreter (About the Metasploit Meterpreter, 2019), SET (Social Engineering Toolkit) (Pavković & Perkov, n.d.), or the Metasploit Framework including an extensive list of modules that are already built-in and ready to use for the purpose of plotting additional attacks (Tian et al., 2018).

The fileless malware attacks in the organizations or targeted individuals are trending to compromise a targeted system avoids downloading malicious executable files usually to disk; instead, it uses the capability of web-exploits, macros, scripts, or trusted admin tools (Tan et al., 2018; Mansfield-Devine, 2018). Fileless malware can plot any attacks to the systems undetected like reconnaissance, execution, persistence, or data theft. There are not any limitations on what type of attacks can be possible with fileless malware. This survey includes infection mechanisms, legitimate system tools used in the process, analysis of major fileless malware, research challenges while handling such incidents in the process of incident and response. This type of study was not done in the literature, which includes the different perspectives of fileless malware. The main contributions of the paper are mentioned below.

- The background of the malware with evasion and propagation techniques used to identify targets and stay undetected in the victim machines.
- We analyze the behavior of all the fileless malware and discuss their persistent mechanisms in detail.
- We analyze many solutions given by researchers to detect such malware by analyzing the malicious patterns in the process, registry, minor changes in file systems, and event logs.
- We proposed a novel investigative model of incident handling and response, especially in fileless malware. The model includes all the phases with memory forensic, analysis and investigation of such incidents.

The rest of the paper is organized as shown in Table 1.

Background of Fileless malware

Unlike traditional file-based malware attacks, instead of using real malicious executables, it leverages trusted, legitimate processes i.e. LOLBins (Living off the Land Binaries) (Living Off The Land Binaries And Scripts - (LOLBins and LOLScripts), 2019) and built-in tools of operating systems to attack and hide. The detailed comparisons between traditional file-based malware and fileless malware are mentioned in the Table 2 (Afianian et al., 2018). In this section, the formal definition of the fileless malware and execution techniques along with the system tools, is discussed. The section also elaborates on the infection technique used by such malware with attack vectors, as shown in Fig. 1.

Definition

Fileless malware attacks do not download malicious files or write any content to the disk in order to compromise the systems. The attacker exploits merely the vulnerable application to inject malicious code directly into the main memory. The attacker can also leverage the trusted and widely used applications, i.e., Microsoft office or administration tools native to Windows OS like PowerShell and WMI to run scripts and load malicious code directly into volatile memory (Stop Fileless Attacksat Pre-execution, 2017; Zhang, 2018). The procedures and attack vectors are mentioned in Fig. 1.

Execution of fileless malware

The malware authors have leveraged the advantage of two powerful legitimate windows applications Windows Management Instrumentation and PowerShell to execute their malicious binaries to make the attack undetected by AV solutions (Graeber, 2015).

The life cycle of the fileless malware works in three phases. First, attack vector, which has methods through which the attacker targets their victims. Second, the execution mechanism in this the initial malicious code could try to create a registry entry for its persistence or WMI object with VBScript/JScript to invoke an instance of PowerShell. Third, PowerShell can further able to execute the malicious program into the legitimate process memory directly without dropping any files to the file system making its target compromised by fileless malware; the infection life cycle is shown in Fig. 1.

Table 1 Outline of the paper

Section	Description
Introduction	The change in cybersecurity threat-landscape and associated threat actors over time, especially in malware perspective from traditional file-based malware to fileless malware. This section also includes the motivation and contribution of this survey.
Background of Fileless Malware	This section explains the definition of fileless malware and exploits mechanism with tools through which the initial infection of fileless malware.
Analysis of Fileless Malware Based on Their Persistent Techniques	This section have the analysis of the behavior of fileless malwares and classify them into their persistent mechanism to hide and execute into the targeted systems.
Detection Techniques for Fileless Malware	This section presents the prevention and detection systems are used to detect the malicious programs running in memory, leaving no physical file on the compromised system. The section presents the detection mechanism to detect these type of malicious programs.
Proposed Process Model for Incident Response	A novel investigative framework is proposed to break-down the analysis into simple steps for effective detection and analysis of root-cause of such attacks.
Research Challenges	The problems which are facing by the state-of-art investigation procedure in each step are explained in detail.
Conclusion	In this section, we concluded the paper.

Windows management instrumentation

With the emergence of new technologies, many changes have occurred in the Windows Operating System over the period, but WMI remains powerful since Windows NT 4.0, and Windows 95. It has an excellent reputation in the security community to launch an attack across many phases of the attack life-cycle like reconnaissance, AV/VM detection, code execution, lateral movement, covert data storage, to persistence. Using these capabilities of WMI, an attacker can build a pure backdoor without even dropping the single file to the file system. In addition to this, it can be used to execute malicious JavaScript/VBScript directly in the memory to possibly evade the AV solutions (Graeber, 2015; O'Murchu & Gutierrez, 2015; Ruff, 2008).

Table 2 Comparison between file-based malware and fileless malware

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.

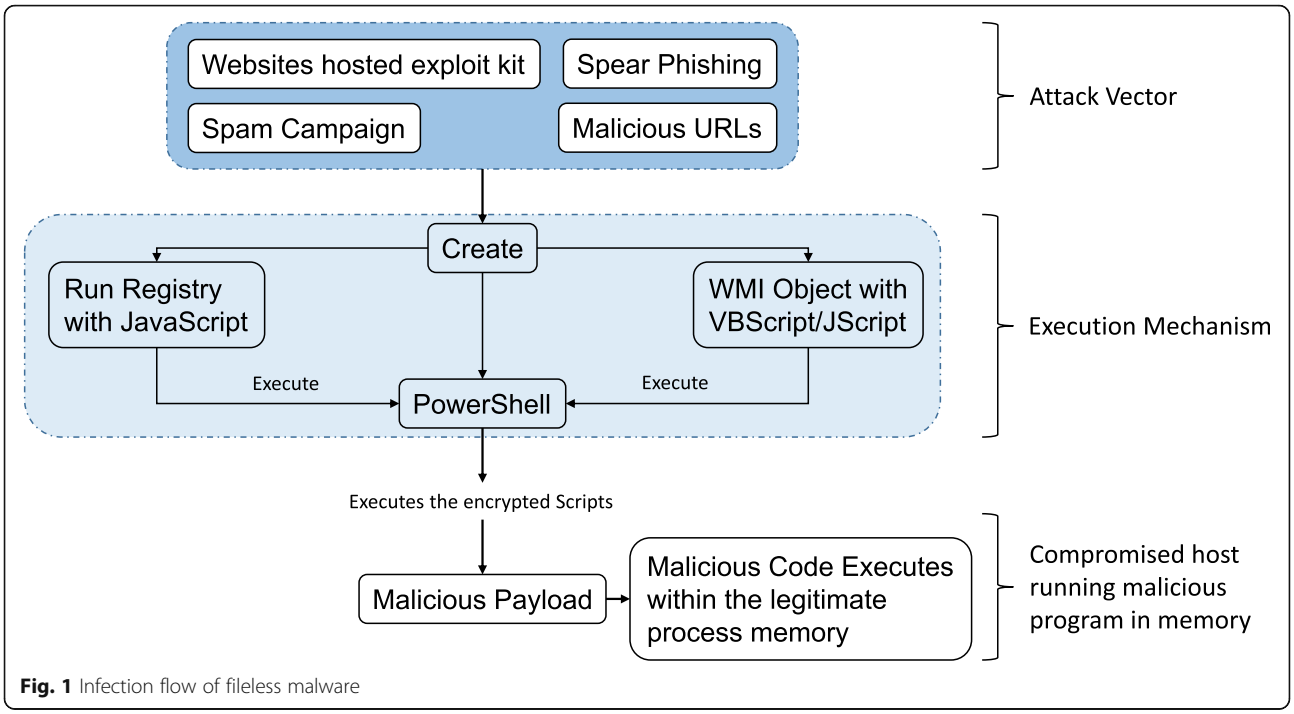

Fig. 1 Infection flow of fileless malware

PowerShell

PowerShell has a rich number of features to facilitate the attacker to even bypasses the detection capability of AV solutions to maintain persistence or spy on the system. Windows operating system has already whitelisted many modules for PowerShell. For example, evasive techniques (Bulazel & Yener, 2017) can be used to dynamically load PowerShell scripts into the memory without even writing anything on the file system (Pontiroli & Martinez, 2015; Case & Richard III, 2017).

Analysis of Fileless malware based on their persistent techniques

There are major three categories of fileless malware, which are described below by their persistence techniques and detailed classification of their attack vectors in Table 3. The fileless malware can hide their location to make difficulties in the process of detection by traditional AV solutions and also for the security analyst (Demystifying Fileless Threats, 2019; Rivera & Inocencio, 2015).

Memory-resident malware

The malware that wholly resides in the main memory without touching the file systems. It uses only legitimate process or authentic windows files to execute and stays there until it triggered. The malware having such capabilities are described in this section:

Code Red (Zou et al., 2002; Danyliw & Householder, 2001; Rhodes, 2001): Code red infect Microsoft's Internet Information Server (IIS) of version 4.0 and 5.0 having known buffer overflow vulnerability. The system is infected when server GET/default.ida request on TCP port 80 allowing the worm to run code on the server.

SQL Slammer (O'Murchu & Gutierrez, 2015; MS SQL Slammer/Sapphire Worm, 2003): SQL Slammer is a computer worm (SQL Slammer, 2019). It has the power to choke the bandwidth of the network resulting in a denial-of-service condition. The worm has used the method to

Table 3 Classification of Attack Vectors for Fileless Malware detection

Functionality/ Attack Vector	Memory Resident Malware				Windows Registry Malware		Rootkit Malware
	SQL Slammer	Code Red	Poweliks	Lurk	PowerWare	Kovter	Phase Bot
Infection method	In memory	In memory	In memory	In memory	In memory	Registry	In memory
Configuration data	In memory	In memory	In memory	In memory	No	Registry	Registry
Injection data	In memory	In memory	Registry	In memory	No	Registry	Registry
Persistence method	No	No	Registry	No	No	Registry	Registry
Malware Category	Worm	Worm	Click fraud Bot	Trojan	Ransomware	Ad-Click fraud	Bot

propagate and infect by scanning the buffer-overflow vulnerability over the internet.

Lurk Trojan (Golovanov, 2012; Shulmin & Prokhorenko, 2016)**:** Lurk is a banking Trojan, infection could be possible either using command "regsrv32" and "netsh add helper dll" or via the ShellIconOverlayIdentifiers branch of the system registry. The Trojan uses its specific features to gain access to the users' sensitive data, and with the help of it, an attacker can compromise their online banking services.

Poweliks (O'Murchu & Gutierrez, 2015; Team, 2017; Zeltser, 2017): Poweliks is fileless malware that is further being developed from file-based malware, known as Wowliks. The malware installs itself into the registry as well as use it to persist in the system, thus escape from AV solutions as it did not leave any files written on the disk. Also, the malware installs PowerShell in the background without alarming the defensive system if the system does not have it already. The system is penetrated by exploiting the Microsoft Office vulnerabilities and used the PowerShell along with JavaScript with shellcode to directly execute into legitimate memory. The attack vectors are compared between Wowliks and Poweliks in Table 4.

- ***Persistence mechanism*** **-** Poweliks uses system registries to achieve persistence, to stay undetected. It added two registries to the run key. First, in the form of JavaScript program encoded data written under (Default) value and other is the autorun entry that reads and decodes the encoded JavaScript data.

Windows registry malware

Registry is the database for storing low-level settings of the Windows operating system and some critical apps. In there, the malware authors managed to store complete malicious code into the registry in an encrypted manner, to make it undetected. To obtain the persistence, it can exploit some operating systems thumbnail cache using registry. However, the file is set to self-destruct once it carried out its malicious task (Wueest & Anand, 2017).

Kovter (Team, 2017; Fileless Malware - A Behavioural Analysis Of Kovter Persistence, 2016): Kovter can conceal itself in the registry and maintain persistence through the use of registry run key. The infection mechanism of Kovter has the feature to leave very few file traces of artifacts. It uses PowerShell for the execution of commands to achieve its malicious venture. Once the execution is completed PowerShell losses all the environment variables and it does not log the list of executed commands due to this, it leaves little chance to recover the script executed and the sample or information about the final payload (Zaharia, 2016).

- **Persistence mechanism -** JavaScript code is added into the registry and is executed by a legitimate Windows file, mshta.exe, via WMI instead of mshtml.dll:

HKLM\path{\Software\Microsoft\Windows\Current Version\Run}.

Data: mshta javascript: {javascript code}.

Kovter decrypts the first stage JavaScript code, which resides in the registry leading to the second stage JavaScript containing a PowerShell script, which decodes the encoded shellcode and injects it to the legitimate windows process (regsvr32.exe) to execute, using a technique called Process Hollowing (Process Hollowing, 2019). The flow of injecting shellcode is mentioned in Fig. 2.

PowerWare (Valdez & Sconzo, 2016): It is a fileless ransomware, which is mostly delivered via a macro-enabled Microsoft Word document. The malware uses the core utilities of windows operating the system such as PowerShell. By leveraging the capabilities of it, the ransomware entirely avoids writing any file on the disk and perform its malicious activities.

Table 4 The Evolution of Poweliks

Functionality/ Attack Vector	Wowliks	Poweliks
Infection method	File-based	In memory
Configuration data	On-disk	In memory
Injection data	DLL file on disk	Registry
Persistence method	DLL file on disk	Registry

Rootkits fileless malware

An attacker can install this kind of malware after getting the administrator level privilege to hide the malicious code into the kernel of the Windows operating system. While this is not a 100% fileless infection either, it fits here.

Phase Bot (Zeltser, 2017; Phase Bot - A Fileless Rootkit (Part 1), 2014; Phase Bot - A Fileless Rootkit (Part 2), 2014): It is a type of bot, which can grab its' victim information by applying form-grabbing approach (Sood et al., 2011) and stealing FTP data connection (Allman & Ostermann, 1999) with the ability to run without a file. Phase hides its relocatable code encrypted in the registry and uses PowerShell to read and execute this independent position code into memory. Both Phase Bot and Poweliks uses similar persistence mechanism.

Detection techniques for Fileless malware

In the case of fileless malware, PowerShell and WMI could be used to reconnaissance, establishing persistence, lateral movement, remote command execution, and file transfer, make it difficult to track evidence left behind during a compromise (Pontiroli & Martinez, 2015). In order to detect such malware infection, various techniques (Section 4.1–4.3) have been proposed by the

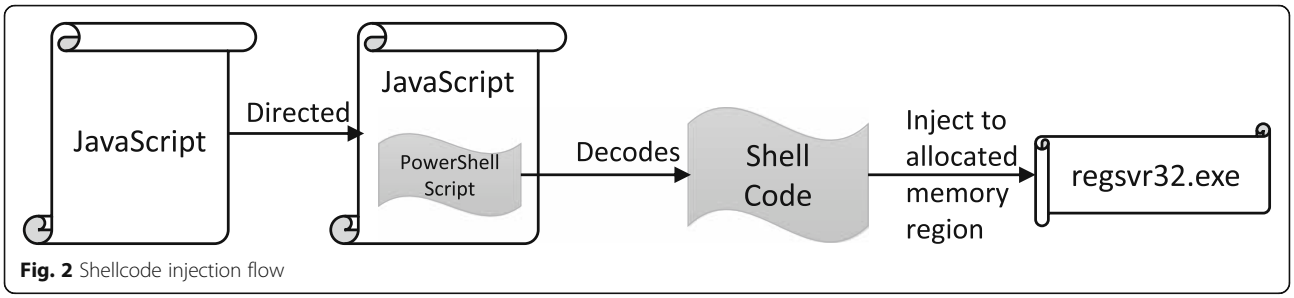

Fig. 2 Shellcode injection flow

researchers in their work. First two techniques (Section 4.1 and Section 4.2) are manual inspection techniques, need a security professional to look into the evidence prescribed by the researcher to successfully identify such attacks, whereas the third technique (Section 4.3) is only a concept yet to implement.

Detection by monitoring the behaviour of the system

In order to detect fileless malware, the system needs to consider two things. First, the processes which have elevated privileges after becoming live into the memory and Second, monitor the security events for the program execution by command-line console or PowerShell (Pontiroli & Martinez, 2015).

1. The attacker first aim is to gain the root access to its victim machine to take the full privilege of the PowerShell. To identify fileless infections, the system needs to monitor all the essential features that are accomplished by PowerShell capabilities, such as:

a. Remote command execution by the PowerShell.
b. Change of standard user privilege to administrative privilege to access WMI and .NET Framework base class library.
c. Programs, which are executing in the main memory, may be malicious.

2. It is essential to identify the principal sources of information such as network traffic, network connections, and suspicious modifications to particular Windows registry keys. In addition, the Windows event log, being on guard for clear indicators that may suggest the malicious activity has taken place. Some of the indispensable events need to be adequately monitored, such as:

a. **Event ID 4688:** The system needs to monitor all the newly created processes whose parent process is PowerShell.
b. **Event ID 7040:** If the service has been changing to auto-start from disabled/demand start of the Windows Remote Management (WS-Management).
c. **Event ID 10148:** This event is responsible for listening to the specific IP and Port for WS-Management related requests.

Detection by rule-based

Majority of malicious programs spread across the internet via targeted by the attacker or by the botnet to find the vulnerable victim is packed with Microsoft Office applications such as winword.exe, excel.exe, and powerpnt.exe. Furthermore, the detection of such programs, which trigger the cmd.exe or powershell.exe, could be malicious. Hence, the detection mechanism may work by the rule that can distinguish between benign process and malicious process (Valdez & Sconzo, 2016).

These similar rules can be implemented in the browsers to block such malicious apps from executing PowerShell and Command prompt. This should also help with other types of malware leveraging other Microsoft Office applications as mentioned in Fig. 3.

- **CASE-1:** ProcessName: cmd.exe AND (parentName: winword.exe OR parentName: excel.exe OR parentName: powerpnt.exe OR parentName: outlook.exe) AND chilprocessName: powershell.exe
- **CASE-2:** ProcessName: cmd.exe AND (parentName: winword.exe OR parentName: excel.exe OR parentName: powerpnt.exe OR parentName: outlook.exe)
- **CASE-3:** ProcessName: powershell.exe AND parentName: winword.exe
- **CASE-4:** ProcessName: powershell.exe AND filemodCount: [1000 to *]

Detection by learning behavior of attack

A framework can be developed in the paradigm of client-server, wherein all the endpoints have a client deployed, and a server in the cloud. The framework is

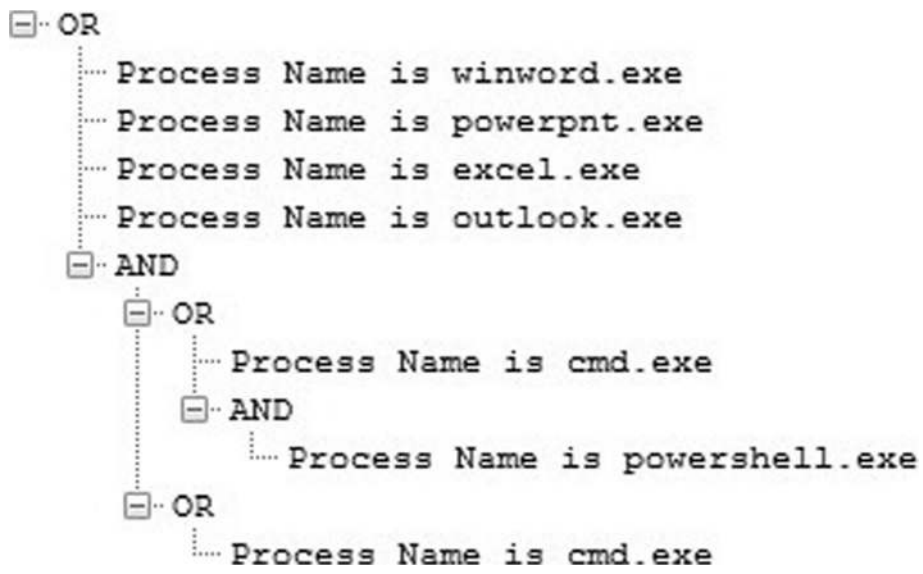

Fig. 3 Malicious process flow

divided into three stages, like capturing events, tagging events, and learning from the events. In this system, the client can capture all the generated events by the host machine as mentioned in the Fig. 4, to monitor the full stream of activity. In addition, the client also assigns a tag to each event appropriately to uncover the attacker's progress. At last, in the server, many analysis engines working on the tagged-events supplied by the client to detect the malicious activity in the host machine. The tagged-events will be the raw data for learning algorithms and analyzing the behavior of the patterns to prevent or detect malicious activity through the co-relation amongst the event streams (Series, 2019).

Proposed process model for incident response

The proposed model is especially for malware related incidents both in real-time attacks and after attack scenarios. The process flow diagram and block diagram are thoroughly explained in Fig. 5 and Fig. 6. The first five-phase (including incident response) is for the identification and investigation of the incident. Each organization should have an incident response team to handle cybersecurity incidents. The team can follow this process model and process flow to investigation the root cause of such attacks.

The analysis of evidence and artifacts are starts from the examination stage, where forensic examiner concludes whether the malware sample needs to investigate further. In this scenario, the attack pattern can be identified with the help of many techniques. Which are useful in detecting anomalies between the benign and malicious behavior of the system.

Preparation

The primary motive of this phase is continuous capability enhancement of the handling incident based on risk assessment and experiences of the incident handling team. In this context, regular training of the individual incident handler needs to be arranged to keep them ready for new security threats and related security tools for more sophisticated malicious programs like fileless malware, botnet, APT (Advanced Persistent Threat) and DDoS.

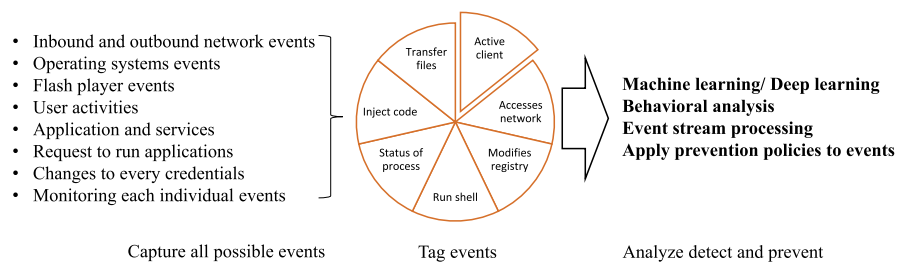

Fig. 4 A framework to detect and prevent fileless malware attacks

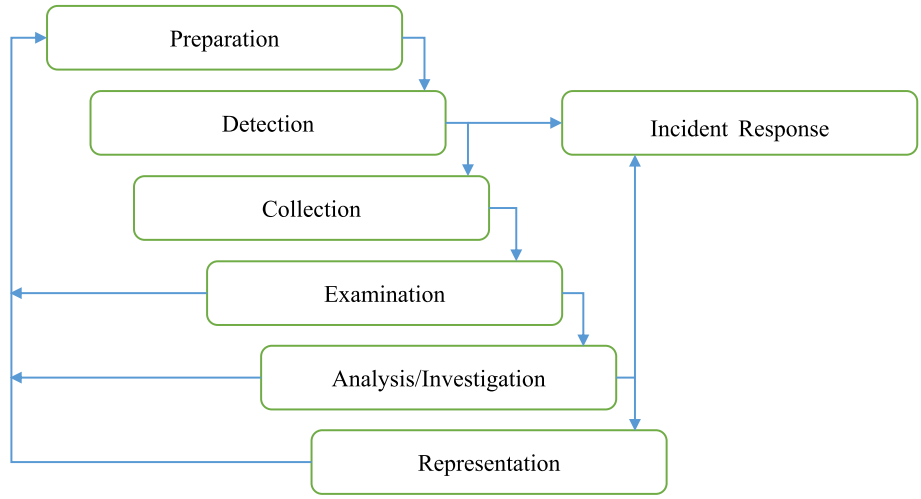

Fig. 5 A process model for memory forensics

An incident handler needs to gather all available information about the incident. To facilitate the incident handler, forensic software and most advanced malware analysis tools are required. In case of occurrence of such incidents, an organization must have a policy in place containing reporting, information sharing, especially with outside organizations such as law enforcement (Khurana et al., 2009).

Detection

In the organization, security tools (anti-malware agents, intrusion detection/prevention systems, network sandboxes, and firewalls, etc.) must installed in their infrastructure to prevent and detect it from the security breaches and policy violations. If any alert are unauthorized events or anomalies triggers, then it requires a security analyst to examine logs and analyze the root cause of the unauthorized event (Shackleford, 2016). The malicious events could lead to the existence of malware, which could be determined by the various parameters, and quick validation can be done to confirm the attack. The validation is required to determine whether to investigate the suspected attack or ignore it. This phase can generate two branches – *incident response* and *collection* (Pilli et al., 2010).

Incident response

The reactive and proactive response will be generated for intrusion detection according to the organization's policy and guidelines. The damage already caused due to a cyberattack could be mitigated, and future action plans can be determined for such attacks. In addition, a similar response with more information obtained from the investigation can be initiated (Khurana et al., 2009; Pilli et al., 2010).

Collection

The evidence of the incident can be acquired from the sensors, which are in place in the network or from the compromised machines. The sensors can be honeypot/honeynet (Watson & Riden, 2008) to collect malicious samples or to capture the network behavior. All the collected evidence are preserved for further analysis and investigation purpose, as mentioned in Fig. 6. All the analysis and investigation must be done on a copy of the original data, and the original evidence file is untouched to facilitate legal requirements (Khurana et al., 2009).

Examination

The traces obtained from various security sensors are integrated and fused to form one extensive data set on which analysis can be performed. The data set may contain redundant information, which needs to be removed and rectified efficiently. In this process, the crucial data should be intact (Khurana et al., 2009; Ren, 2004). The indicators of cybercrime can be searched from the extensive data set of evidence like malicious network traffic (Pilli et al., 2010; Tobergte & Curtis, 2013). The finding may be shared with the development team of security tools for improvement purposes (Case & Richard III, 2017; Cohen, 2017; Burdach, 2006).

Analysis & Investigation

While analysis of such incidents, an expert should consider the behavior of the systems such as registry entries, network communication, operating system fingerprinting, memory analysis, and process analysis to match for any malicious activities. Thorough investigations must be performed from a victim network or system through any intermediate nodes and communication pathway to pinpoint from where the attack originated.

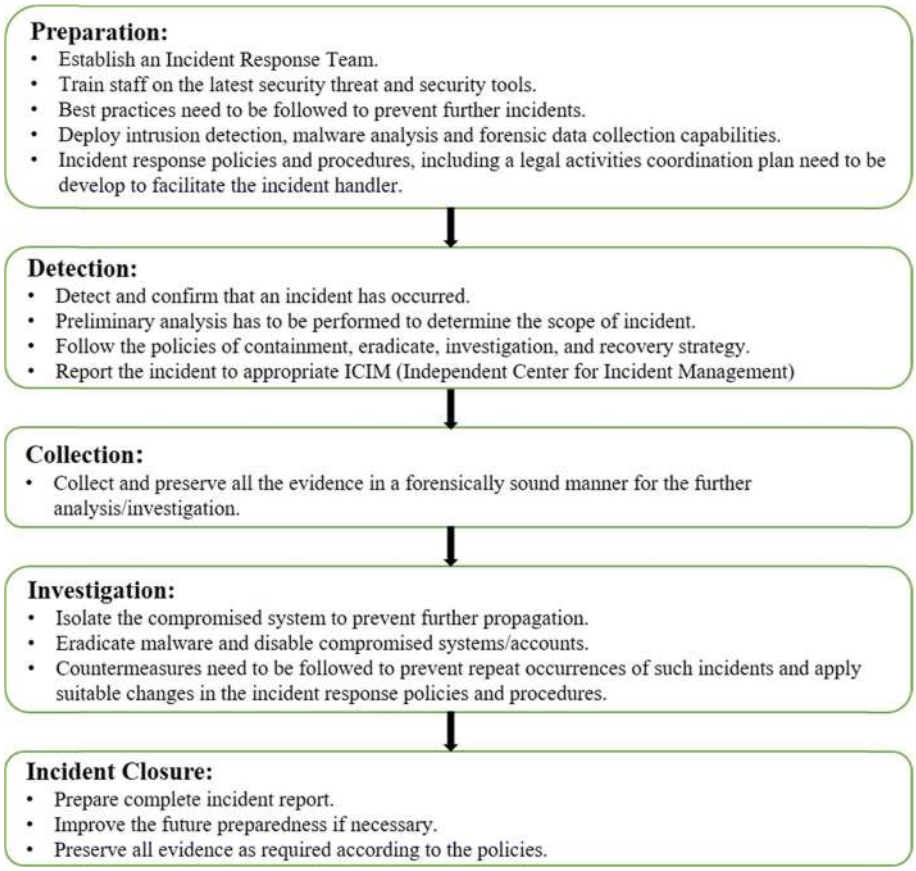

Fig. 6 The process flow of cybersecurity incident handling and response

The attacker can remove their footprints from the crime scene, such as securely delete the log files, payloads, registry entries, and cookies. IP spoofing and stepping stone attack strategies can also be used by the attacker to hide their IP address (Vacca, 2013). The investigation phase provides data for incident response and prosecution of the attacker (Khurana et al., 2009; Pilli et al., 2010).

Representation

All the findings from the different phases, such as examination and analysis/investigations, are collected and presented in a proper, understandable language for legal purposes with evidence. The detailed documentation can be presented in the form of a report with visualization so that they could be easily grasped (Khurana et al., 2009; Pilli et al., 2010).

Research challenges

Fileless malware already presents a significant problem, and it is gaining further popularity among attackers because it is undetectable by traditional file-based prevention and detection systems. Since, there are no files written on the disk, when the malware is persisting exclusively through process memory and registry files. The malicious process is not accessible without doing in-memory analysis because the source code of the process is not available. The fileless attacks can evade these AV tools without triggering alarms as deduce from the eq. 1.

$$No\ Code\ OR\ Files \Rightarrow No\ Detection \qquad (1(1))$$

In the process of prevention, detection, and collection of the malicious files or attack vectors in the infrastructure poses the various research challenges in incident handling and response, which are discussed in the following subsections. In Table 5, the specific and common challenges are compared for both file-based and fileless malware.

Detection and collection

The first step to detect the fileless malware attacks is to identify the malicious pattern of the system and the malicious scripts, which are running in the memory. The

Table 5 Comparison of challenges between file-based and fileless malware

Challenges	File-based malware	Fileless malware
1. Detection and collection	Malicious files are available and detect by the AV solutions.	Since, the malicious files are unavailable, it required to do in-depth memory analysis for the identification of malicious programs running in memory and collect malicious patterns as evidence.
2. Examination	Perform static and dynamic analysis of the malicious sample to extract indicators of compromises (IOCs).	To establish and validate the attack, all pieces of evidence, such as network events, logs of all security tools, and hosts are required to examine.
3. Analysis & investigation	Co-relate the intention of the attacker from the IOCs and investigate the attacker IP through mapping with IP-geolocation.	Co-relate the intention of the attacker from the IOCs and investigate the attacker IP through mapping with IP-geolocation.
4. Incident response	The accurate response of malicious activity should be communicated to mitigate the threat.	The accurate response of malicious activity should be communicated to mitigate the threat.

malicious scripts are interpreted files such as JavaScript, Visual Basic, and PowerShell, which also need to consider as a parameter of detection engine. However, some points raise even more significant questions for the researcher (Gorelik & Moshailov, 2017):

- Are we want to scan all text files, scripts, and XML files?
- Are we to build a parser/interpreter for each type of interpreted files?
- Are we to block any suspicious string, even if it is just a comment in the scripts?

Researchers are facing these questions when trying to balance false positive in early detection systems. The detection system must consider different attack vectors and source of evidence to block fileless attacks at the pre-execution stage. It must leverage the machine-learning algorithm to analyze command lines, scrutinize internet connections, monitor process behavior and protect the memory space of the running process to detect such attacks (Stop Fileless Attacksat Pre-execution, 2017). The system should collect all the relevant data from different sources to feed on the machine learning algorithm and get the best out of it (Tobergte & Curtis, 2013; Tian et al., 2019e; Kumar et al., 2019b). The data from the different sources may be quite large, so to handle the data, the system can use big data technologies (Sudhakar & S.K., 2018) with a machine learning algorithm for more efficient results.

Examination

The system is compromised with fileless malware having less possible to find traces of malicious activity. Although the evidence must be collected from different sources. Data fusion of all the evidence and logs collected from various security tools deployed in each host on the entire network is major challenges faced by the security researchers (Ren, 2004). The traces from the logs, attack vectors from various tools and reconnaissance of attributes from different hosts validate an attack. Characterization of anomalous network events, the malicious behavior of the system, and distinguishing attack traffic from legitimate traffic by searching for patterns of anomalies is a significant challenge (Stop Fileless Attacksat Pre-execution, 2017).

Analysis & Investigation

The large volume of data can be used to scrutinize to understand the relationship of attack and attacker intention, to do so the classification and clustering techniques can be used on the suspected data events to separate the benign events and malicious events (Aljaedi et al., 2011). Pattern recognition can be used to find anomalies in the malicious events. A malicious cluster can help to categorize the attack patterns and attack reconstruction to uncover the motivation and intention of the attacker can be a challenge (Almulhem, 2009).

All the collected evidence needs a thorough investigation to find the source IP of the attacker. Security researchers are facing problems to trace back the attacker IP address due to the advanced techniques of IP spoofing (Mitropoulos & Dimitrios Patsos, 2005). Identifying a mechanism to find the IP location mapping with attacker geolocation is itself a significant challenge (Nikkel, 2007).

Incident response

In the whole process, the incident response is the phase where the result is reflected. The response should be quick and accurate so that the malicious activity should be mitigated and attacker unable to damage any further (Khurana et al., 2009; Aljaedi et al., 2011).

Conclusion

Security defenders should have a significant focus on detecting and preventing fileless malware attacks. The attackers use the legitimate application to fulfill their malicious motives. Since, the PowerShell and WMI can also be used to bypass signature-based detection systems, maintain persistence or ex-filtrate data makes it difficult to detect malicious activities. In this paper, we

categorized most of the existing detection methods and types of fileless malware, which are targeting the real world. The proliferation of non-malware attacks has only accentuated this issue. Major global hacks against SWIFT (Society for Worldwide Interbank Financial Telecommunication) and the Ukraine power grid, among others, have served as clarion calls that critical infrastructure and worldwide financial systems will continue to be targeted by this type of sophisticated attacks.

Acknowledgements
Not applicable.

Author's contribution
The first author conceived the idea of the study and wrote the paper; all authors discussed and revised the final manuscript. All authors read and approved the final manuscript.

Funding
Not applicable.

Availability of data and materials
Not applicable.

Competing interests
The authors declare that they have no competing interests.

Author details
[1]School of Computer & Systems Sciences, Jawaharlal Nehru University, 110067, New Delhi, India. [2]Indian Computer Emergency Response Team, Ministry of Electronics & Information Technology, 110003, New Delhi, India. [3]School of Computer & Systems Sciences, Jawaharlal Nehru University, 110067, New Delhi, India.

Received: 29 July 2019 Accepted: 23 December 2019
Published online: 14 January 2020

References

About the Metasploit Meterpreter (2019). https://www.offensive-security.com/metasploit-unleashed/about-meterpreter/
Afianian A, Niksefat S, Sadeghiyan B, Baptiste D (2018) Malware dynamic analysis evasion techniques: A survey. arXiv preprint arXiv 1811:01190
Aljaedi A, Lindskog D, Zavarsky P, Ruhl R, Almari F (2011) Comparative analysis of volatile memory forensics: Live response vs. memory imaging. In: 2011 IEEE Third International Conference on Privacy, Security, Risk and Trust and 2011 IEEE Third International Conference on Social Computing, pp 1253–1258. https://doi.org/10.1109/PASSAT/SocialCom.2011.68
Allman, M., Ostermann, S.: FTP security considerations (1999). https://tools.ietf.org/html/rfc2577
Almulhem, A.: Network forensics: Notions and challenges. In: 2009 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT), pp. 463–466 (2009). doi:https://doi.org/10.1109/ISSPIT.2009.5407485. IEEE
Bhasin V, Kumar S, Saxena P, Katti C (2018) Security architectures in wireless sensor network. Int J Inf Technol:1–12. https://doi.org/10.1007/s41870-018-0103-6
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
Burdach M (2006) Physical memory forensics. USA, Black Hat
Case A, Richard GG III (2017) Memory forensics: the path forward. Digit Investig 20:23–33
Cohen M (2017) Scanning memory with Yara. Digit Investig 20:34–43. https://doi.org/10.1016/j.diin.2017.02.005
Danyliw R, Householder A (2001) CERT advisory CA-2001-19: "code red" worm exploiting buffer overflow in IIS indexing service DLL
Demystifying Fileless Threats (2019). https://www.mcafee.com/enterprise/en-in/lp/endpoint/fileless-attacks.htm
Falliere, N., Murchu, L.O., Chien, E.: W32. stuxnet dossier. White paper, Symantec Corp., Security Response 5(6), 29 (2011)
Farwell JP, Rohozinski R (2011) Stuxnet and the future of cyber war. Survival 53(1):23–40. https://doi.org/10.1080/00396338.2011.555586
Fileless Malware - A Behavioural Analysis Of Kovter Persistence (2016). https://airbus-cyber-security.com/fileless-malware-behavioural-analysis-kovter-persistence/
Golovanov, S.: A unique `bodiless' bot attacks news site visitors (2012). https://securelist.com/a-unique-bodiless-bot-attacks-news-site-visitors-3/32383/
Gorelik, M., Moshailov, R.: Fileless Malware: Attack Trend Exposed (2017). http://blog.morphisec.com/fileless-malware-attack-trend-exposed Accessed 2018-05-02
Graeber M (2015) Abusing windows management instrumentation (WMI) to build a persistent, asynchronous, and fileless backdoor. Black Hat, Las Vegas
Khurana H, Basney J, Bakht M, Freemon M, Welch V, Butler R (2009) Palantir: a framework for collaborative incident response and investigation. In: Proceedings of the 8th symposium on identity and trust on the internet, pp 38–51 ACM
Kumar S, Dohare U, Kumar K, Prasad Dora D, Naseer Qureshi K, Kharel R (2019a) Cybersecurity measures for geocasting in vehicular cyber physical system environments. IEEE Internet Things J 6(4):5916–5926. https://doi.org/10.1109/JIOT.2018.2872474
Kumar S, Singh K, Kumar S, Kaiwartya O, Cao Y, Zhou H (2019b) Delimitated anti jammer scheme for internet of vehicle: Machine learning based security approach. IEEE Access 7:113311–113323
Living Off The Land Binaries And Scripts - (LOLBins and LOLScripts) (2019). https://github.com/LOLBAS-Project/LOLBAS
Mansfield-Devine S (2017) Fileless attacks: compromising targets without malware. Netw Secur 2017(4):7–11
Mansfield-Devine, S.: The malware arms race. Computer Fraud & Security 2018(2), 15{20 (2018)
Mitropoulos S, Dimitrios Patsos CD (2005) Network forensics towards a classification of traceback mechanisms. In: Workshop of the 1st International Conference on Security and Privacy for Emerging Areas in Communication Networks, pp 9–16
MS SQL Slammer/Sapphire Worm (2003). https://www.giac.org/paper/gsec/3091/ms-sql-slammer-sapphire-worm/105136
Nikkel BJ (2007) An introduction to investigating IPv6 networks. Digit Investig 4(2):59–67. https://doi.org/10.1016/J.DIIN.2007.06.001
O'Murchu L, Gutierrez FP. The evolution of the fileless click-fraud malware poweliks. Symantec Corp. (2015)
Patten, D.: The evolution to fileless malware (2017). http://www.infosecwriters.com/Papers/DPatten Fileless.pdf
Pavković N, Perkov L Social engineering toolkit—a systematic approach to social engineering. In: 2011 proceedings of the 34th international convention MIPRO 2011 may 23 (pp. 1485-1489). IEEE
Phase Bot - A Fileless Rootkit (Part 1) (2014). https://www.malwaretech.com/2014/12/phase-bot-fileless-rootki.html
Phase Bot - A Fileless Rootkit (Part 2) (2014). https://www.malwaretech.com/2014/12/phase-bot-fileless-rootkit-part-2.html
Pilli ES, Joshi RC, Niyogi R (2010) Network forensic frameworks: survey and research challenges. Digit Investig 7(1–2):14–27. https://doi.org/10.1016/j.diin.2010.02.003. 1112.6098
Pontiroli SM, Martinez FR (2015) The tao of .net and powershell malware analysis. In: Virus Bulletin Conference
Process Hollowing (2019). https://attack.mitre.org/techniques/T1093/
Rani R, Kumar S, Dohare U (2019) Trust evaluation for light weight security in sensor enabled internet of things: game theory oriented approach. IEEE Internet Things J 6(5):8421–8432. https://doi.org/10.1109/JIOT.2019.2917763
Ren, W.: On A Network Forensics Model For Information Security. ISTA, 229–234 (2004)
Carbon Black Threat Report: Non-malware attacks and Ransomware take center stage in 2016 (2016). https://www.carbonblack.com/2016/12/15/carbon-black-threat-report-non-malware-attacks-ransomware-takecenter-stage-2016/
Rhodes KA (2001) Code red, code red II, and SirCam attacks highlight need for proactive measures. GAO Testimony Before the Subcommittee on Government Efficiency
Rivera BS, Inocencio RU (2015) Doing more with less: a study of fileless infection attacks. VB 2015
Ruff N (2008) Windows memory forensics. J Comput Virol 4(2):83–100
Informational Series: What is Fileless malware? (2019). https://www.carbonblack.com/resources/definitions/what-is-fileless-malware/

Shackleford D (2016) Active breach detection: the next-generation security technology? SANS institute information security Reading room. In: SANS Institute Information Security Reading Room
Shulmin, A., Prokhorenko, M.: Lurk banker Trojan: exclusively for Russia (2016). https://securelist.com/lurk-banker-trojan-exclusively-for-russia/75040/
Sood, A.K., Enbody, R.J., Bansal, R.: The art of stealing banking information - form grabbing on fire (2011). https://www.virusbulletin.com/virusbulletin/2011/11/art-stealing-banking-information-form-grabbing-fire
SQL Slammer (2019). https://en.wikipedia.org/wiki/SQL Slammer
Stop Fileless Attacksat Pre-execution (2017). https://explore.bitdefender.com/solution-briefs/stop-fileless-attacks-pre-execution
Sudhakar P, S.K. (2018) An approach to improve load balancing in distributed storage systems for NoSQL databases: MongoDB. In: Pattnaik PK, Rautaray SS, Das H, Nayak J (eds) Progress in computing, analytics and networking. Springer, Singapore, pp 529–538
Tan Q, Gao Y, Shi J, Wang X, Fang B, Tian Z (2018) Toward a comprehensive insight into the eclipse attacks of tor hidden services. IEEE Internet Things J 6(2):1584–1593
Team CTG (2017) Threat spotlight: the truth about fileless malware [blog post]
Tian Z, Cui Y, An L, Su S, Yin X, Yin L, Cui X (2018) A real-time correlation of host-level events in cyber range service for smart campus. IEEE Access 6:35355–35364
Tian Z, Gao X, Su S, Qiu J, Du X, Guizani M (2019b) Evaluating reputation management schemes of internet of vehicles based on evolutionary game theory. IEEE Trans Veh Technol 68(6):5971–5980. https://doi.org/10.1109/TVT.2019.2910217
Tian Z, Li M, Qiu M, Sun Y, Su S (2019d) Block-def: a secure digital evidence framework using blockchain. Inf Sci 491:151–165
Tian Z, Luo C, Qiu J, Du X, Guizani M (2019e) A distributed deep learning system for web attack detection on edge devices. IEEE Transactions on Industrial Informatics
Tian Z, Shi W, Wang Y, Zhu C, Du X, Su S, Sun Y, Guizani N (2019a) Real-time lateral movement detection based on evidence reasoning network for edge computing environment. IEEE Trans Ind Inform 15(7):4285–4294. https://doi.org/10.1109/TII.2019.2907754
Tian Z, Su S, Shi W, Du X, Guizani M, Yu X (2019c) A data-driven method for future internet route decision modeling. Futur Gener Comput Syst 95:212–220
Tobergte DR, Curtis S (2013) The Art of Memory Forensics, vol 53, pp 1689–1699. https://doi.org/10.1017/CBO9781107415324.004 arXiv:1011.1669v3
Vacca, J.R.: Network Forensics. In: Computer and Information Security Handbook, 2nd edi edn., pp.649–660. Morgan Kaufmann Publishers is an imprint of Elsevier, United States (2013)
Valdez, R., Sconzo, M.: Threat alert: "PowerWare," new Ransomware written in PowerShell, targets organizations via Microsoft word (2016). https://www.carbonblack.com/2016/03/25/threat-alert-powerwarenew-ransomware-written-in-powershell-targets-organizations-via-microsoft-word/
Watson D, Riden J (2008) The honeynet project: data collection tools, infrastructure, archives and analysis. Proceedings - WOMBAT Workshop on Information Security Threats Data Collection and Sharing, WISTDCS 2008:24–30. https://doi.org/10.1109/WISTDCS.2008.11
Wueest, C., Anand, H.: Living off the land and fileless attack techniques (2017). https://www.symantec.com/content/dam/symantec/docs/security-center/white-papers/istr-living-off-the-land-and-fileless-attack-techniques-en.pdf
Zaharia, A.: Understanding Fileless malware infections - the full guide (2016). https://heimdalsecurity.com/blog/fileless-malware-infections-guide/
Zeltser L (2017) The history of Fileless malware-looking beyond the buzzword
Zhang, E.: What is Fileless malware (or a non-malware attack)? Definition and best practices for Fileless malware protection (2018). https://digitalguardian.com/blog/what-fileless-malware-or-non-malware-attack-definition-and-best-practices-fileless-malware
Zou CC, Gong W, Towsley D (2002) Code red worm propagation modeling and analysis. In: Proceedings of the 9th ACM conference on computer and communications security, vol 147, p 138 ACM

Publisher's Note