An Assistive Tool For Fileless Malware Detection

Conference Paper · August 2021

DOI: 10.23919/WAC50355.2021.9559449

5 authors, including:

Vikas Sihag
Sardar Patel University of Police, Security and Criminal Justice, Jodhpur
36 PUBLICATIONS 189 CITATIONS
SEE PROFILE

Gaurav Choudhary
Technical University of Denmark
52 PUBLICATIONS 517 CITATIONS
SEE PROFILE

Manu Vardhan
National Institute of Technology Raipur
88 PUBLICATIONS 965 CITATIONS
SEE PROFILE

Pradeep Singh
National Institute of Technology Raipur
85 PUBLICATIONS 998 CITATIONS
SEE PROFILE

An Assistive Tool For Fileless Malware Detection

Pramod Borana[1],Vikas Sihag[1]*, Gaurav Choudhary[2], Manu Vardhan[3], Pradeep Singh[3]
[1]*Sardar Patel University of Police, Security and Criminal Justice, Jodhpur*
[2]*School of Computing Science and Engineering ,VIT Bhopal University, Bhopal, Madhya Pradesh*
[3]*National Institute of Technology, Raipur, India*
{vikas.sihag,spu18cs05}@policeuniversity.ac.in, {gauravchoudhary7777}@gmail.com,
{mvardhan.cs,psingh.cs}@nitrr.ac.in

***Abstract*—While fileless and file-based malware attacks are two completely different approaches, even their assistance and detection tools vary. Fileless malware executes in a non-traditional way without leaving traces on the file system, thus evading detection engines. They are powerful because of their persistence and evasion methods. To analyze such malware, we propose an assistive tool for forensic examiners. It assists in identifying abnormal process, network and system activities. This paper also presents background knowledge of fileless life cycle and infection methods.**

***Index Terms*—Cyber Security, Malware Analysis, Fileless Malware, Memory Forensics, Memory Resident**

I. Introduction

Malware is often defined as software with malicious intent, that is to interfere with the normal working of the benign process present in the system. Killing processes, data stealing, altering, encrypting, and taking control of basic system functionality are often the purpose of malware. Origin and history of the malware go way back to the mid-seventies where the only threat at the time was the boot sector and the malware spread through the floppy disk. Creeper designed in 1971 is recognized as the first computer virus displayed the message, ”I'm the creeper, catch me if you can!”. The intention behind writing a piece of malware has also changed from experimental pranks to organized crimes. A malware is broadly categorized into virus, worms, trojans, spyware, adware, ransomware, bots, fileless malware and keyloggers based on its working, infection and spreading method.

Anti-malware systems tries to prevent, protect, detect and disinfect the operating system against malware attacks. They scan for files in preferred locations and match them against their malware-signature database. Moreover, being limited to locations on filesystems, it does not consider files residing in process RAM dump. The malware with capabilities to reside in main memory with least file system interaction are known as fileless malware, which can go undetected [10, 19]. It takes advantage of process in the system to facilitate an attack or to collaborate with other malware.

Malware authors infect machines using legitimate softwares (for eg. web browser, PDF readers, flash player, java player) by exploiting vulnerabilities in them. Infection is generally user initiated using email spams, leading to malware being loaded into main memory without its footprint on the host file system [9, 11]. Multiple Windows system tools (such as wmi, msiexec, certutil, wscript and powershell) are expoited by malware for privilege escalation [5, 8]. Figure 1 illustrates timeline of popular fileless malware attacks.

This paper provides a brief study of fileless malware infection methods and their persistence mechanisms. It proposes an assistive tool for analysis of fileless malware. In section II, we discuss existing work on fileless malware in literature. Section III provides background knowledge required for fileless malware analysis. Section IV provides insight into the proposed assistive tool for fileless malware analysis, followed by conclusion and future direction in section VI.

II. Related Works

Recently, malware has gained significant growth and developers adopted the high-level language for creating malicious codes that affect the whole malware industry. The fileless malware has the ability to compromise the system without downloading malicious files or inject any content into the disk. The Fileless malware is attacked by the attack vectors and makes an entry for its persistence. In the emerging growth of fileless malware various researchers community has significantly contributed towards various detection mechanism. Sudhakar and Kumar [5] presented a study on fileless malware and proposed a novel investigative model of incident handling and response. The authors discussed fileless malware behavior and their persistent mechanisms. Steve Mansfield-Devine [7] discussed current trends in the malware industry and how it helps in compromising targets. Saad et al. [12] presented the fileless malware injection by taking advantage of new features in Javascript and HTML5. The authors discussed associated threats of fileless malware in modern web applications by some free and commercial tools.

Dang et al. [1] explored the possibilities of fileless malware in various IoT applications. The authors deployed some IoT honeypot for analyses of fileless malware attacks in the wild. Sanjay et al. [13] discussed the technical details of Fileless malware and also focused on associated attacks, detection, and mitigation techniques. Lee et al. [6] focused on the recent cyber security attacks with the help of fileless malware. The authors classified the techniques and methodologies used in fileless cyberattacks and focused on the responsiveness timeline against such malware. With the help of the infection chain,

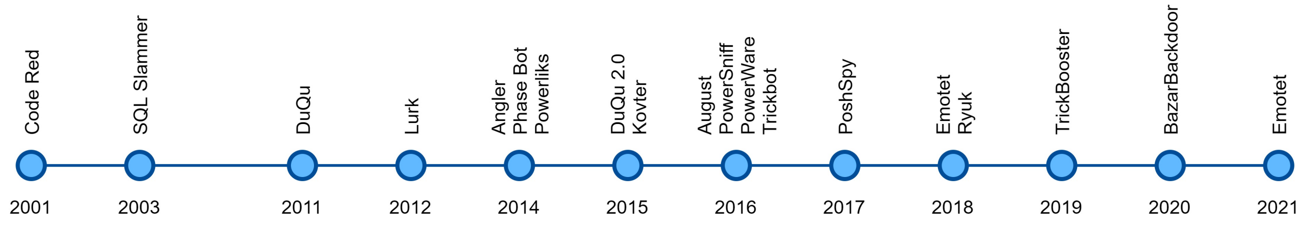

Fig. 1. Timeline of fileless malware evolution

the fileless malware enters into system memory. In this trial, Handaya et al. [2] discussed machine learning methods for the detection of fileless malware in cryptocurrency mining. Jesse Smelcer [18] presented a detailed study of fileless malware and its attack vectors. The author discussed the future research prospects of fileless malware in various domains. Fileless malware is a form of malware that is becoming more dangerous as well as it is difficult to detect in this form of malicious software [4]. Fileless malware will keep on advancing over years and get more troublesome with the accessibility of open-source tools. Over the time frame, the research on modern security frameworks can shield and moderate a wide range of fileless malware attacks.

III. Fileless malware background

With the advancement in technical capabilities over the decades, malware to have evolved themselves. They are continuously evolving and hardening to evade detection and scanning engines [14, 15]. The first known instance of fileless malware is around 2001 with the exploitation of buffer overflow against Microsoft IIS server by Code Red malware. A timeline illustrating evolution of fileless malware is shown in figure 1.

Fileless malware also known as stealthy, zero-footprint, or macro malware, does not use traditional files to spread on the victims machine, rather than that it uses the system benign process against it. Malware code rather than existing on the physical drive exists in the volatile RAM of the system and from where it propagates with the help of process currently in execution. Since it exists in a volatile RAM and works on behalf of a benign process, it is quite difficult to detect it, but on the other hand, it can be stopped in the early phase of attack rebooting or clearing the contents of RAM. Fileless malware being non-persistent in nature, tries to gain it so it may reside in system even after a reboot. As illustrated in figure 2, life cycle of a fileless malware can be divided into malware delivery, persistence and execution.

1) Delivery: During this stage, the attacker tries to deliver and infect the target machine. Techniques employed may include executables, malicious documents with vulnerability expoits, etc. It starts with malicious macro code in files or documents (such as office documents and PDFs). Infection can be user-initiated, by click bating on a spam. Network vulnerabilities and exploit kits can also be used for code injection or commands.

2) Persistence: During this stage malware tries become resident in the system. It does so by making changes to the system registry and setting up scripts that run even after system restart. Tools employed include registry, task scheduler, WMI, etc.

3) Execution: In this stage, the malware performs the intended malicious objective. It misuses tools in the target machine such as msiexec, BITSAdmin and CertUtil. Attacks may use scripts that run directly on the command line for fileless payload delivery and malicious command execution. Malicious scripts are executed via command lines, JavaScript, VBScript, PowerShell, etc.

Common system utilities and tools used by fileless malware are: bitsadmin, certutil, MS Office macros, mshta, msiexec, powershell, psexec, registry, regsvr32, task scheduler, wmi and wsctipt.

IV. Proposed work

In this section, we present the proposed assistive tool for detection of fileless malware. The designed tool is based on runtime system and process information collected from the Windows operating system. The tool was designed on system with Windows 7 OS, 4GB DDR3 RAM, and AMD APU A4 processor. Figure 3 illustrates architecture of the proposed solution. Proposed tool comprises of modules to collect process information and present to the user, based on which a system administrator can take a decision. Listed features are:

- *Process Monitoring:* It collects and lists currently running processes. It provides all the actively running processes and services including both user and admin level services. Furthermore, it include process id, parent process id, up-time, status of the process, priority of process, and process name.
- *Network Monitoring:* It lists active system network connections and IP addresses monitored. It displays information such as network protocol, associated local address, corresponding foreign address, network connection state, associated process id, process name and port number.
- *High priority detection:* This module lists processes with priority higher than normal. Windows OS generally includes priority levels in range of 0-31 [3].
- *Process Termination:* A process can be terminated if found abnormal using it. Termination of a process will

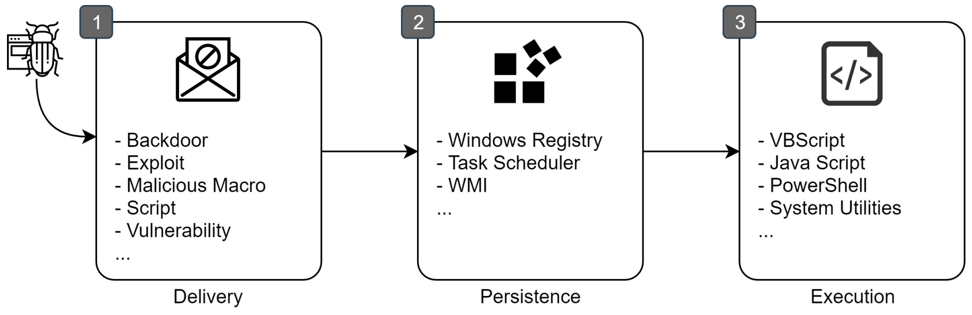

Fig. 2. Lifecycle of a fileless malware

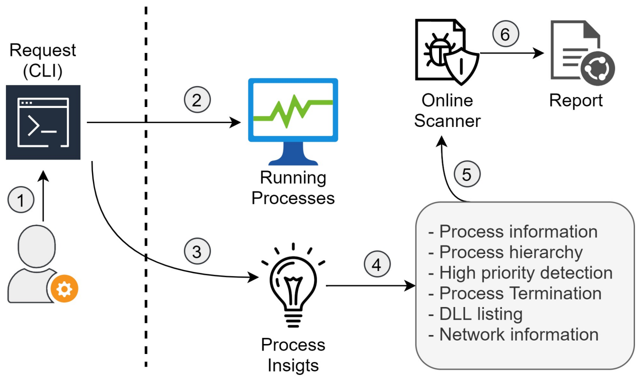

Fig. 3. Architecture of the proposed assistive tool.

terminate all its child processes and threads currently running under that process id.

- *Process Hierarchy:* It displays thread and child process running under the given process id, here this process id is treated as a parent, and all the sub-process and threads are displayed which are currently residing under this id. The list of the process displayed here contains their process id, common parent process id, their individual priority along with their name, and status of the process.
- *Command line arguments:* It displays the argument which were used to call the process. These arguments include all parameters which were passed to them at the time they were called and initialized.
- *Dynamic Link Library:* When provided with process id, this module of code section responds with the list of dynamic link libraries used by the process. When provided with parent process id, it list all the dynamic library by it and it's child processes. With the help of this list of libraries, we can check that whether all the libraries imported are benign or some other malicious libraries are being used, under that benign process.
- *Online Scanner:* This module uploads the process dump to online virus scanning engine API (Virus-Total). Hash value of the application is checked against the existing online database of Virus-Total. If not found the application is uploaded for scanning. The user is redirected to the generated report.

The pseudo-code for the proposed fileless malware detection mechanism is shown in algorithm 1.

Algorithm 1: Fileless Malware Detection Algorithm

```
1  Collect info about active processes;
2  Get pointer to each process;
3  while i ¡= len(all-process) do
4      print(process_info);
5      map = process_info;
6  end
7  Print system network info;
8  Print process network info;
9  Check for unidentified or unknown connection;
10 List high priority process;
11 Check the need for process termination;
12 As per user requirement for specific process, print pid ;
13 print child process ;
14 print command line arguments;
15 print dll's used;
16 active scan for given path;
17 go to step 12;
```

V. Result and Discussion

There is still a talk in research about the future of malware and anti-malware agency/actors and their security frameworks. Recently, automation gained much hype, and with all the fuss about machine learning and AI grab much attraction. The malware authors/ Actors are going to utilize this technology to automate their manual presence even deeper into the system. Near future, If they create user interactive tools that can be used by novice hackers, provoking the field into the mind of youth even more. These will lead in both positive and negative way because while some will be alerted and try to learn to defend themselves and other, a lot more will start to see this as their plaything or say for amusement[17].

While fileless and file-based malware attacks are two completely different approaches, even their assistance and detection tools vary. As there is no lack of detection and assistance tools in the file-based attack, there still a lack of awareness, techniques, and knowledge through which one can thwart the fileless attack. These attacks try to obfuscate their presence or run through scripts, but even more crucial running behind the benign process. That is why most of the time anti-malware software is not able to differentiate between benign and a threat actor.

Keeping the above facts in mind, the objective behind this tool/code is to assist the user, who has a basic understanding of the windows process and service. This tool assists in filtering out any odd running process and it provides a list of the processes that are currently active and with what priority. The proposed tools give the list of child processes or associated processes to a given process id .i.e. a list of all processes currently running under a given process id.

The proposed tool even has a separate code module for network connection, in which it provides the details about the active network connection, local and foreign IP addresses associated. The tools filters out the currently active network connection in the system and the process id they are associated with. It also identifies the active local servers and the port associated along with the process id.

This Code also filters out any process running with priority above the normal base level, which can help to detect any unusual activities/processes currently active on the system. After that, you can terminate them, which is also one such facility provided by the tool. There is also a feature for displaying all dynamic link libraries associated with the process so that one can filter out any odd library injected in-turn filtering out malicious process. Command-line arguments passed between processes are also being monitored. Lastly, and an additional feature is also provided for scanning any directory or file currently present on the system, by providing the path to the file or directory. Once the file or directory is scanned the hash codes are used to match them with online tool virus total for any finding, and results are displayed only for positive finding .i.e. malicious file or application, thus redirecting us to the virus total page for more detail information.

VI. Conclusion

Fileless malware is a form of malware that is becoming more dangerous as well as it is difficult to detect in this form of malicious software. In this paper, we have provided a brief understanding that how fileless malware works and also added a basic assistance module, a java-based program. The proposed tool is a collection of many different modules, which can detect any unusual process activities residing in the system and provide much assistance to the user about any suspicious process or network activity residing or running in the system. Fileless malware will keep on advancing over years and get more troublesome with the accessibility of open-source tools. Over the time frame, the research on modern security frameworks can shield and moderate a wide range of fileless malware attacks. To defend against such malware, the behavioral runtime analysis of the system is an essential requirement because for prevention the system has to be on-guard every time despite the high resource consumption. Including machine learning can be a good solution along with data mining and AI immune system and then specific countermeasure for each of them.

References

[1] F. Dang, Z. Li, Y. Liu, E. Zhai, Q. A. Chen, T. Xu, Y. Chen, and J. Yang, "Understanding fileless attacks on linux-based iot devices with honeycloud," in *Proceedings of the 17th Annual International Conference on Mobile Systems, Applications, and Services*, 2019, pp. 482–493.

[2] W. Handaya, M. Yusoff, and A. Jantan, "Machine learning approach for detection of fileless cryptocurrency mining malware," in *Journal of Physics: Conference Series*, vol. 1450, no. 1. IOP Publishing, 2020, p. 012075.

[3] Karl-Bridge-Microsoft, "Scheduling priorities - windows applications," Microsoft.com, 05 2018. [Online]. Available: https://docs.microsoft.com/en-us/windows/win32/procthread/scheduling-priorities

[4] A. Kim, G. Han, and S.-H. Seo, "Secure and usable bio-passwords based on confidence interval," *Journal of Internet Services and Information Security (JISIS)*, vol. 7, no. 1, pp. 14–27, February 2017.

[5] S. Kumar *et al.*, "An emerging threat fileless malware: a survey and research challenges," *Cybersecurity*, vol. 3, no. 1, pp. 1–12, 2020.

[6] G. Lee, S. Shim, B. Cho, T. Kim, and K. Kim, "Fileless cyberattacks: Analysis and classification," *ETRI Journal*, 2020.

[7] S. Mansfield-Devine, "Fileless attacks: compromising targets without malware," *Network Security*, vol. 2017, no. 4, pp. 7–11, 2017.

[8] A. L. Marra, F. Martinelli, F. Mercaldo, A. Saracino, and M. Sheikhalishahi, "D-bridemaid: A distributed framework for collaborative and dynamic analysis of android malware," *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA)*, vol. 11, no. 3, pp. 1–28, September 2020.

[9] M. Park, S. Kim, and J. Kim, "Research on note-taking apps with security features," *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA)*, vol. 11, no. 4, pp. 63–76, December 2020.

[10] D. Patten, "The evolution to fileless malware," *Retrieved from*, 2017.

[11] S. M. Pontiroli and F. R. Martinez, "The tao of .net and powershell malware analysis," in *Virus Bulletin Conference*, 2015.

[12] S. Saad, F. Mahmood, W. Briguglio, and H. Elmiligi, "Jsless: A tale of a fileless javascript memory-resident malware," in *International Conference on Information Security Practice and Experience*. Springer, 2019, pp. 113–131.

[13] B. Sanjay, D. Rakshith, R. Akash, and V. V. Hegde, "An approach to detect fileless malware and defend its evasive

mechanisms," in *2018 3rd International Conference on Computational Systems and Information Technology for Sustainable Solutions (CSITSS)*. IEEE, 2018, pp. 234–239.

[14] V. Sihag, M. Vardhan, and P. Singh, "Blade: Robust malware detection against obfuscation in android," *Forensic Science International: Digital Investigation*, vol. 38, p. 301176, 2021.

[15] V. Sihag, M. Vardhan, and P. Singh, "A survey of android application and malware hardening," *Computer Science Review*, vol. 39, p. 100365, 2021.

[16] V. Sihag, M. Vardhan, P. Singh, G. Choudhary, and S. Son, "De-lady: Deep learning based android malware detection using dynamic features," *Journal of Internet Services and Information Security (JISIS)*, vol. 11, no. 2, pp. 34–45, 2021.

[17] R. Sihwail, K. Omar, and K. A. Z. Ariffin, "A survey on malware analysis techniques: static, dynamic, hybrid and memory analysis," *International Journal on Advanced Science, Engineering and Information Technology*, vol. 8, no. 4-2, p. 1662, 2018.

[18] J. Smelcer, "Rise of fileless malware," Ph.D. dissertation, Utica College, 2017.

[19] Y. Wang, M. Xiao, Y. Miao, W. Liu, and Q. Huang, "Signature scheme from trapdoor functions," *Journal of Internet Services and Information Security (JISIS)*, vol. 9, no. 2, pp. 31–41, May 2019.