Real-Time Detection System Against Malicious Tools by Monitoring DLL on Client Computers

1st Wataru Matsuda
The University of Tokyo
Tokyo, Japan

2nd Mariko Fujimoto
The University of Tokyo
Tokyo, Japan

3nd Takuho Mitsunaga
The University of Tokyo
Tokyo, Japan

Abstract—The targeted attacks cause severe damage worldwide. Detecting targeted attacks are challenging because the attack methods are very sophisticated. Network-based solutions such as Firewall, Proxy Server, and Intrusion Detection System (IDS) have been widely used. In addition to this, recently, detection methods for malicious programs by monitoring behavior on the endpoints called Endpoint Detection and Response (EDR) have been proposed. Also, some researchers introduce detection methods using DLLs by analyzing suspicious files on the sandbox, such as Cuckoo. Using Cuckoo is one of the solutions for analyzing files that are already identified as malicious. In this research, we propose a real-time detection method of malicious tools using DLL information collected by System Monitor (Sysmon): a free logging tool provided by Microsoft. The purpose of our method is detecting new malicious processes in the production environment. We focus on DLLs commonly loaded by malicious tools regardless of the environments, then propose "the common DLL lists" for detection. Moreover, we introduce a practical detection method that utilizes Elastic Stack as Security Information and Event Management (SIEM). By using Elastic Stack, DLL information loaded on computers can be uniformly monitored and enables real-time detection by comparing logs with the common DLL lists. We evaluate the effectivity of the proposed method using four free malicious tools introduced by US-CERT: China Chopper, Mimikatz, PowerShell Empire, and HUC Packet Transmitter. As a result, our method detected China Chopper, Mimikatz, PowerShell Empire with 100% accuracy. A few false positive occurred for HUC Packet Transmitter, and false positive rate was 0.55%. We confirmed that the common DLL lists are useful for detecting malicious tools in real-time using Elastic Stack.

Keywords Sysmon, DLL, targeted attacks, detection, Elastic Stack

I. Introduction

Detecting targeted attacks are challenging because the methods of these attacks are sophisticated. In targeted attacks, several malicious tools are leveraged by attackers. According to [1], the following five tools are seen in cyber incidents worldwide.

- JBiFrost: A kind of remote access trojan. Once installed on a victim's computer, it allows remote administrative control.
- China Chopper: webshell that has been in widespread use since 2012.
- Mimikatz: It is mainly used by attackers to collect the credentials of other users, who are logged into a targeted Windows machine.
- PowerShell Empire: It is designed for lateral movement after gaining initial access.
- HUC Packet Transmitter: It is a proxy tool used to intercept and redirect Transmission Control Protocol (TCP) connections for obfuscate attackers' communications with victim networks.

Recently detection utilizing Endpoint Detection and Response (EDR)[1] on each computers is getting more common. As part of EDR, malware detection based on file name or hash is one of popular method, however, attackers tend to avoid detection by changing the file name of malicious tools or rebuild them. Therefore, detecting these malicious tools is difficult [2] [3] [4]. To solve the problem, some researchers introduce detection methods using DLLs by analyzing suspicious files on the sandbox such as Cuckoo [5] [6] [7]. DLL (Dynamic Link Library) is a shared library used by Windows processes, and some malicious tools also use legitimate DLLs. If legitimate DLLs loaded by malicious tools have unique characteristics, it can detect malicious tools even if the file name is changed or rebuilt.

In this study, we propose a detection method of malicious files using DLL information collected by Sysmon [8]. Cuckoo is a solutions for analyzing files that are already identified as malicious, on the other hands, our method is for detecting unknown malicious files in the production environment in real-time . We found that DLLs loaded by malicious tools are different depending on the Windows version and the version of malicious tools. Therefore, we investigate the DLLs loaded by the above four tools on each Windows version. We extract DLLs commonly loaded by each tool regardless of the Windows versions (from now on called "the common DLL lists"). We confirmed that the proposed method could detect China Chopper, Mimikatz, PowerShell Empire, and HUC Packet Transmitter with high accuracy. We also introduce a specific method to detect the compromised computers in a timely manner using Elastic Stack: an open-source log analysis tool [9].

II. Attacks using the malicious tools

A targeted attack has a clear objective, such as stealing a specific organization's information. Attackers who can in-

[1]The equipments that focus on identifying and exploring malicious activities on the endpoints

978-1-7281-3306-5/19/$31.00 ©2019 IEEE

trude into the organization tend to expand infections until they accomplish their purpose. Although it is difficult to prevent intrusions, it is possible to minimize the damages if we can detect attack activities in the early stage. It is difficult to detect malicious tools used for targeted attacks. Usually, attackers try to avoid detection such as changing the file name of the malicious tools, rebuild them etc. For example, since source code of Mimikatz is published on the internet, attackers can rebuild it to create their own tools. According to the result of scanning service such as VirusTotal [10] detection rate is poor after rebuilding [3] [11]. The detection rate of Mimikatz without modification was 100%, on the other hand after trivial modification of replacing the word "Mimikatz" to other words "kitikatz," detection rate was only 7.2% (4 out of 54) [11]. It is clear that security products that use the file name or hash value for detection, detection is difficult if attackers customize malicious tools.

III. Previous Research

Several studies have been proposed to detect attacks using DLL information. For related research, there are Several studies to detect malware using Windows API call. In this section, we introduce related research using DLL information, Windows API call and Sysmon.

A. Detecting attacks using DLLs

Cuckoo Sandbox is a malware analyzing system using DLLs [5] [6]. In aspects of using DLLs for detection is the same with our method, but the purpose is fundamentally different. Cuckoo Sandbox is a solution for analyzing files that are already identified as malicious. On the other hand, the purpose of our method is detecting new malicious processes on the production environment. Masoud Narouei et al. propose a malware detection method by analyzing dependency of DLL without execution of the target programs [7]. The method could detect many malicious applications with high accuracy, but need to install developer's tool on the target computers.

B. Detecting attacks using Windows API calls

Windows API is call deeply related with DLL loading. Several researchers focused on Windows API calls by malware. Sun HM. et al. propose a detection method for malicious shellCodes by monitoring API calls [12]. The method can detect anomaly behaviors, but affects the system performance since it needs API hooking[2]. The authors said that their solution decreases 8% - 9% performance. Youngjoon Ki et al. and Carsten Willems introduce malware analysis methods using API call sequence [13] [14]. This research is for analyzing malicious programs deeply that are already identified (similar with III-A).

[2]A technique by which we can instrument and modify the behavior and flow of API calls.

TABLE I
Windows versions for this research

Windows version
Windows Server 2016 (x64)
Windows 10 (x64)
Windows Server 2012 (x64)
Windows 8.1 (x64)
Windows Server 2008 R2 (x64)
Windows 7 (x64)

C. Detecting attacks using DLLs and Sysmon

Ueltschi and Haag introduce a method to detect malicious tools using the Splunk [15] and Sysmon [16] [17]. The method focuses on very specific characteristics of malicious tools. Thus more common approach is needed. Liefer focused on the DLLs loaded by Invoke-Mimikatz[3] and proposed a detection method using Sysmon [18]. We refer this method to detect malicious tools.

D. Issues with previous research

In this section, we describe the issue of the most similar research III-C with our research.

The previous research [18] are mere proofs of the concept that allude to the technical possibilities of detection. Their research has not been evaluated in the exhaustive environment. For example, the loaded DLLs differ depending on the versions of Windows and Mimikatz. Thus, when such DLLs are used for detection in uncertain environments, the results will include false negatives. We evaluated the detection rate using the DLL list introduced by [18]. The test environment consists of the combination of 6 Windows versions shown in Table I and three Mimikatz versions (2.0 alpha, 2.1 and 2.1.1), a total of 18 environments. As a result, we detected only one environment among 18 environments shown in Table II. The one reason for False Negative is that "C: \Windows\System32\apphelp.dll" is introduced as one of DLLs to be detected, but Mimikatz on the Windows 7 does not loaded it. To solve the problem, [19] investigates the differences between DLLs loaded by different versions of Windows and Mimikatz and further proposes DLLs that should be monitored regardless of the environment. We expand this approach to other malicious tools.

IV. Proposed Method

In this research, we evaluate the difference of DLL loading among the combination of Windows versions and malicious tools. We investigate the DLL loading of not only Mimikatz but also China Chopper, PowerShell Empire, and HUC Packet Transmitter. Note that JBiFrost is excluded from our research, since the tool needs to pay for a subscriber fee [20], and should not contribute to attackers. Then we propose the common DLL list for each tool that are commonly loaded regardless of the Windows

[3]A PowerShell module for using Mimikatz without executable file

TABLE II
Detection result on previous research

	Mimikatz 2.1.1 (Aug.1 2017)	Mimikatz 2.1.1 (Jan.7 2018)	Mimikatz 2.1.1 (Mar.6 2018)
Windows 7 64bit	Failed	Failed	Failed
Windows 8.1 64bit	Failed	Failed	Failed
Windows 10 64bit	Failed	Failed	Detected
Windows Server 2008 R2 64bit	Failed	Failed	Failed
Windows Server 2012 R2 64bit	Failed	Failed	Failed
Windows Server 2016 64bit	Failed	Failed	Failed

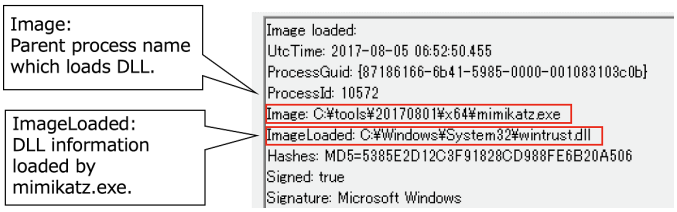

Fig. 1. An example of Sysmon

version and version of malicious tools. Our contribution is that if attackers change the malicious tools' file name or rebuild them, our method can detect them with high accuracy.

A. Detection using Sysmon

Sysmon is a Windows service that remains resident across system reboots to monitor and log system activity to the Windows Event Log, and is useful for recording detail traces of Windows OS. The Windows OS records activities in Event Log. However, detailed information of processes including malicious tools is not recorded under the default configuration. By using Sysmon, it is possible to record detailed information about the execution of processes and loaded DLLs. We use Sysmon v6.03 to collect DLL information. Figure 1 shows an example of Event Log (Event ID: 7) about executed process recorded by Sysmon. "Image" field indicates the executed process name, and "ImageLoaded" field indicates the child process of the process recorded in the "Image" field. In this example, Mimikatz.exe loads wintrust.dll.

B. Method for creating the common DLL lists

The method for creating the common DLL lists is as follows.

1) Install Sysmon (version 6.03) on each evaluation environment shown in Table I.
2) Execute each malicious tool shown in Table III on each environment.
3) Collect Sysmon logs from each environment.
4) Extract DLLs (Image loaded field) from Sysmon logs.
5) Group them by the parent process name.

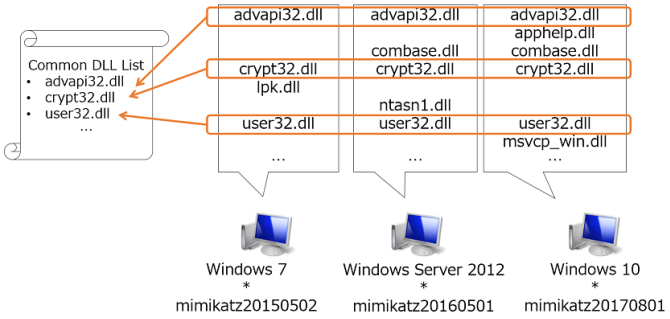

Fig. 2. How to create the common DLL lists

TABLE III
Malicious tools' version for creating the common DLL lists

Tool name	Version
China Chopper	20141213 20111116
Mimikatz	2.1.1 (Aug. 1, 2017) 2.1 (May 1, 2016) 2.0 alpha (May 2, 2015)
PowerShell Empire	2.0 2.3
HUC Packet Transmitter	2003-10-20

6) Extract DLLs loaded by each malicious tool shown in Table III.
7) Extract DLLs that are commonly loaded regardless of the environment, then create the Common DLL List for each malicious tool.

Figure 2 shows conceptual scheme for creating the common Dll lists.

Our decision policy of each malicious tool's version for creating the common DLL lists is as follows. Note that "the latest version" in this paper means the latest version at the time of writing (as of July 2019).

- China Chopper: There are three available versions, we use old two versions for creating the common DLL list, and use the latest version for evaluation.
- Mimikatz: There are many available versions, so we select three versions for creating the common DLL list. 2.0 alpha (May 2, 2015) is the version which was released when many targeted attacks were reported. We also use two versions about a year and two year after the 2.0 alpha (May 2, 2015). Then we use the latest version for evaluation.
- PowerShell Empire: There are many available versions, so we select two versions for creating the common DLL list. 2.0 is the oldest version of 2.x. 2.3 is the latest version in 2017. We use the latest version for evaluation. To control the target with PowerShell Empire, attackers should create program called launcher and lead users to open it [21]. In this research, we collect DLLs loaded by the launcher.
- HUC Packet Transmitter: There is only one available

TABLE IV
The number of DLLs on the common DLL list for each malicious tool

Tool name	The number of DLLs on common DLL list
China Chopper	23
Mimikatz	20
PowerShell Empire	49
HUC Packet Transmitter	8

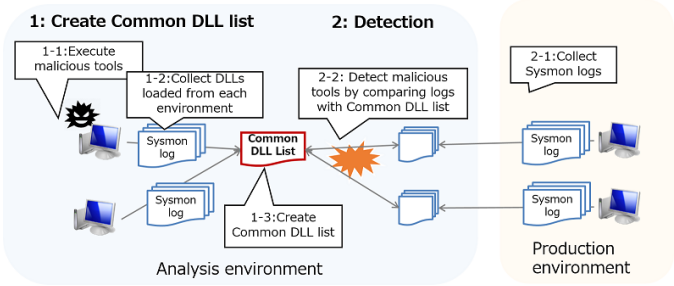

Fig. 3. Detection method using the common DLL lists

version, so we use it both for creating the common DLL list and evaluation.

Table IV shows the number of DLLs on the common DLL list for each malicious tool. All common DLL lists and a tool for creating the common DLL lists are published in [22].

C. Detection using common DLL Lists

The detection method of malicious tools using the common DLL lists is as follows. Figure 3 shows overview of the detection method.

1) Collect Sysmon logs from target computers.
2) Extract DLLs (Image loaded field) from Sysmon logs.
3) Group them by the parent process name (Image filed).
4) Extract DLLs loaded by each process.
5) Compare DLLs with the common DLL lists, and if a process loads all DLLs included in the common DLL lists, judge the process is the malicious tool.

V. Evaluation of the Proposed Method

We evaluated detection accuracy of the proposed method using the common DLL lists discussed in section IV-B. Windows version for evaluation is shown in Table I, and we conduct evaluation on different computers from the computers that used for creating the common DLL lists.

A. The evaluation environment for false negatives

We evaluate whether the proposed method can detect execution of the malicious tools correctly using the common DLL lists. Each malicious tool is executed several times in that environment. As we discussed in subsection IV-B, we selected old versions of malicious tools for creating the common DLL lists. Therefore, we use the latest version (as of July 2019) of malicious tools shown in Table V for evaluation. For HUC Packet Transmitter, only one version (2003-10-20) is released, so we use the same version with which the common DLL list is created.

We also evaluate whether the common DLL lists can detect malicious tools even if the file name of the tool is changed and rebuilt. In our evaluation, we rename and rebuild the Mimikatz source code based on [11] (replace the word from "Mimikatz" to "kitikatz").

TABLE V
Version of malicious tools for evaluation

Tool name	Version
China Chopper	20160622
Mimikatz	2.2.0 (July 20, 2019)
PowerShell Empire	2.5
HUC Packet Transmitter	2003-10-20

TABLE VI
Typical office works

Executed works
Power off / Power on
Log on /Log off
Start and use Outlook (send e-mail)
Start and use Microsoft Word
Start and use Microsoft Excel
Start and use Microsoft PowerPoint
Access the file server
Browse the Internet with Internet Explorer
Start and use the command prompt
Start and use the PowerShell
Windows Update
Connect to a remote desktop

B. The evaluation environment for false positives

We evaluated false positive rate of the proposed method using normal logs during the typical office works shown in Table VI. The versions of Windows for evaluation are shown in Table I. As PowerShell Empire abuses the PowerShell, we also evaluate whether the common DLL list for PowerShell Empire does not detect the execution of legitimate PowerShell by mistake.

C. The evaluation method

1) Extract DLLs (Image loaded field) and process name (Image field) from Sysmon logs on target computers.
2) Extract DLLs loaded by each process.
3) If a process loads all DLLs included in the common DLL lists and process name matches the malicious tool's name, judge as "True Positive."
4) If a process does not load all DLLs included in the common DLL lists and process name does not match any malicious tools' name, judge as "True Negative."

TABLE VII
Evaluation result

	Total	True Positive	True Negative	False Positive	False Negative
China Chopper	4174	7	4167	0	0
Powershell Empire	4223	8	4215	0	0
Mimikatz	4183	49	4134	0	0
HUC Packet Transmitter	4378	18	4336	24	0

5) If a process loads all DLLs included in the common DLL lists and process name does not match any malicious tools' name, judge as "False Positive."
6) If a process does not load all DLLs included in the common DLL lists and process name matches the malicious tool's name, judge as "False Negative."

D. The evaluation result

Table VII shows the evaluation result. The proposed method was able to detect China Chopper, Mimikatz, PowerShell Empire without false detection, and a few false positive occurred for HUC Packet Transmitter.

We confirmed that the common DLL lists for China Chopper, Mimikatz and PowerShell Empire are useful for detecting malicious tools. The following is discussion on the evaluation result.

- The common DLL lists for old version malicious tools could detect the latest version malicious tools.
- The common DLL list for Mimikatz could detect rebuilt Mimikatz. It can be said that even if the source code of malicious tools slightly, our method is useful for detection.
- The common DLL list for Mimikatz could detect not only Mimikatz executable file but also Invoke-Mimikatz(version 2.1) [11].
- PowerShell Empire abuses PowerShell, but false positives did not occur by the common DLL list for PowerShell Empire.
- The common DLL list for HUC Packet Transmitter obtained 24 false positives (false positive rate was 0.55%). The reason is that the number of DLLs on the common DLL list is only eight, this number is quite fewer than other malicious tools. Therefore the possibility of matching DLLs loaded by legitimate application with the common DLL list for HUC Packet Transmitter is higher than other's one.

VI. A practical use for production environments

In this chapter, a we will introduce the specific implementation for practical use of our research.

A. Real-time detection using Elastic Stack

The following is the issue of detections using Sysmon.

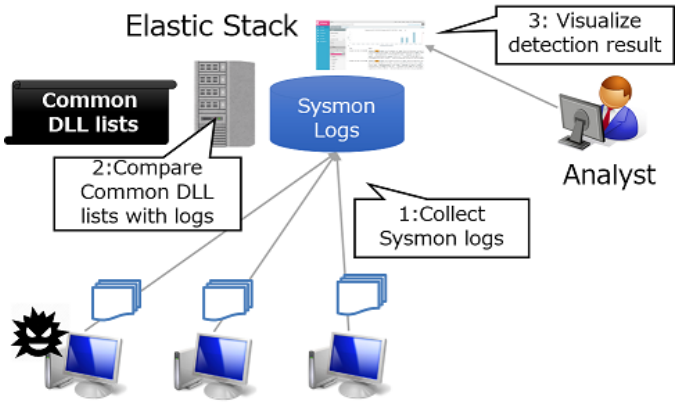

Fig. 4. Online detection method using Elastic Stack

- Sysmon logs are recorded on each computer, and attackers who can compromize the computer could delete Sysmon Logs to hide their attack traces.
- Sysmon logs on all target computers should be collected and analyzed. It takes a lot of operational cost.

To solve these issues, we introduce a example of implementation method using Elastic Stack: log collection and analysis platform. Many studies use Elastic Stack to analyze a large number of logs [23] [24] [25] [26]. We implements the system that collect logs from computers and match them with the common DLL lists using Elastic Stack based on [26]. Figure 3 shows overview of the implementation method using Elastic Stack. The following is advantage for using Elastic Stack.

- Logs are automatically and uniformly collected to Elastic Stack. It is possible to preserve logs and also can reduce operational costs.
- Logs can be collected almost in real-time and visualised. It can help quick incident response.

We confirmed detecting malicious tools in real-time using Elastic Stack. Our implementations are published on GitHub [22].

B. Tools for creating the common DLL lists

DLLs loaded by malicious tools could be changed by some reasons (e.g. major version up for Window or malicious tools), and it leads to false negatives. In preparing for that case, we publish a tool for creating the common DLL lists [22]. This tool can create the common DLL lists from Event Logs exported as CVS files using a Windows built-in function. It is recommended to recreate the common DLL lists when new major version of malicious tools are released or upgrade Windows of office computers.

C. Discussion

As we mentioned in subsection VI-B, loaded DLLS by malicious tools could be changed in the future. The re-creation of the common DLL lists is one of solution but take costs. Now we are investigating characteristics of a time series of loading DLLs of malicious tools. We are

TABLE VIII
The common DLL list for China Chopper

No.	DLL name
1	advapi32.dll
2	bcryptprimitives.dll
3	comctl32.dll
4	cryptbase.dll
5	gdi32.dll
6	imm32.dll
7	kernel32.dll
8	KernelBase.dll
9	mfc42u.dll
10	msctf.dll
11	msvcrt.dll
12	ntdll.dll
13	odbc32.dll
14	ole32.dll
15	oleaut32.dll
16	rpcrt4.dll
17	sechost.dll
18	shell32.dll
19	shlwapi.dll
20	user32.dll
21	wininet.dll
22	winmm.dll
23	ws2_32.dll

considering to use a Deep Learning technique for time series analysis as future work.

VII. Conclusion

In the targeted attacks, attackers tend to avoid detection by changing the file name and rebuilding tools. We introduce the common DLL lists that are commonly loaded by each malicious tool regardless of the environment. It can be said that the common DLL lists are useful to detect malicious tools even if attackers try to avoid detection. In this research, we evaluate only four malicious tools, but our approach could be extended for other malicious tools. Additionally, we introduced the real-time detection method using Elastic Stack to detect compromised computers effectively in the production environment. For future work, we are investigating the use of a Deep Learning technique for a more practical solution.

Appendix
The common DLL list for China Chopper

Table VIII presents the Common DLL list of China Chopper. All common DLL lists are published in [22].

References

[1] CISA. Publicly available tools seen in cyber incidents worldwide. https://www.us-cert.gov/ncas/alerts/AA18-284A, 2018.

[2] James Mulder. The sans institute: mimikatz overview, defenses and detection. https://www.sans.org/reading-room/whitepapers/detection/mimikatz-overview-defenses-detection-36780, 2016.

[3] RenditionSec. Antivirus isn't dead, but you need monitoring too. https://blog.renditioninfosec.com/2017/11/antivirus-isnt-dead-but-you-need-monitoring-too/, 2017.

[4] M. Ussath, D. Jaeger, Feng Cheng, and C. Meinel. Advanced persistent threats: Behind the scenes. In 2016 Annual Conference on Information Science and Systems (CISS), pages 181–186, March 2016.

[5] Joshua Tommy Juwono, Charles Lim, and Alva Erwin. A comparative study of behavior analysis sandboxes in malware detection. 11 2015.

[6] Krishnadeva Kesavan, Sripa Bannakkotuwa, V V Yy Wickramanayake, M P D H De Silva, J M D Fernando, Kalpa Sampath, and Prabath Rupasinghe. Clustermal: Automated malware analysis with clustering, anomaly detection and classification of existing and new behavioral analysis. 08 2016.

[7] Masoud Narouei, Mansour Ahmadi, Giorgio Giacinto, Hassan Takabi, and Ashkan Sami. Dllminer: Structural mining for malware detection. Sec. and Commun. Netw., 8(18):3311–3322, December 2015.

[8] Mark Russinovich and Thomas Garnier. Sysmon v10.2. https://docs.microsoft.com/en-us/sysinternals/downloads/sysmon, 2019.

[9] Elasticsearch. Elasticsearch. https://www.elastic.co/products/elasticsearch, 2019.

[10] Virus total. https://www.virustotal.com, 2018.

[11] Active Directory Security. Unofficial guide to mimikatz & command reference. https://adsecurity.org/?page_id=1821, 2018.

[12] Hung-Min Sun, Yue-Hsun Lin, and Ming-Fung Wu. Api monitoring system for defeating worms and exploits in ms-windows system. In Lynn Margaret Batten and Reihaneh Safavi-Naini, editors, Information Security and Privacy, pages 159–170, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[13] Youngjoon Ki, Eunjin Kim, and Huy Kang Kim. A novel approach to detect malware based on api call sequence analysis. International Journal of Distributed Sensor Networks, 11(6):659101, 2015.

[14] C. Willems, T. Holz, and F. Freiling. Toward automated dynamic malware analysis using cwsandbox. IEEE Security Privacy, 5(2):32–39, March 2007.

[15] Splunk Inc. Splunk. https://www.splunk.com/, 2019.

[16] Tom Ueltschi. Advanced incident detection and threat hunting using sysmon and splunk. In FIRST 2017. Swiss Post CERT, 2017.

[17] Michael Haag. Splunking the endpoint: Threat hunting with sysmon.

[18] Jake Liefer. Detecting in-memory mimikatz. https://securityriskadvisors.com/blog/detecting-in-memory-mimikatz/, 2016.

[19] Wataru Matsuda. Detection of malicious use of mimikatz utilizing sysmon (japanese). In Computer Security Symposium 2017. University of Tokyo III/GSII, 2017.

[20] Rommel Joven and Roland Dela Paz. Jbifrost: Yet another incarnation of the adwind rat. https://www.fortinet.com/blog/threat-research/jbifrost-yet-another-incarnation-of-the-adwind-rat.html, 2016.

[21] George Diathesopoulos. Computer laboratory setup for the assessment of state-of-the-art penetration testing tools. Master's thesis, University of Piraeus, 2017.

[22] sisoc tokyo. attacktooldetection_sysmon. https://github.com/sisoc-tokyo/attackToolDetection_Sysmon, 2019.

[23] U. Jain. Lateral Movement Detection Using ELK Stack. University of Houston, 2018.

[24] S Bagnasco, D Berzano, A Guarise, S Lusso, M Masera, and S Vallero. Monitoring of IaaS and scientific applications on the cloud using the elasticsearch ecosystem. Journal of Physics: Conference Series, 608:012016, may 2015.

[25] D. Chen, Y. Chen, B. N. Brownlow, P. P. Kanjamala, C. A. G. Arredondo, B. L. Radspinner, and M. A. Raveling. Real-time or near real-time persisting daily healthcare data into hdfs and elasticsearch index inside a big data platform. IEEE Transactions on Industrial Informatics, 13(2):595–606, April 2017.

[26] Roberto Rodriguez. Chronicles of a threat hunter: Hunting for in-memory mimikatz with sysmon and elk - part i (event id 7). https://cyberwardog.blogspot.com/2017/03/chronicles-of-threat-hunter-hunting-for.html, 2017.