Mobile and Forensics (MF)
Vol. 3, No. 1, March 2021, pp. 18-27
P-ISSN: 2656-6257, E-ISSN: 2714-6685, DOI: http://dx.doi.org/10.12928/mf.v1i1.3764

MALWARE STATIC ANALYSIS ON MICROSOFT MACRO ATTACK

[1]Redho Maland Aresta *, [2]Ero Wahyu Pratomo, [3]Vicky Geraldino, [4]Achmad Fauzi, [5]Joko Dwi Santoso
[1,2,3,4,5]Department of Computer Engineering, Universitas Amikom Yogyakarta, Indonesia
e-mail: redho.a@students.amikom.ac.id

Abstract

In the 21st century, technology is increasing rapidly, the increase in technology is the potential for cyber attacks on today's technological infrastructure. Malware that is designed to damage computer systems without the owner's knowledge at a considerable cost becomes a cyber crime. This macro malware analysis is to study the code and behavior of malware when run on an operating system. To analyze this malware, this study uses a static analysis method by analyzing malware without running the program.

Keywords: Malware Analysis, Static Analysis, Dynamic Analysis, Cyber attack, Macro Malware

INTRODUCTION

More and more various modes of computer crime are developing, the attack can use a variety of attack techniques [1]. One of the most common of these crimes is a Malware attack, which is an attack that is created with a specific intent to have a detrimental effect. Often malware arrives via file downloads in online use [2].

Due to the very rapid development and frequent occurrence of malware attacks, the ability of malware analysis for forensic investigators [3] and even analysis is a guide in analyzing the malware [4]. Analysis using dynamic and static techniques is one of the solutions that can be utilized. The use of the Macros feature in Microsoft Word can be used in cybercrime [5]. Macros are command tools that use other programs as interpretation for execution [6].

One example of a virus is Melissa, a virus that once made a scene because it used several Microsoft Products (Microsoft Outlook & Microsoft Office) by reproducing so that it was able to spread by sending self-sending via email address book to other users. The recipient of the e-mail will not be suspicious of the person who sent the e-mail because it is known through the e-mail address book when it is opened so that it can infect the recipient's computer [7]. This self-sender activity continues until all the names in the address list. As a result, the SMTP line or email line becomes busy so that it even makes the email server down [8].

The lack of user awareness of a feature or security awareness [9] that can be used as an opportunity by irresponsible people to commit crimes via computer is a problem in this study.

Based on the background problems that have been described, this study will analyze Microsoft Macro Attack Malware using Static Analysis so that the data sent is difficult to solve and the data is safe.

Received January 28, 2021; Revised February 15, 2021; Accepted March 13, 2021

RELATED WORKS

After conducting a study and analysis of several existing journals about Microsoft Macro Malware we will use static analysis to solve Microsoft Macro Attack. Static analysis is used in a way without interpreting existing malware, whereas dynamic analysis is observed when the malware is run on a virtual machine [10]. There are several methods for detecting viruses / malware, namely signature scanning, integrity checking, heuristic scanning. Macro viruses are easy to make but difficult to detect and there are several problems in detecting macro viruses, namely: the number of viruses that are made with high variations, signature scanning is not effective in detecting macro viruses, not all macros are not including viruses. [11] This research will use static and dynamic analysis methods to detect malware that attacks Microsoft Macros.

METHODS

Malware Analysis

Malware analysis is the activity of a series of activities to obtain malware information that is often installed behind programs or general files such as (exe, pdf, doc, xls, jpg, gif, etc.) [12]. There are 2 methods of malware analysis that can be used, static analysis [13] and dynamic analysis [14]. In general, the comparison between static and dynamic analysis can be seen in Table 1.

Basically, dynamic analysis is done by running a program / malware on a virtual machine to get information, behavior, and impacts that occur when malware is executed / executed directly. While static analysis is by seeing and understanding each malware code in detail about the mechanism of action of malware to obtain definite information.

Table 1. Overview of Static and Dynamic Analysis

	Dynamic	**Static**
Overview	Run malware on virtual machines and monitoring the behavior of the malware	Reads code in a binary file and judges the function of each line of malware code.
Output	File system, registry, process, network activity	Boot commands, encode / decode methods
Security Risk	High	Medium
Scope of Analysis	Medium	High

Identify Malware Activity on Microsoft Macros

In general, malware runs by accidentally performing actions such as downloading files, clicking links in e-mails, or visiting less-trusted sites. When hackers create malware, they spread it through free software download services by embedding them in the program. Another way to execute malware is by embedding it on a USB or called Bad USB, when the USB is loaded into the device it will run automatically and cannot be detected easily by the system [15].

Macro viruses are viruses that use the Visual Basic for Application (VBA) language which can take advantage of several facilities that have been provided in Microsoft Office [16]. This virus can produce itself and spreads by sending itself through the e-mall address book so that other users are not suspicious and will repeat itself until the e-mail address book will be sent all so that the e-mail line becomes busy and even down [17].

Finding and Anticipating Malware Macro Attack

Browsers are often used to become a scene of crime [18]. What often happens is the user's negligence in downloading a file with a virus indication. Anticipating this crime, many developers of a browser try to update their browser features to protect users from downloading malicious files [19]. Some browsers can protect against malicious files in the form of malware. One of them is the Google Chrome browser, if you download a file that is indicated as a dangerous file, Google Chrome can protect, and force delete the downloaded file [20].

One of the steps to find malware is by downloading malicious files using a browser. The browser can detect the downloaded files with or without giving a warning notification. In some browsers such as Google Chrome, when downloading a malicious file, a warning notification will appear from the browser as seen in Figure 1. Whereas in a browser like TOR, a warning notification does not appear when downloading a malicious file as shown in Figure 2.

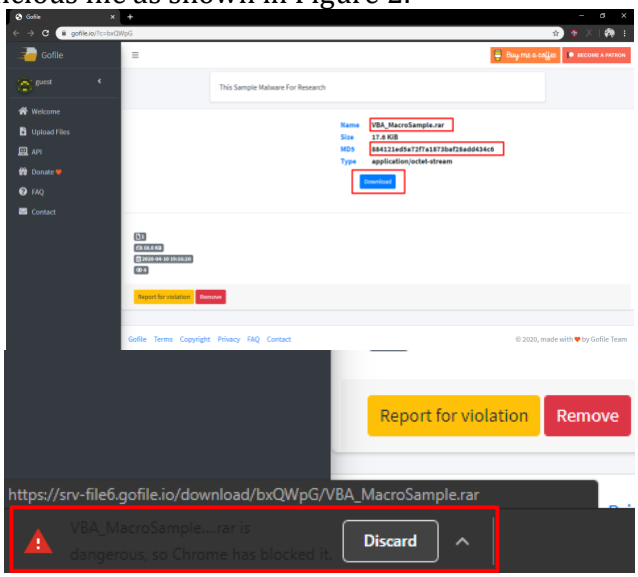


Figure 1. Detect malicious files on the Chrome browser with warning notifications.

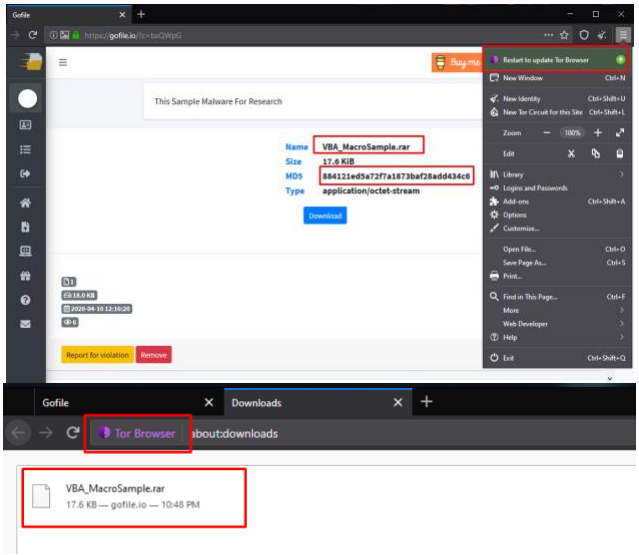


Figure 2. Detect malicious files on the TOR browser without warning notifications.

If from the browser side it is unable to detect a malicious file, another way that can be used is to test it through the virusportal.com site or use an already installed antivirus. Figure 3 shows the steps to detect malicious files via the virusportal.com site.

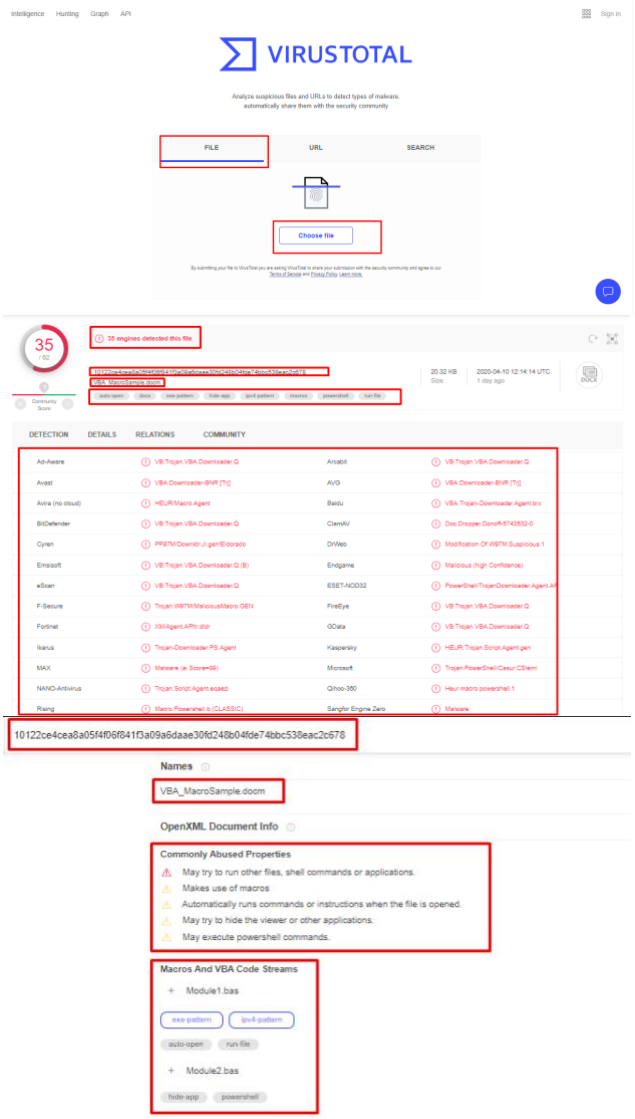


Figure 3. Detect malicious files through virusportal.com

When we find a malicious file, one of the precautionary steps that can be taken is to disable the Macro feature using Microsoft Word. By using the function of "Disable all macros without notification", all macros in a word document will automatically be disabled. Figure 4 shows the steps to disable the Macro feature in Microsoft Word.

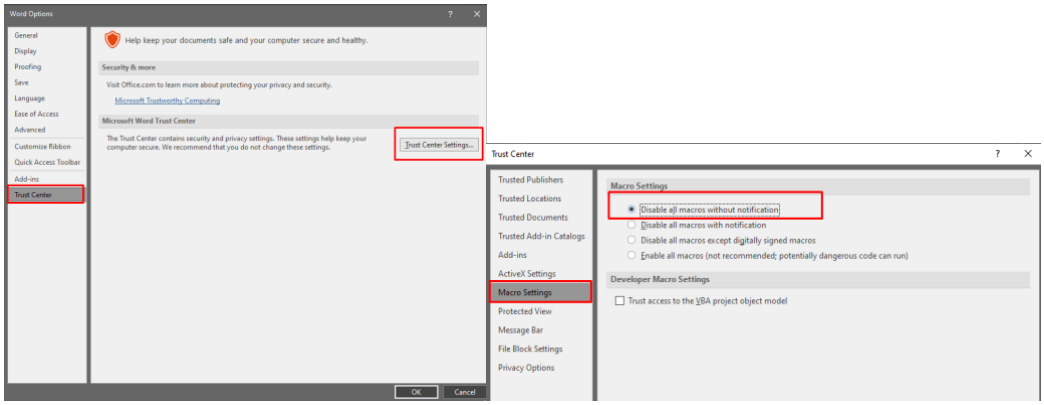


Figure 4. Disable the Macro feature in Microsoft Word.

RESULTS AND DISCUSSION

Generate Sample Backdoor Macro Attack by Empire

By using the Kali linux operating system with Linux version 5.4.0-kali4-amd64 # 1 Debian SMP 5.4.19-1kali1 (2020-02-17) x86_64 GNU / Linux. Creation of VBA Macro Malware Samples generated by Empire as seen in Figure 5. The sample Malware that has been created from Empire is then added with the VBA macro code into the document. The process of adding samples to the VBA Macro Document can be seen in Figure 6.

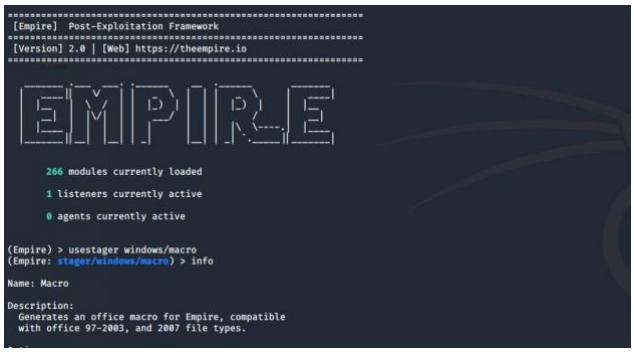


Figure 5. Generate Sample Backdoor Macro Attack

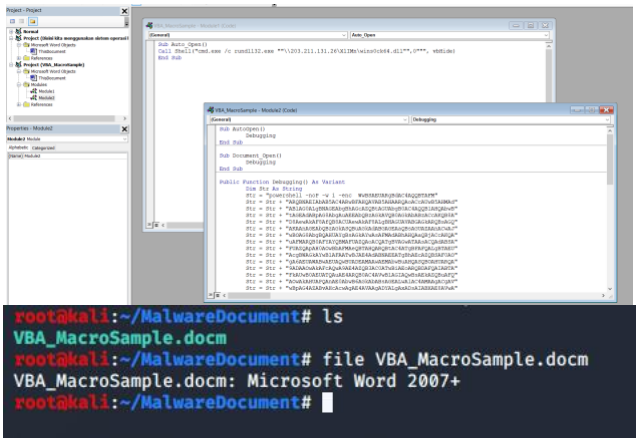


Figure 6. Macro VBA files

The malware samples can be downloaded at https://gofile.io/?c=bxQWpG. In performing the analysis, use the oleVBA tool to extract and analyze VBA macro code embedded in Office documents without the need to run it. To extract the Macro from the document, run the olevba command along with the file to be analyzed, VBA_MacroSample.docm is an Office document that is sampled as malware.

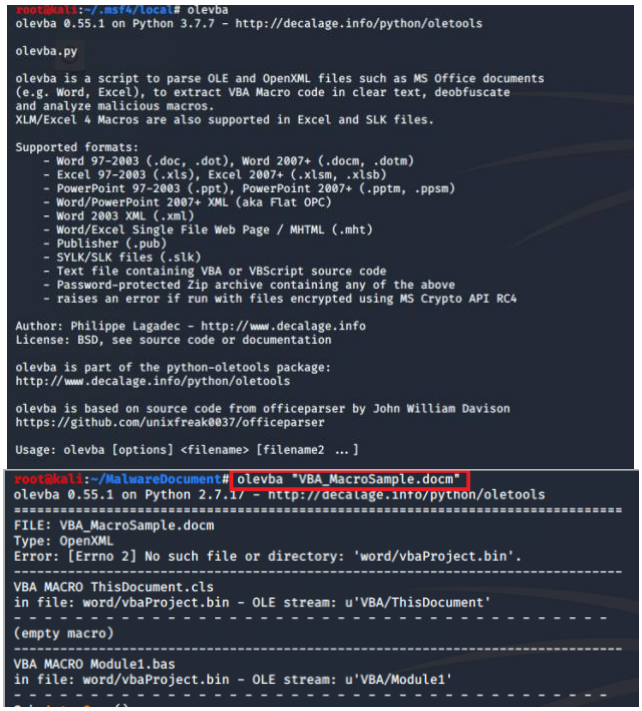


Figure 7. OleVBA Macro

The analysis process starts from the first stream. As we can see from the output, the first stream has a Sub Auto_Open function which is responsible for running macros directly when the document is opened. Then proceed with the macro code that will call the Command Prompt shell command (cmd.exe) to run rundll32.exe to load and run the dynamic link library (.dll) file. The OleVBA Macro output can be seen in Figure 8.

Figure 8. Output OleVBA Macro

Call Shell ("cmd.exe / c rundll32.exe" "\\ 203.211.131.26 \ XlIMn \ wins0ck64.dll" ", 0" "", vbHide)

The file to be loaded and run comes from ip 203.211.131.26 with the file name wins0ck64.dll. The file is malware that will be used by hackers.

Call Shell ("cmd.exe / c rundll32.exe" "\\ 203.211.131.26 \ XlIMn \ wins0ck64.dll" ", 0" "", vbHide)
Researchers used iplocation.com to find and identify the IP addresses used by hackers. The location of the hackers can be seen in Figure 9.

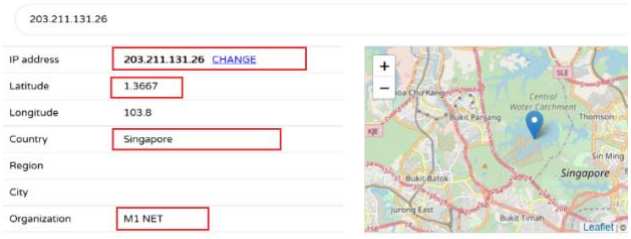


Figure 9. The location of the hackers

We can also see the results of the analysis directly provided by the tools by VBA in the form of a table. For example in the IOC section which provides IP address information and the application that the macro will run. The second part of the stream also has a Sub Auto_Open function which is responsible for running macros directly when the document is opened and followed by a public function containing base64 encoded code. Figure 10 shows the VBA analysis process and the process of creating the malware Str variable.

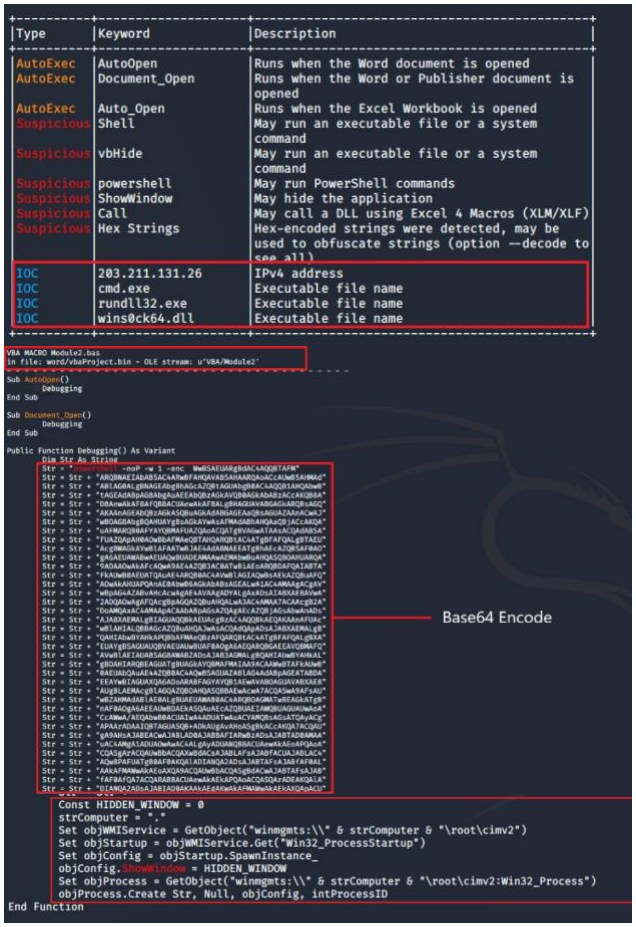
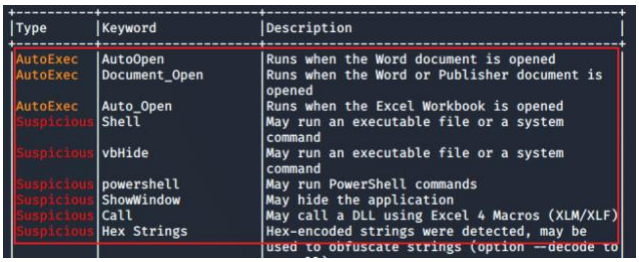


Figure 10. VBA Analysis and Variabel Str Malware

In this macro code, it will create a variable named Str with a Value string that will accommodate the malware load in the form of Base64 Encoding. Furthermore, the macro code will make the application invisible in the window when run using Win32_Process and Win32_ProcessStartup.

The results of the analysis are shown in the Figure 11, there are several keywords that are considered dangerous, such as powershell, vbHide, ShowWindow, and Hex Strings commands.



Type	Keyword	Description
AutoExec	AutoOpen	Runs when the Word document is opened
AutoExec	Document_Open	Runs when the Word or Publisher document is opened
AutoExec	Auto_Open	Runs when the Excel Workbook is opened
Suspicious	Shell	May run an executable file or a system command
Suspicious	vbHide	May run an executable file or a system command
Suspicious	powershell	May run PowerShell commands
Suspicious	ShowWindow	May hide the application
Suspicious	Call	May call a DLL using Excel 4 Macros (XLM/XLF)
Suspicious	Hex Strings	Hex-encoded strings were detected, may be used to obfuscate strings (option --decode to

Figure 11. Analysis Output

Powershell has a function to run powershell commands (interactive command line or object-oriented scripting). VbHide & ShowWindow functions Run applications and commands hiddenly, Hex Strings functions to encode strings so that they cannot be detected, and Call Shell functions to call a command that you want to run, for example a command prompt.

Decodes the malware payload used by hackers to find the hacker's IP address. Figure 12 shows a Malware decodig process.

Figure 12. Decoding Malware

Script 1 is an example of Code (Script):

```
#!/usr/bin/python

import base64
import re

str = "powershell -noP -w 1 -enc  WwBSAEUARgBdAC4AQQBTAFM"
str = str + "ARQBNAEIAbAB5AC4ARwBFAHQAVAB5AHAARQAoACcAUwB5AHMAd"
str = str + "ABlAG0ALgBNAGEAbgBhAGcAZQBtAGUAbgB0AC4AQQB1AHQAbwB"
str = str + "tAGEAdABpAG8AbgAuAEEAbQBzAGkAVQB0AGkAbABzACcAKQB8A"
str = str + "D8AewAkAF8AfQB8ACUAewAkAF8ALgBHAGUAVABGAGkARQBsAGQ"
str = str + "AKAAnAGEAbQBzAGkASQBuAGkAdABGAGEAaQBsAGUAZAAnACwAJ"
str = str + "wBOAG8AbgBQAHUAYgBsAGkAYwAsAFMAdABhAHQAaQBjACcAKQA"
str = str + "uAFMARQB0AFYAYQBMAFUAZQAoACQATgBVAGwATAAsACQAdABSA"
str = str + "FUAZQApAH0AOwBbAFMAeQBTAHQARQBtAC4ATgBFAFQALgBTAEU"
..
..
str = str + "GQAQBUAGEAIAAoACQASQBWACsAJABLACkAKQBBAEkARQBYAA=="
str = str + "="

pars = str.split("enc",1)[1]
data = base64.b64decode(pars)
print(data)
```

```
str = str + "FUAZQApAH0AOwBbAFMAeQBTAHQARQBtAC4ATgBFAFQALgBTAEU"
..
..
str = str + "GQAQQBUAGEAIAAoACQASQBWACsAJABLACkAKQB8AEkARQBYAA="
str = str + "="

pars = str.split("enc",1)
data = base64.b64decode(pars)
print(data)
```

Script 1. Sample Code

Run the script and we get the hacker's IP which is used for Command and control (C&C). The payload will make the victim or victim make an HTTP request to the hacker's server and then the hacker will send the request back along with the payload to get access to the victim's computer. Researchers used iplocation.com to find and identify the IP addresses used by hackers as shown in Figure 13.

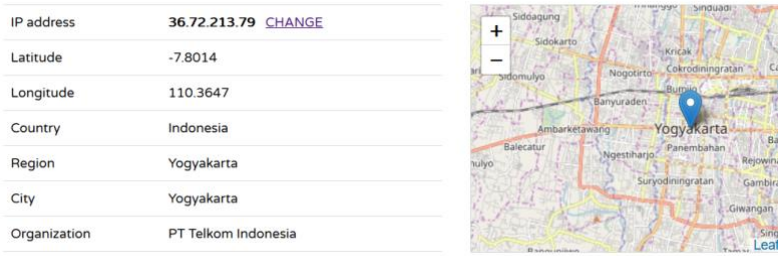


Figure 13. Location of the Victim

CONCLUSION

This Microsoft Macro Attack research is able to provide a clear picture of the characteristics of the malware by using static analysis on each code to obtain complete information about the characteristics of the malware. By using this method, malware can be identified and can determine the next steps. With these results, it is hoped that awareness can be increased to avoid malware attacks that can harm users and know how to avoid this malware.

REFERENCES

[1] S. Saad, W. Briguglio, and H. Elmiligi, 'The Curious Case of Machine Learning In Malware Detection', *arXiv:1905.07573 [cs]*, May 2019, Accessed: Feb. 11, 2022. [Online]. Available: http://arxiv.org/abs/1905.07573

[2] J. C. Sapalo Sicato, P. K. Sharma, V. Loia, and J. H. Park, 'VPNFilter Malware Analysis on Cyber Threat in Smart Home Network', *Applied Sciences*, vol. 9, no. 13, p. 2763, Jul. 2019, doi: 10.3390/app9132763.

[3] H. F. Atlam, E. El-Din Hemdan, A. Alenezi, M. O. Alassafi, and G. B. Wills, 'Internet of Things Forensics: A Review', *Internet of Things*, vol. 11, p. 100220, Sep. 2020, doi: 10.1016/j.iot.2020.100220.

[4] B. Yu, Y. Fang, Q. Yang, Y. Tang, and L. Liu, 'A survey of malware behavior description and analysis', *Frontiers Inf Technol Electronic Eng*, vol. 19, no. 5, pp. 583–603, May 2018, doi: 10.1631/FITEE.1601745.

[5] P. Singh, S. Tapaswi, and S. Gupta, 'Malware Detection in PDF and Office Documents: A survey', *Information Security Journal: A Global Perspective*, vol. 29, no. 3, pp. 134–153, May 2020, doi: 10.1080/19393555.2020.1723747.
[6] M. Elingiusti, L. Aniello, L. Querzoni, and R. Baldoni, 'PDF-Malware Detection: A Survey and Taxonomy of Current Techniques', in *Cyber Threat Intelligence*, vol. 70, A. Dehghantanha, M. Conti, and T. Dargahi, Eds. Cham: Springer International Publishing, 2018, pp. 169–191. doi: 10.1007/978-3-319-73951-9_9.
[7] E. G. Dada, J. S. Bassi, H. Chiroma, S. M. Abdulhamid, A. O. Adetunmbi, and O. E. Ajibuwa, 'Machine learning for email spam filtering: review, approaches and open research problems', *Heliyon*, vol. 5, no. 6, p. e01802, Jun. 2019, doi: 10.1016/j.heliyon.2019.e01802.
[8] A. Karim, S. Azam, B. Shanmugam, K. Kannoorpatti, and M. Alazab, 'A Comprehensive Survey for Intelligent Spam Email Detection', *IEEE Access*, vol. 7, pp. 168261–168295, 2019, doi: 10.1109/ACCESS.2019.2954791.
[9] L. Jaeger, 'Information Security Awareness: Literature Review and Integrative Framework', presented at the Hawaii International Conference on System Sciences, 2018. doi: 10.24251/HICSS.2018.593.
[10] J. G. Drever, B. Third, and C. Sampson, 'The use of static analysis to detect malware in embedded systems', in *8th IET International System Safety Conference incorporating the Cyber Security Conference 2013*, Cardiff, UK, 2013, p. 6.2-6.2. doi: 10.1049/cp.2013.1722.
[11] U. Mishra, 'Detecting Macro Viruses- A TRIZ Based Analysis', *SSRN Journal*, 2012, doi: 10.2139/ssrn.1981892.
[12] D. Ucci, L. Aniello, and R. Baldoni, 'Survey of machine learning techniques for malware analysis', *Computers & Security*, vol. 81, pp. 123–147, Mar. 2019, doi: 10.1016/j.cose.2018.11.001.
[13] S. Sibi Chakkaravarthy, D. Sangeetha, and V. Vaidehi, 'A Survey on malware analysis and mitigation techniques', *Computer Science Review*, vol. 32, pp. 1–23, May 2019, doi: 10.1016/j.cosrev.2019.01.002.
[14] Q.-D. Ngo, H.-T. Nguyen, V.-H. Le, and D.-H. Nguyen, 'A survey of IoT malware and detection methods based on static features', *ICT Express*, vol. 6, no. 4, pp. 280–286, Dec. 2020, doi: 10.1016/j.icte.2020.04.005.
[15] S. K. Nayak and A. C. Ojha, 'Data Leakage Detection and Prevention: Review and Research Directions', in *Machine Learning and Information Processing*, vol. 1101, D. Swain, P. K. Pattnaik, and P. K. Gupta, Eds. Singapore: Springer Singapore, 2020, pp. 203–212. doi: 10.1007/978-981-15-1884-3_19.
[16] M. N. Alenezi, H. Alabdulrazzaq, A. A. Alshaher, and M. M. Alkharang, 'Evolution of Malware Threats and Techniques: A Review', vol. 12, no. 3, p. 12, 2020.
[17] A. Kumar, M. Gupta, G. Kumar, A. Handa, N. Kumar, and S. K. Shukla, 'A Review: Malware Analysis Work at IIT Kanpur', in *Cyber Security in India*, vol. 4, S. K. Shukla and M. Agrawal, Eds. Singapore: Springer Singapore, 2020, pp. 39–48. doi: 10.1007/978-981-15-1675-7_5.
[18] D. Maimon and E. R. Louderback, 'Cyber-Dependent Crimes: An Interdisciplinary Review', *Annu. Rev. Criminol.*, vol. 2, no. 1, pp. 191–216, Jan. 2019, doi: 10.1146/annurev-criminol-032317-092057.
[19] Md. F. Sohan and A. Basalamah, 'A Systematic Literature Review and Quality Analysis of Javascript Malware Detection', *IEEE Access*, vol. 8, pp. 190539–190552, 2020, doi: 10.1109/ACCESS.2020.3031690.
[20] P. Romaniuk, 'Crime and Criminal Justice', in *The Oxford Handbook on the United Nations*, T. G. Weiss and S. Daws, Eds. Oxford University Press, 2018, pp. 514–527. doi: 10.1093/oxfordhb/9780198803164.013.28.