ENHANCING SECURITY IN PYTHON DEVELOPMENT ENVIRONMENTS: DETECTION AND ANALYSIS OF THE EMBEDDED MALWARE

MAJOR PROJECT II

SUBMITTED IN PARTIAL FULFILMENT OF THE AWARD OF THE

DEGREE

OF

MASTER OF TECHNOLOGY

IN

INFORMATION TECHNOLOGY

Submitted By:

Tanya Agarwal

2K22/ISY/21

Under the supervision of

PROF. KAPIL SHARMA

DEPARTMENT OF INFORMATION TECHNOLOGY

DELHI TECHNOLOGICAL UNIVERSITY (Formerly Delhi College of Engineering)

Shahbad Daulatpur, Bawana Road,

Delhi-110042

July, 2024

DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Shahbad Daulatpur, Main Bawana Road, Delhi-42

CANDIDATE'S DECLARATION

I, Tanya Agarwal, hereby certify that the work which is being presented in the thesis entitled "Enhancing Security in Python Development Environments: Detection and Analysis of the Embedded Malware" in partial fulfillment of the requirements for the award of the Degree of Master of Technology, submitted in the Department of Information Technology, Delhi Technological University is an authentic record of my own work carried out during the period from 2022 to 2024 under the supervision of Professor Kapil Sharma.

The matter presented in the thesis has not been submitted by me for the award of any other degree of this or any other Institute.

Candidate's Signature

DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Shahbad Daulatpur, Main Bawana Road, Delhi-42

CERTIFICATE BY THE SUPERVISOR

Certified that **Tanya Agarwal** (2K22/ISY/21) has carried out their search work presented in this thesis entitled **"Enhancing Security in Python Development Environments: Detection and Analysis of the Embedded Malware"** for the award of **Master of Technology** from Department of Information Technology, Delhi Technological University, Delhi, under my supervision. The thesis embodies results of original work, and studies are carried out by the student herself and the contents of the thesis do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

Prof. Kapil Sharma

Professor, IT

(DTU, Delhi)

Date:

ABSTRACT

In our below research, we execute a methodology that filters out file-based and fileless malware from a dataset of malicious Python libraries. We have developed an approach to identify malicious code within these Python libraries. Our approach makes use of static and dynamic analysis in addition to behavioral analysis to accurately distinguish between file-based and fileless Python modules.

Further, we analyze the malicious packages that execute fileless processes. This research aims to contribute towards enhancing the security of Python development ecosystems by providing developers and security professionals with tools to identify and mitigate malware threats effectively. We have also discussed some background information on fileless malwares.

CONTENTS

Candidates Declaration	ii
Certificate by the Supervisor	iii
Abstract	iv
Contents	v
List of Figures	vi
CHAPTER 1 Introduction	1
1.1 General	1
1.2 File-less Malware	2
CHAPTER 2 Literature Review	3
2.1 Literature Review	3
2.2 Malicious PyPI Packages	4
CHAPTER 3 Methodology	7
3.1 Methodology	7
3.2 Fileless Malicious Package Detection Algorithm	9
CHAPTER 4 Results	10
CHAPTER 5 Conclusion, Future Work & Social Impact	12
List of References	13

LIST OF FIGURES

Figure No.	Figure Name	Page No.
Figure 2.1	Malicious package on PyPI	4
Figure 3.1	Proposed system architecture	7
Figure 4.1	Processes created and network interaction by the fileless malware	10
Figure 4.2	The processes created and their interactions with remote IPs	11

INTRODUCTION

With the exportation of most tasks to digital devices, the use of computers has quadrupled in the recent few years. This has led to the spread of understanding and skills regarding the computer networks and systems [1]. To survive in this digital world, the knowledge of computers has become the basic need. This leads to the curious minds to dig into the systems and know more, which may lead to intentional or unintentional cyber-attacks. With digitization, almost every person has a computer in their vicinity, which makes them vulnerable to a cyber-attack. These cyber-attacks may lead to personal information leaking, which in turn leads to serious crimes like asset-theft, compromise of personal accounts, ransom demands, AI-personification, etc. [4][6]

Generally, malwares infect a system as a file. These files are downloaded to the machine when an attack takes place or via an untrustworthy website. When this happens, the malware file are executable codes or a set of executable codes which run to pose an attack [3]. These files wait for a trigger from the attacker or execute automatically after a certain event occurs in the host system. However, these files of the malwares are tracked by the anti-virus software with little effort and are eliminated from the root of the system. Thus, file-based attacks are easily intercepted by the anti-virus software and don’t pose that much of a risk [8]. To overcome this obstacle, the malicious executable code had to be in a form that these anti-virus softwares are not able to intercept them. Thus, the new emerging type of file-less malwares are born.

FILE-LESS MALWARE

With the exportation of most tasks to digital devices, the use of computers has quadrupled in the recent few years. This has led to the spread of understanding and skills regarding the computer networks and systems. To survive in this digital world, the knowledge of computers has become the basic need[1]. This leads to the curious minds to dig into the systems and know more, which may lead to intentional or unintentional cyber-attacks. With digitization, almost every person has a computer in their vicinity, which makes them vulnerable to a cyber-attack. These cyber-attacks may lead to personal information leaking, which in turn leads to serious crimes like asset-theft, compromise of personal accounts, ransom demands, AI-personification, etc [4][6].

Generally, malwares infect a system as a file. These files are downloaded to the machine when an attack takes place or via an untrustworthy website. When this happens, the malware file are executable codes or a set of executable codes run inside the memory to pose an attack [3]. These files wait for a trigger from the attacker or execute automatically after a certain event occurs in the host system. However, these files of the malwares are tracked by the anti-virus software with little effort and are eliminated from the root of the system. Thus, file-based attacks are easily intercepted by the anti-virus software and don’t pose that much of a risk [8].To overcome this obstacle, the malicious executable code had to be in a form that these anti-virus softwares are not able to intercept them.

Thus, the new emerging type of file-less malwares are born.

LITERATURE REVIEW

Advanced research is going on to categorize and detect fileless malware. In [6] RAM resident, script-based malwares which are memory-resident or sometimes execute from within a software are discussed.

GyungMin and others analysed the need of a classification criteria of fileless malwares in [2]. The research identified the absence of categorical reports of fileless malware attacks. Their research proposes classifying the fileless attacks on the basis of the attack technique in accordance with the ATT&CK kill chain by MITRE. The characteristics of the attack like Persistence, Privilege Escalation, etc. are mapped to the ATT&CK kill chain and the algorithm is proposed.

The difficulty of anitvirus softwares to detect fileless malwares is analyzed and smart memory based hardware is proposed in [3]. The in memory and near memory scanning hardware is researched upon to enable continuous scanning of the system memory for suspicious processes. In case a suspicious behavior is detected, the process or program is interrupted from executing.This limitation of disk-based antivirus softwares is addressed here.

Machine Learning models are also analyzed for fileless malware detection by observing the characteristics of the attack and their behavior in the system in [7]. Malmi and Buddhika have analyzed the persistent behavior of fileless malware in [5]. Detection of such fileless malware is proposed using deep learning models. The behavior of malware is analyzed on Cuckoo Sandbox and stored in a dataset. The LSTM and BI-LSTM algorithms are utilized to select the model for malware detection and the accuracy of both algorithms are observed.

MALICIOUS PyPI PACKAGES

In python, we use packages to re-use the functions that have already been built by other developers. This way we can make use of these open-source packages and efficiently utilize our time in more pressing development modules that require our attention. Python packages are basically a collection of modules of the similar type. Whereas, modules contain variables, classes, functions and other structures that are needed to perform an application.

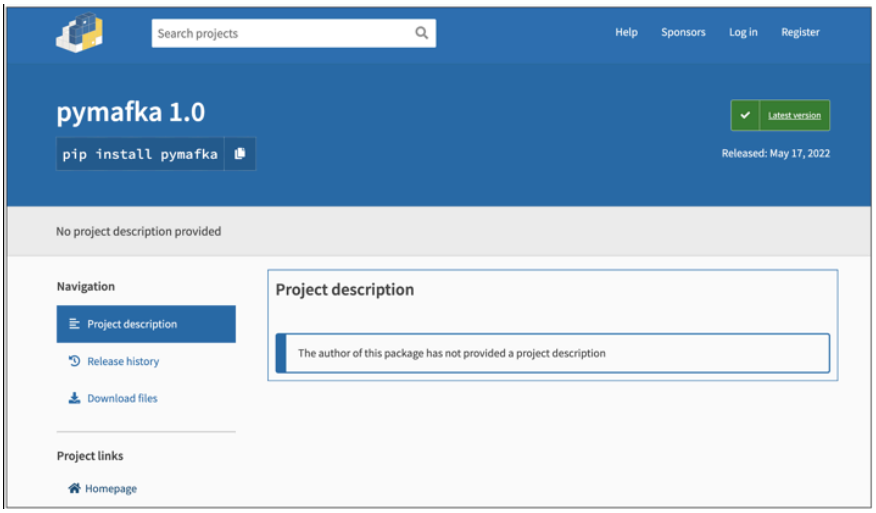

Figure 2.1

This event occurred in May, 2022 and was wiped off from the PyPI registry just after a few hundreds of downloads thankfully. This malicious package stole the developers' credentials and provided access to their system network. With the execution of the 'setup.py' file, the file-less attack started. First, the operating system of the host system is detected, and then the corresponding payload was executed. The developers' credentials and other system and network data was sent to a remote server as was observed in the code.

Further, a reverse shell attack was suspected to take place. On detecting the existence of such a malicious package, the package was took down and inspected. The remote server to which it sent the data was inspected, but was found to be inactive shortly.

Example: Consider below package, ‘Pkg’ that contains modules – ‘Mod1’, ‘Mod2’,….., ‘Modn’

- Pkg
    - Mod1.py
    - Mod2.py
    - .
    - .
    - .
    - Modn.py

These modules, Mod1….Modn contain various data structures and functions that can be imported to our python program files and make use of the inbuilt variables or functions, etc. These packages are open-source and hence, are imported to the program as and when needed. Certain well-known libraries form the basis of any web-application. These are famous and need to be installed to our python programming environment using ‘pip install package-name’.

However, there have been increasing cases where a package with a similar name as the well -known packages are uploaded to the open-source platform but the internal code that runs from the get-go is way different than the code of the original package. This leads to the scenario where the malicious package gets installed and a file-less attack takes place in the

developer's system environment unknown to the owner. PyPI is the open-source repository frequently used by developers to reuse the packages made by fellow developers. Here, developers can also upload their code as packages to contribute to the common utilities required by most developers that fulfill a certain application. These packages can be installed or downloaded to the development environment using the pip function.

In a recent report, it was reported that a malicious python package called 'pymafka' was uploaded on PyPI. This name is similar to the Apache Kafka client package 'pykafka', which as a widespread use in millions over the PyPI platform.

METHODOLOGY

Malicious packages and libraries are installed in the host system and pose the threat of file-less attack. This tool makes use of the fact that file-less malwares run directly in the host system memory in the form of a process without the need of an explicit file-creation. It is designed for Linux system which is widely used by the software developer community in Python and becomes the major target for attacks through Python libraries.

We further analyse the system processes, and collect their network behavior. The security administrators can then make decisions based on this information.

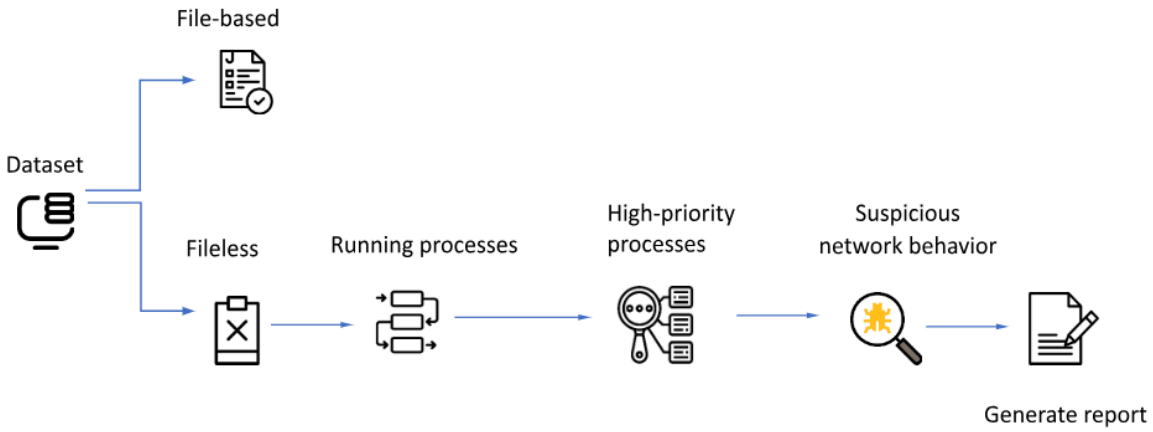

Fig. 3.1

The approach is as follows:

A. Classify Packages: From the dataset (MP), classify the packages that create executable files as File-based malware. Packages that execute directly from the host system process are classified as Fileless malwares.

B. Estimate Memory Consumption: Fileless malwares reside in the host system memory, hence, memory consumption becomes a major criterion to detect whether a package is malicious or not. Fileless malwares consume high memory as they exchange large bytes to and from the attacker.

C. Analyse Network Behavior: Next is to analyse the network properties and the connections being established by these packages. This helps us understand the goal and motive of these malware such as credential-stealing, ransomware, etc.

In a monitored Sandbox environment, a virtual machine is created with Ubuntu version 20.04.6. This operating system being the most widely used for software development serves as the ideal platform for testing the behavior of suspicious PyPI packages.

Next, we create developer credentials and install some common tools and libraries used in software development. We start with a dataset of around 500+ PyPI packages. Using the above playground, we install these packages and observe their file creation behavior. Among these packages, the packages that possess the capability to create an executable file were excluded from further analysis as our focus is on fileless behavior. Further, a consistent filtering method identified approximately 20 packages that exhibit suspicious characteristics without creating a file. These packages are then separated and their behavior is analysed. This subset of approximately 20 packages are then analyzed on following aspects: the impact they have on the attacked system's memory consumption and a thorough examination of of their network behavior and characteristics. For memory consumption, the processes associated with these target packages are monitored. This involved assessing the memory resources

being utilized by the associated processes created by each of theses 20 target PyPI packages by executing them within a Sandbox environment.

FILELESS MALICIOUS PACKAGE DETECTION ALGORITHM

```
1 for pkg in MP:
        if not creates_file(pkg):
                print(pkg)
2 Get reference to each fileless package
3 Print running processes in system
4 Get highest_memory_usage_threshold
5 for p in processes:
        If  ram_usage(p) > high_memory_usage_threshold:
                add p to high_priority_process
6 Print network parameters
7 Print command line arguments
8 Scan network for unknown remote connection.
9 Tabulate the logs.
10 Goto step 4.
```

The network characteristics of these processes created is deeply analyzed for the communication patterns, network traffic, and any suspicious network activities that occur during execution of the packages. Network establishment with any remote IP is flagged along with any data transmission that occurs. This multi-parameter analysis aims to understand the potential threats posed by these 20 packages in a fileless method of execution. The focus on memory and network behavior gives us insights in their operation and the activities associated with these packages. Below is the pseudo code for the proposed method. The below pseudo code outlines the major steps involved in the behavioral analysis for a better understanding if the overall method. The code takes the dataset of 500+ packages as the input and starts filtering out the file-based packages till we are left with fileless packages for our next steps.

RESULTS

Upon analysis of the fileless malware, it was observed that the malicious code interacted with several critical system processes such as the initialization system (initd/systemd), SSH daemon (sshd), and cron. Further, the malware embedded itself inside legitimate processes, camouflaging itself. To observe the processes created or manipulated by these python packages, they are plotted as pie chart along with the network connections with which they interacted with in Fig.2.

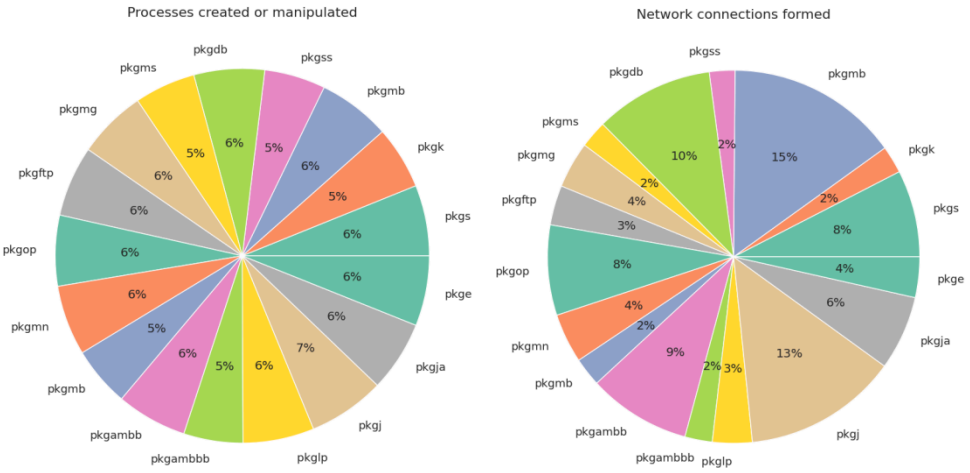

Fig. 4.1

The malwares exhibited suspicious network characteristics such as attempting to connect to external IP, increased load in packets, irregular data exchanges. The ability of such malwares to exploit legitimate system processes poses considerable threats.

Understanding its process interactions and network behavior is important to derive a method to secure systems against such attacks. Below (Fig. 3) depicts the increment in network interactions through creation or modification of additional processes to describe the stealthy nature of such malware attacks.

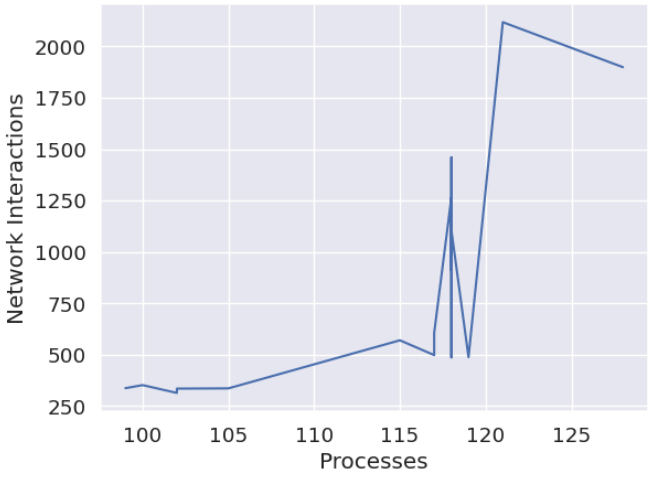

Fig. 4.2

This paper is a contribution to the steps to check a package or a library for malwares, fileless specifically before installing it into the developer system. Packages and libraries are widely used by developers to reuse the already existing code and innovate upon it. Thus, utilizing the resources and skills to create new unique features.

CONCLUSION, FUTURE WORK & SOCIAL IMPACT

The intrusion of fileless malware into developers' systems presents a significant risk by injecting into genuine system functionalities while evading detection. They manipulate the process interactions and maliciously interact over the network. This research becomes important to strengthen defenses for Linux-based systems against these advanced attacks. The file-less attacks through the open-source libraries meant to add to the productivity of the software developers, make the systems and the organizational networks vulnerable to malware attacks.

Use of Machine Learning from the security perspective has to be automated into tools to prevent such attacks from happening.

LIST OF REFERENCES

[1] Atapattu Malmi, Jayawardena Buddhika (2021), An Approach to Detect Fileless Malware that Maintains Persistence in Windows Environment, International Conference on Advances in Computing and Technology (ICACT–2021) Faculty of Computing and Technology (FCT), University of Kelaniya, Sri Lanka 47-52

[2]https://cyborgsecurity.medium.com/python-malware-on-the-rise-39aa75b84848 - By Austin Jackson From Cyborg Security, Jul 14, 2020

[3]https://blog.sonatype.com/new-pymafka-malicious-package-drops-cobalt-strike-on-macos-windows-linux - May 20, 2022 By Ax Sharma

[4] B. N. Sanjay, D. C. Rakshith, R. B. Akash and D. V. V. Hegde, "An Approach to Detect Fileless Malware and Defend its Evasive mechanisms," 2018 3rd International Conference on Computational Systems and Information Technology for Sustainable Solutions (CSITSS), Bengaluru, India, 2018, pp. 234-239, doi: 10.1109/CSITSS.2018.8768769.

[5]https://www.bleepingcomputer.com/news/security/malicious-pypi-package-opens-backdoors-on-windows-linux-and-macs/ - By Bill Toulas May 21, 2022

[6] Botacin, Marcus & Grégio, André & Alves, Marco. (2020). Near-Memory & In-Memory Detection of Fileless Malware. 23-38. 10.1145/3422575.3422775.

[7] Khushali, Vala. (2020). A Review on Fileless Malware Analysis Techniques. International Journal of Engineering Research and. V9. 10.17577/IJERTV9IS050068.

[8] Kim, Kyounggon & Lee, GyungMin & Shim, Shinwoo & Cho, ByoungMo & Kim, Taekyu. (2020). Fileless cyberattacks: Analysis and classification. Etri Journal. 43. 1-12. 10.4218/etrij.2020-0086.

[9] Koutsokostas, Vasilios & Patsakis, Constantinos. (2021). Python and Malware: Developing Stealth and Evasive Malware Without Obfuscation.

[10] P. Borana, V. Sihag, G. Choudhary, M. Vardhan and P. Singh, "An Assistive Tool For Fileless Malware Detection," 2021 World Automation Congress (WAC), Taipei, Taiwan, 2021, pp. 21-25, doi: 10.23919/WAC50355.2021.9559449.

[11] https://resources.infosecinstitute.com/topic/art-fileless-malware/ - May 7, 2018 by Pedro Tavares

[12] Saad, Sherif & Briguglio, William & Elmiligi, Haytham. (2019). The Curious Case of Machine Learning In Malware Detection.

[13] Sudhakar, Kumar, S. An emerging threat Fileless malware: a survey and research challenges. *Cybersecur* **3**, 1 (2020). https://doi.org/10.1186/s42400-019-0043-x

(Formerly Delhi College of Engineering)
Shahbad Daulatpur, Main Bawana Road, Delhi-42

PLAGIARISM VERIFICATION

Title of the Thesis "Enhancing Security in Python Development Environments: Detection and Analysis of the Embedded Malware"

Total Pages: 19

Name of the Scholar: Tanya Agarwal

Supervisor: Prof. Kapil Sharma

Department: Information Technology

This is to report that the above thesis was scanned for similarity detection. Process and outcome is given below:

Software used: Turnitin

Similarity Index: 5%,

Total Word Count: 15625

Date:

Candidate's Signature **Signature of Supervisor(s)**

5% Overall Similarity

Top sources found in the following databases:

- 4% Internet database
- Crossref database
- 2% Submitted Works database
- 2% Publications database
- Crossref Posted Content database

TOP SOURCES

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1	dspace.dtu.ac.in:8080 Internet	2%
2	criminologysociety13.wordpress.com Internet	<1%
3	open.alberta.ca Internet	<1%
4	slideshare.net Internet	<1%
5	repository.kln.ac.lk Internet	<1%
6	GyungMin Lee, ShinWoo Shim, ByoungMo Cho, TaeKyu Kim, Kyounggo... Crossref	<1%
7	University of Bradford on 2021-12-10 Submitted works	<1%