Malware exposed: an in-depth analysis of its behavior and threats

Chinmay Anand,
Dept. of Computer Science and Engineering,
New Horizon College of Engineering (VTU),
Bengaluru, India
1nh19cs037.chinmayanand@gmail.com

Shreya Korada,
Dept. of Computer Science and Engineering,
New Horizon College of Engineering (VTU),
Bengaluru, India
1nh19cs167.shreyakorada@gmail.com

Raksha S.
Department of Computer Science and Engineering,
New Horizon College of Engineering (VTU),
Bengaluru, India
1nh19cs140.raksha.s@gmail.com

Dr. B. Meenakshi Sundaram,
Professor, Department of Computer Science and Engineering,
New Horizon College of Engineering,
bmsundaram@gmail.com

Dr. B. Rajalakshmi
Professor, Department of Computer Science and Engineering,
New Horizon College of Engineering,
dr_rajalakshmi_imprint@yahoo.com

Abstract— **Any software that acts maliciously towards a user, device, or network is referred to as malware. Malware analysis consists of four fundamental processes that make use of a variety of technologies to comprehend operation and pinpoint areas for removal. In order to examine the behavior of the malware on the system, the second phase, known as Basic Dynamic analysis, is running the malware in a secure environment. Running malware in a Sandbox environment is a crucial step in the Basic Dynamic Analysis process. In order to determine which sandbox gives the greatest flexibility for running malware for research, this study will examine a variety of sandboxes. Based on the desired characteristics of a sandbox environment, a rubric was developed. Scalability, the capability to examine different file kinds, and the presence of sandbox detection evasion strategies are a few of the parameters taken into account. To do Basic Dynamic Analysis for malware analysis after examining some of the most well-known sandboxes, Norman Sandbox, GFI Sandbox, and Anubis.**

Keywords— Analysis, malware, Antivirus, Cyber Security, Dynamic analysis, Malware analysis, Malware detection, Malicious

I. INTRODUCTION

Sandboxing, and malware analysis are key strategies in the realm of computer security. Malware analysis is the procedure used to analyze and comprehend the actions and features of malicious software to ascertain how it may affect a system. On the other hand, sandboxing entails isolating the virus in a regulated setting so that you can watch how it behaves and stop it from propagating to other areas of the network. Malware analysis and sandboxing together offer a thorough method for identifying, comprehending, and reducing the threat caused by harmful software. Through this procedure, security professionals may assess the potential harm that malware may cause and take the necessary precautions to stop it from spreading. Organizations can reduce the risk of data breaches, intellectual property theft, and other security issues brought on by malware by implementing these measures.

II. METHODOLOGY

A. Malware Analysis

One of the biggest risks to information security that any group or organization may encounter is malware. In current history, assault volume and sophistication have outpaced available antivirus detection and removal techniques and software. It is crucial to be able to evaluate the malware in an effort to remove it right away without having to wait for the most recent antivirus software because it is a constantly changing threat. Malware can appear in a variety of forms, including viruses, spyware, rootkits, worms, and others.

- Basic Static Analysis: Functionality can be determined by inspecting the executable rather than running it. Resource Hacker, Dependency Walker, Preview, and PEiD are some of the tools used.
- Basic Dynamic Analysis: launching the malware and monitoring the effects it has on the system. GFI Sandbox, Process Monitor, Process Explorer, Regshot, and Wireshark are some of the tools used.
- Advanced Static Analysis: This stage entails loading the malware into a disassembler and studying the program's instructions to reverse-engineer it. The Interactive Disassembler Professional (IDA Pro), one of the most potent disassemblers, is frequently used for sophisticated static analysis.
- Advanced Dynamic Analysis: examines the internal workings of malware that is currently operating using a debugger. OllyDbg is a widely popular x86 debugger used for sophisticated dynamic analysis. WinDbg is an additional debugging tool that is excellent for kernel debugging.

B. Sandboxes:

Various tools are employed in each procedure to assist in the malware investigation. Running the executable in a sandbox environment is the initial step in Basic Dynamic Analysis.[3] A sandbox is a security tool that enables users to run an untrusted program in a secure setting without worrying about damaging the host system or operating system. Sandboxes are a fantastic tool for early screening because they will evaluate malware for free while simulating various network functions to ensure the executable will execute smoothly. [1]Running malware in a sandbox as opposed to solely in a virtual environment has the advantage of producing a report after execution.

There are various types in sandbox:

a) Norman Sandbox:

One of the initial businesses to do research and create proactive cybersecurity software solutions was Norman, and it was Norman that owned the Norman Sandbox Analyzer. Norman was one of the better-known companies to give sandbox software not just to enterprises, but to the common user. Despite being outdated and no longer operating (it was acquired by AVG in 2014), Norman was still used by many users. The Norman Sandbox Analyzer is a tool that, in Norman's words, "is aimed to automate, simplify, and speed up the information-gathering process while analyzing edc malware."[8]

b) GFI Sandbox:

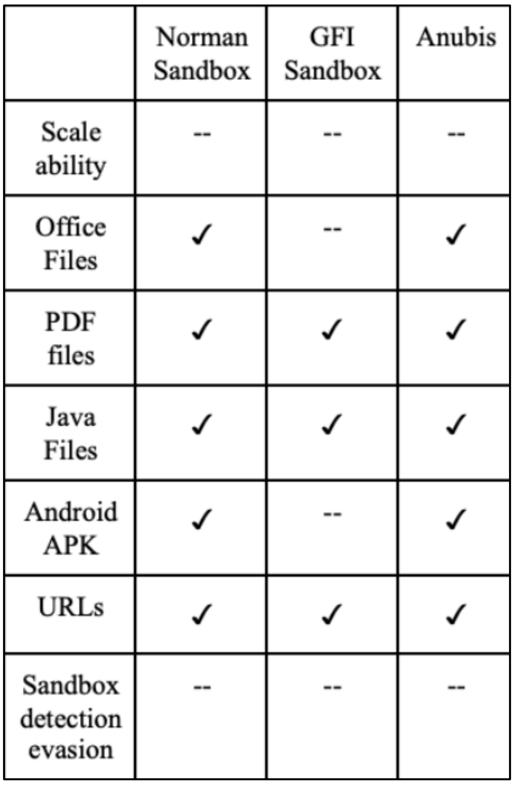
GFI Technology Partners is the owner of GFI software, including GFI Sandbox. This is a collection of businesses that provide their clients with technology goods and services. The GFI Sandbox differs from other sandboxes in a variety of ways, including the following:

- User engagement that is automatic
- Centralized administration of various sandbox configurations
- Section-organized PDF report production[2]

c)Anubis:

Given that the analysis is carried out online, Anubis (Analyzing Unknown Binaries) is a unique kind of sandbox. The Anubis website allows for the analysis and upload of Windows executable files. Four report formats—HTML, XML, PDF, and text—as well as the fact that this tool is an online service and does not require downloading software, are among the advantages of this website. The study of the malware just requires the malware to be uploaded, and the host machine is never used in any way.[7]

Comparison between the sandboxes:



	Norman Sandbox	GFI Sandbox	Anubis
Scale ability	--	--	--
Office Files	✓	--	✓
PDF files	✓	✓	✓
Java Files	✓	✓	✓
Android APK	✓	--	✓
URLs	✓	✓	✓
Sandbox detection evasion	--	--	--

Fig. 1. Comparison between sandboxes

C. Description of all the tools used in the software:

Python:

Python is the programming language used to create and manage virtual machines, among other scripted and automated functions that are part of the software as a whole.

Virtualization Platforms:

VirtualBox, VMware, KVM, and QEMU are just a few of the virtualization platforms on which the software can be used. These systems offer the controlled environment needed for the analysis of the malware.

Operating Systems:

On a variety of operating systems, including Windows, Linux, and Android, it can analyze malware samples. This makes it possible to thoroughly analyze how the infection behaves on various platforms.

YARA:

With the aid of this open-source program, we can recognize and categorize malware according to its traits, such as particular words or file types. It is possible to write and modify YARA rules to meet certain malware analysis requirements.

Volatility:

This memory forensics framework is used to examine the virtual machine's memory while the malware is operating. Volatility can locate malware artifacts that are not apparent in the file system, such as hidden processes, rootkits, and other malware artifacts. This contributes to painting a fuller picture of the behavior of the malware.

Wireshark:

The network traffic produced by the malware sample is captured and examined using this network protocol analyzer. This makes it easier to spot any malware that exhibits network-based behavior, such as communicating with command-and-control servers.

Tcpdump:

This command-line utility is intended to record and instantly analyze network data. Compared to Wireshark, Tcpdump can record packets at a lower level, making it valuable for spotting specific network-based threats.

Suricata:

This open-source intrusion detection system is employed to identify and notify users of any suspicious network activity. Port scans and DDoS attacks are only two examples of the many network-based attacks that Suricata can identify and notify users about.

PyDeep:

This Python package offers malware sample analysis based on machine learning. PyDeep can examine the malware's composition and actions in order to provide light on its nefarious motives.

Virus Total:

Virus Total integration allows for a comparison of the analyzed malware samples with a database of known malware behaviors and signatures. Information on the malware's behavior and probable effects can assist determine whether it is a version of a known threat.

Overall, each of these technologies is essential to the software's performance and aids in providing an in-depth examination of malware samples.

D. Architecture of the Software:

The threat posed by malware and other types of cyber-attacks is growing along with the usage of technology. Because of this, there is an increased need for antivirus software and other software analyzers that can assist users in identifying and avoiding such assaults. In this post, we'll examine the characteristics of antivirus software and how it might assist users in securing their networks and devices.

Real-time scanning of files and programs on a user's device is one of an antivirus tool's key capabilities. The tool can be used by the user to start a scan, verify the status of the scan, and view alerts about potential dangers that have been found. By designating particular scanning schedules, file types, or directories to be scanned, users of some antivirus software can further tailor the scanning procedure. When a potential threat is located, the antivirus program will notify the user with a warning message that includes information about the threat's type and level of risk.[4] After that, the user can take the necessary action, including quarantining or erasing the infected file or application. Sometimes the antivirus program will also advise users to take precautions against infection in the future, such as updating software or changing passwords.

Antivirus technologies can find vulnerabilities in a user's system that an attacker could utilize, in addition to checking for malware and other types of unwanted software. For instance, an antivirus program might spot obsolete software or firmware that is potentially attackable and recommend updates or patches to fix these flaws. Finding the attack's primary cause and addressing it are crucial steps in the effective prevention of malware and other cyberattacks. By giving users thorough information about the danger, including how it entered the system and what it has harmed, antivirus software can assist users in determining the primary cause of an attack. After that, users can take precautions to avoid such assaults in the future, such as updating their security software or putting in place more stringent access controls.

In conclusion, antivirus programs are crucial software analysts that can assist users in defending their networks and devices from malware and other types of online threats. Antivirus solutions help users identify dangers quickly and efficiently by offering real-time scanning, warning messages, and prevention advice. A crucial first step in maintaining the protection of your digital assets is to invest in a reliable antivirus program given the constant threat of cyberattacks.

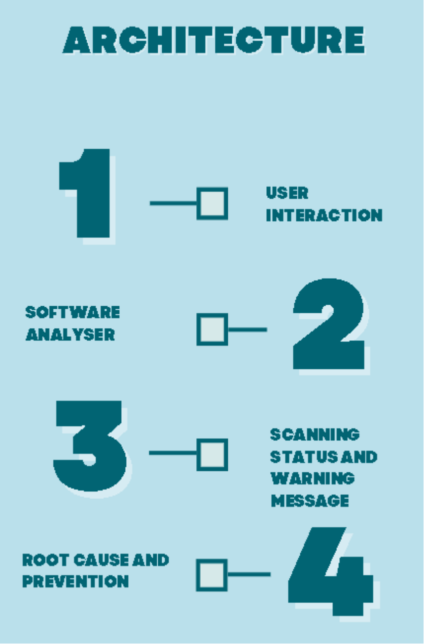

ARCHITECTURE
1
USER INTERACTION
SOFTWARE ANALYSER
2
3
SCANNING STATUS AND WARNING MESSAGE
ROOT CAUSE AND PREVENTION
4

Fig. 2. The Software Architecture

E. In what way is malware analysis achieved:

Static and dynamic analysis approaches are used in combination in our research to analyze malware.

Static analysis entails looking through the malware's binary code without actually running it. It extracts details about the file, such as the file type, import and export routines, and embedded resources, using a range of static analysis methods.[6]

These techniques are used to perform the static analysis:

File analysis: Analyzing the header, resources, imports, and exports of the malicious file. This can assist in determining the file type, any exploits or known vulnerabilities it employs, and any embedded or encrypted code.

Hash analysis, which involves creating a hash of the malicious file and comparing it to a database of other known harmful hashes. This can facilitate the quick identification of malware samples.

Signature analysis: This technique allows for the identification of specific patterns or code sequences found within malicious software. This can aid in the detection of particular malware families or variations.[16]

Metadata analysis: Investigate any metadata connected to the malicious file, including the file name, file size, and file creation date. This data can be used to locate the malware and any other linked files' original locations.

Static analysis is important for spotting specific traits or trends in malware code as well as for swiftly recognizing known malware samples.

Dynamic analysis is running the malware in a controlled setting in order to watch how it behaves. It establishes a virtual machine environment where the malware sample can be run and its actions observed. Our project also gathers a variety of data during execution, including system calls, network activity, and file system modifications.[13]

One can perform dynamic analysis using the following techniques:

Monitoring malware: behavior during its execution in a controlled virtual environment. This involves keeping an eye on any changes to the system, network activity, file system, registry, and other activities.

API hooking: you can inject code into an active process and snoop on calls to system APIs. As a result, Cuckoo Sandbox is able to keep an eye on and control the behavior of the virus while it is running.[14]

Memory analysis: Examining the memory of an active process to find any potentially executing harmful or suspicious code. This can assist in locating malware that is memory-resident or covert and might be challenging to find using conventional techniques.[15]

Traffic Analysis: Analysis of network traffic produced by the malware, such as connections to other networks or contacts with remote servers, is known as traffic analysis. Any malicious network activity, such as command-and-control (C2) communication, can be recognized with the aid of this.

Dynamic analysis is helpful for discovering any new or previously undiscovered malware as well as for determining the behavior and capabilities of the malware in real time. Dynamic analysis, however, can be resource-intensive and may be ineffective against some malware varieties that are engineered to avoid detection.[5]

III. IMPLEMENTATION RESULTS AND DISCUSSION:

A. YARA:

A well-known open-source tool for malware analysis and detection is called YARA. It is a rule-based application that gives analysts the ability to spot malware patterns and traits within files and processes. In order to identify and category's malware, YARA gives users the ability to build and use custom rules that are defined in a syntax resembling regular expressions. In addition to analyzing Office documents and PDF files, the application can look for patterns in network traffic, memory dumps, and binary files. YARA is incredibly adaptable, enabling users to build intricate rules that may identify certain malware strains as well as variations in packers and obfuscation methods. [12]Security researchers and incident response teams frequently utilize it in their work to find and examine malware.

B. TCP DUMP:

On Unix-like operating systems, a common command-line packet capture program called Tcpdump is used for network analysis. Users can record and view network traffic either in real-time or from a stored file. Tcpdump records packets at the level of the network interface, allowing users to study the behavior of network protocols finely.[9]

There are numerous methods for filtering and viewing network packets using Tcpdump. IP addresses, port numbers, and protocol kinds are just a few examples of the criteria that users might use to filter packets. Tcpdump can be used for security-related network traffic analysis, such as spotting possible risks or assaults.

Users can use the Tcpdump command to capture packets and then examine them with other programs like Wireshark or Tshark in order to analyze network traffic. These programs give users the ability to look at the specifics of network packets, such as the data payload and header data.

To sum up, Tcpdump is an effective tool for capturing and examining network data, and it is essential for network analysis and troubleshooting. Network administrators and security experts can examine network behavior, identify problems, and identify security concerns using tcpdump to record and filter packets.[10]

C. NETWORK ANALYSIS:

Network analysis is the process of looking at and evaluating information about computer networks to learn more about their performance, security, and usage patterns. Packet capture, traffic monitoring, and log analysis are just a few of the techniques and tools that can be used during network analysis to gather and examine network data.

Understanding how data moves via a network, identifying any problems or inefficiencies, and spotting any security concerns are the objectives of network analysis. Network analysis can be used to optimize network infrastructure, enhance network security, and address network performance problems.

Detecting and avoiding network assaults, checking network traffic for adherence to security policies, and troubleshooting problems with network devices or applications are a few typical use cases for network analysis.

In general, network analysis is a crucial part of network administration and security, and IT pros and security specialists use it to make sure that computer networks are dependable, available, and secure.

Error values are used in network analysis to describe the state of network communications and to identify network issues. In network analysis, various distinct kinds of incorrect values could appear, including

Response codes: These indicate the status of a web server response to a client request. Common response codes include 200 OK, 404 Not Found, and 500 Internal Server Error.

Packet loss: This occurs when data packets are lost or dropped during transmission, which can cause delays or errors in data transfer.

Latency: This refers to the time it takes for data to travel from one point in the network to another. High latency can cause slow network performance, while low latency is desirable for real-time applications like video conferencing.

Transmission errors: These can occur when data is corrupted or lost during transmission due to noise, interference, or other factors.[11]

DNS errors: These occur when there is an issue with the domain name system (DNS) that translates domain names into IP addresses, preventing users from accessing websites or other network resources.

D. PACKET CAPTURE:

The technique of intercepting and recording data packets that are transmitted over a computer network is known as packet capture. The procedure entails gathering information about network traffic that is sent between servers, routers, and client devices.

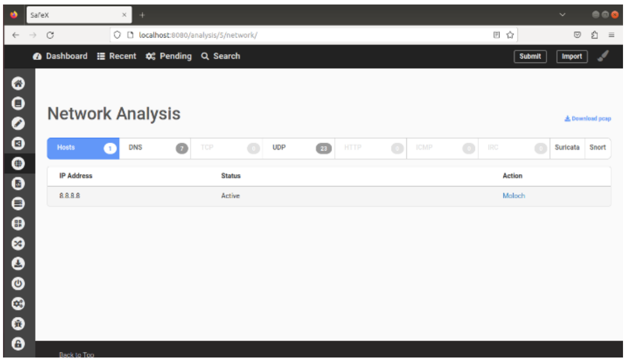
In order to examine network protocol behavior, find network problems, and spot security threats, packet capture is frequently used in network analysis and troubleshooting. To identify performance issues or network failures, captured packets can be studied to learn how network devices are interacting with one another.

A multitude of programs and techniques, including tcpdump, Wireshark, and hardware packet sniffers, can be used to accomplish packet capture. These tools may filter packets based on many factors, including IP addresses, port numbers, and protocol kinds, and can record packets in real time or from a saved file.

In conclusion, packet capture is a crucial method for deciphering network behavior, identifying problems with the network, and identifying security concerns. It is frequently used by IT professionals and security specialists to maintain the dependability, availability, and security of computer networks and plays a crucial role in network administration and security.

IV. OUTCOME AND DISCUSSION:

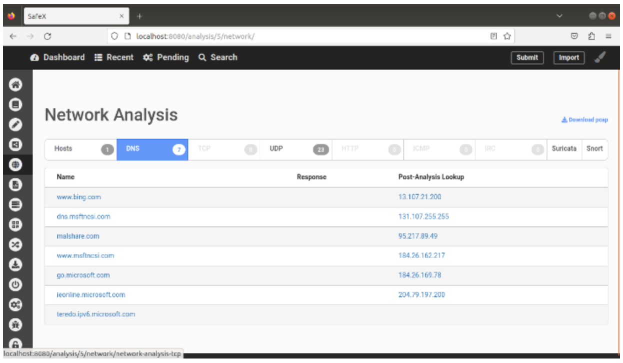
We begin with the network analysis:



IP Address	Status	Action
8.8.8.8	Active	Moloch

Fig. 3. The network Analysis

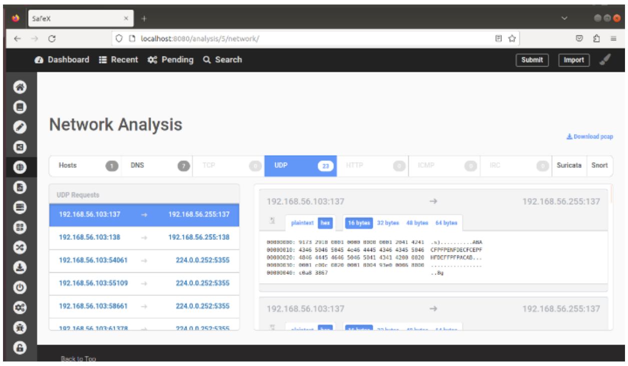
Moreover, DNS values and analysis are included in the network analysis.



Name	Response	Post-Analysis Lookup
www.bing.com		13.107.21.200
dns.msftncsi.com		131.107.255.255
malshare.com		95.217.89.49
www.msftncsi.com		184.26.162.217
go.microsoft.com		184.26.169.78
ieonline.microsoft.com		204.79.197.200
teredo.ipv6.microsoft.com		

Fig. 4. DNS Values

The UDP values and byte distribution are provided:



UDP Requests		
192.168.56.103:137	→	192.168.56.255:137
192.168.56.103:138	→	192.168.56.255:138
192.168.56.103:54061	→	224.0.0.252:5355
192.168.56.103:55109	→	224.0.0.252:5355
192.168.56.103:58661	→	224.0.0.252:5355
192.168.56.103:61378	→	224.0.0.252:5355

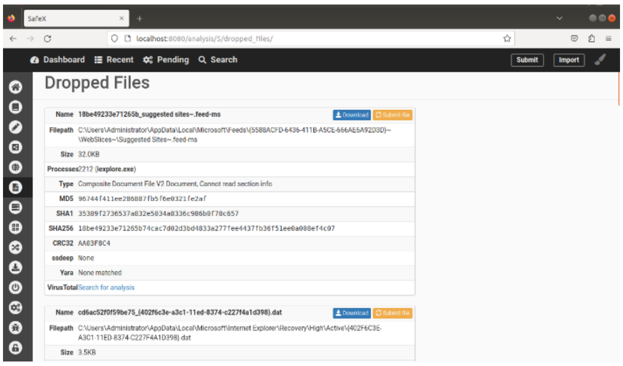
192.168.56.103:137 → 192.168.56.255:137

```
00000000: 9173 2910 0001 0000 0000 0001 2041 4241  .s)..........ABA
00000010: 4346 5046 5045 4e46 4445 4346 4345 5046  CFPFPENFDECFCEPF
00000020: 4846 4445 4646 5046 5041 4341 4200 0020  HFDEFFPFPACAB...
00000030: 0001 c00c 0020 0001 0004 93e0 0006 0000  ................
00000040: c0a8 3867                                ..8g
```

192.168.56.103:137 → 192.168.56.255:137

Fig. 5. UDP Values

The size of the file and the documents we've added or imported from our system for analysis:



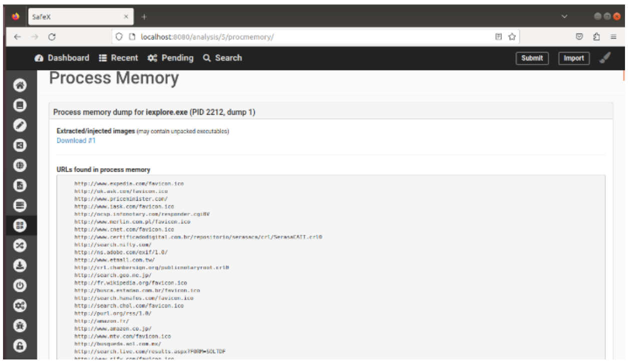
Dropped Files

Name	18be49233e71265b_suggested sites~.feed-ms
Filepath	C:\Users\Administrator\AppData\Local\Microsoft\Feeds\{5588ACFD-6436-411B-A5CE-666AE6A92D3D}~\WebSlices~\Suggested Sites~.feed-ms
Size	32.0KB
Processes	2212 (iexplore.exe)
Type	Composite Document File V2 Document, Cannot read section info
MD5	96744f411ee286887fb5f6e0321fe2af
SHA1	35389f2736537a832e5034a8336c986b8f78c657
SHA256	18be49233e71265b74cac7d02d3bd4833a277fee4437fb36f51ee0a088ef4c07
CRC32	AA03F8C4
ssdeep	None
Yara	None matched
VirusTotal	Search for analysis

Name	cd6ac52f0f59be75_{402f6c3e-a3c1-11ed-8374-c227f4a1d398}.dat
Filepath	C:\Users\Administrator\AppData\Local\Microsoft\Internet Explorer\Recovery\High\Active\{402F6C3E-A3C1-11ED-8374-C227F4A1D398}.dat
Size	3.5KB

Fig. 6. File Analysis

Where the URLs are located in the process memory list:



Process Memory

Process memory dump for iexplore.exe (PID 2212, dump 1)

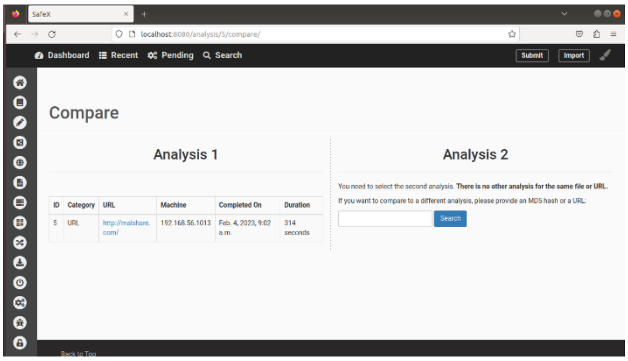
Extracted/injected images (may contain unpacked executables)
Download #1

URLs found in process memory

```
http://www.expedia.com/favicon.ico
http://uk.ask.com/favicon.ico
http://www.priceminister.com/
http://www.iask.com/favicon.ico
http://ocsp.infonotary.com/responder.cgi0V
http://www.merlin.com.pl/favicon.ico
http://www.cnet.com/favicon.ico
http://www.certificadodigital.com.br/repositorio/serasaca/crl/SerasaCAII.crl0
http://search.nifty.com/
http://ns.adobe.com/exif/1.0/
http://www.etmall.com.tw/
http://crl.chambersign.org/publicnotaryroot.crl0
http://search.goo.ne.jp/
http://fr.wikipedia.org/favicon.ico
http://busca.estadao.com.br/favicon.ico
http://search.hanafos.com/favicon.ico
http://search.chol.com/favicon.ico
http://purl.org/rss/1.0/
http://amazon.fr/
http://www.amazon.co.jp/
http://www.mtv.com/favicon.ico
http://busqueda.aol.com.mx/
http://search.live.com/results.aspx?FORM=SOLTDF
```

Fig. 7. Process Memory

Evaluating and comparing URLs:



Compare

Analysis 1

ID	Category	URL	Machine	Completed On	Duration
5	URL	http://malshare.com/	192.168.56.1013	Feb. 4, 2023, 9:02 a.m.	314 seconds

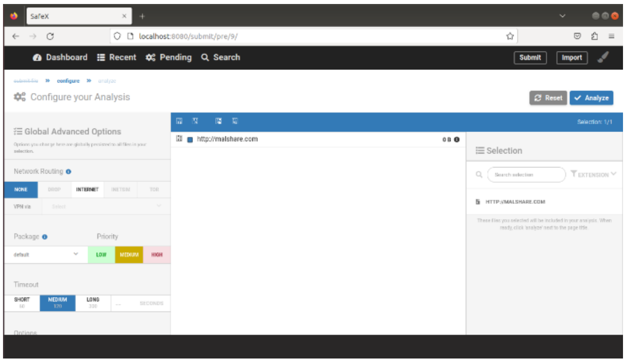
Analysis 2

You need to select the second analysis. **There is no other analysis for the same file or URL.**

If you want to compare to a different analysis, please provide an MD5 hash or a URL:

Fig. 8. Comparison of URLs

Analysing and configuring the URL:



Configure your Analysis

Global Advanced Options

Options you change here are globally persisted to all files in your selection.

Network Routing: INTERNET

VPN via: Select

Package: default

Priority: MEDIUM

Timeout: MEDIUM 120

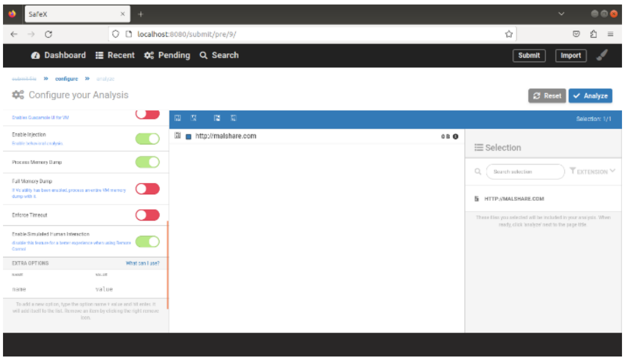
http://malshare.com 0 B

Selection

HTTP://MALSHARE.COM

These files you selected will be included in your analysis. When ready, click 'analyze' next to the page title.

Fig. 9. Analysing the URL



Configure your Analysis

Enables Guacamole UI for VM

Enable Injection
Enable behavioral analysis

Process Memory Dump

Full Memory Dump
If Volatility has been enabled, process an entire VM memory dump with it.

Enforce Timeout

Enable Simulated Human Interaction
disable this feature for a better experience when using Remote Control

EXTRA OPTIONS

name	value
name	value

To add a new option, type the option name + value and hit enter. It will add itself to the list. Remove an item by clicking the right remove icon.

http://malshare.com 0 B

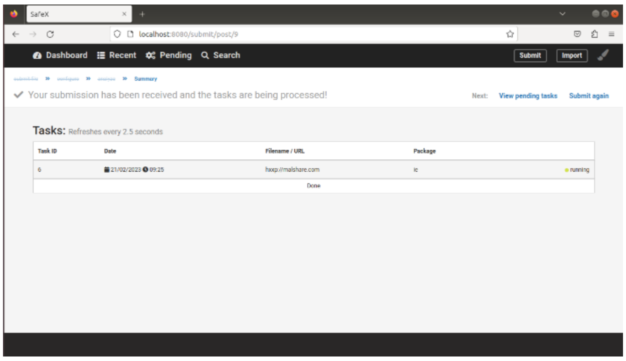
Selection

HTTP://MALSHARE.COM

These files you selected will be included in your analysis. When ready, click 'analyze' next to the page title.

Fig. 10. Analysing the URL- II

The processes that are currently being processed:



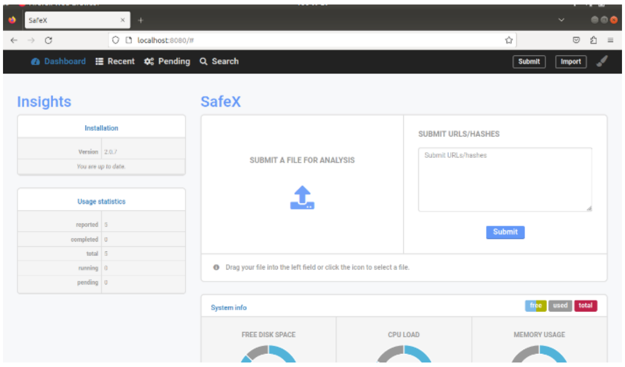
Your submission has been received and the tasks are being processed!

Tasks: Refreshes every 2.5 seconds

Task ID	Date	Filename / URL	Package	
6	21/02/2023 09:25	hxxp://malshare.com	ie	running
Done				

Fig. 11. Active Process

The system's dashboard and monitoring of its disk space:



Insights

Installation

Version	2.0.7

You are up to date.

Usage statistics

reported	5
completed	0
total	5
running	0
pending	0

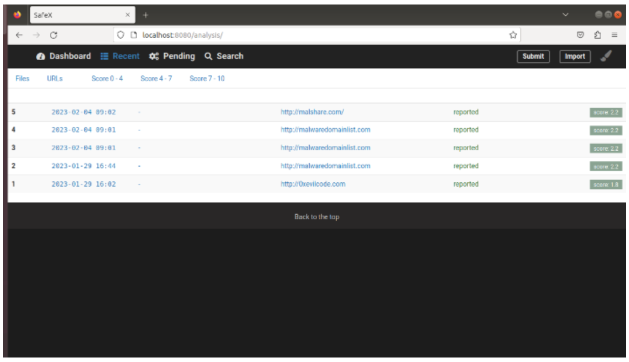
SafeX

Drag your file into the left field or click the icon to select a file.

System info

Fig. 12. The Dashboard

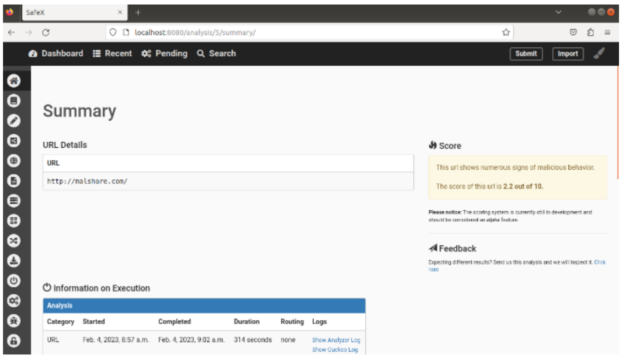
The recently released documents:



5	2023-02-04 09:02	-	http://malshare.com/	reported	score 2.2
4	2023-02-04 09:01	-	http://malwaredomainlist.com	reported	score 2.2
3	2023-02-04 09:01	-	http://malwaredomainlist.com	reported	score 2.2
2	2023-01-29 16:44	-	http://malwaredomainlist.com	reported	score 2.2
1	2023-01-29 16:02	-	http://0xevilcode.com	reported	score 1.8

Fig. 13. Recent Documents

The URL's summation and score value:



Summary

URL Details

URL
http://malshare.com/

Score

This url shows numerous signs of malicious behavior.

The score of this url is **2.2 out of 10.**

Please notice: The scoring system is currently still in development and should be considered an alpha feature.

Feedback

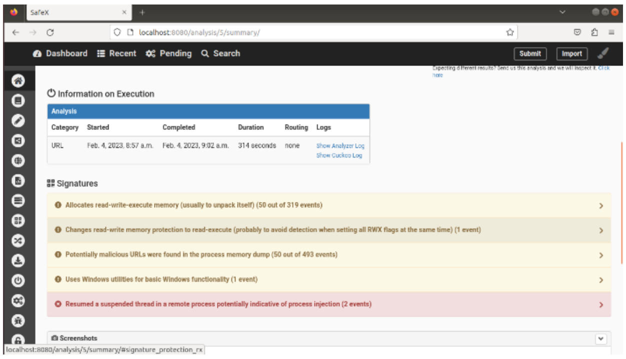
Expecting different results? Send us this analysis and we will inspect it. Click here

Information on Execution

Category	Started	Completed	Duration	Routing	Logs
URL	Feb. 4, 2023, 8:57 a.m.	Feb. 4, 2023, 9:02 a.m.	314 seconds	none	Show Analyzer Log Show Cuckoo Log

Fig. 14. Summation

The URL’s information and execution catalog:


SafeX
localhost:8080/analysis/5/summary/
Dashboard
Recent
Pending
Search
Submit
Import
Screenshots
localhost:8080/analysis/5/summary/#signature_protection_rx

Expecting different results? Send us this analysis and we will inspect it. Click here

Information on Execution

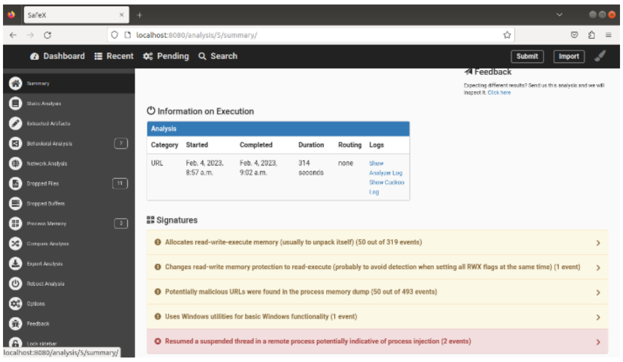
Category	Started	Completed	Duration	Routing	Logs
URL	Feb. 4, 2023, 8:57 a.m.	Feb. 4, 2023, 9:02 a.m.	314 seconds	none	Show Analyzer Log Show Cuckoo Log

Signatures

- Allocates read-write-execute memory (usually to unpack itself) (50 out of 319 events)
- Changes read-write memory protection to read-execute (probably to avoid detection when setting all RWX flags at the same time) (1 event)
- Potentially malicious URLs were found in the process memory dump (50 out of 493 events)
- Uses Windows utilities for basic Windows functionality (1 event)
- Resumed a suspended thread in a remote process potentially indicative of process injection (2 events)

Fig. 15. Information Catalogue

The events related to the file or URL are disclosed by the signature:



Feedback

Expecting different results? Send us this analysis and we will inspect it. Click here

Information on Execution

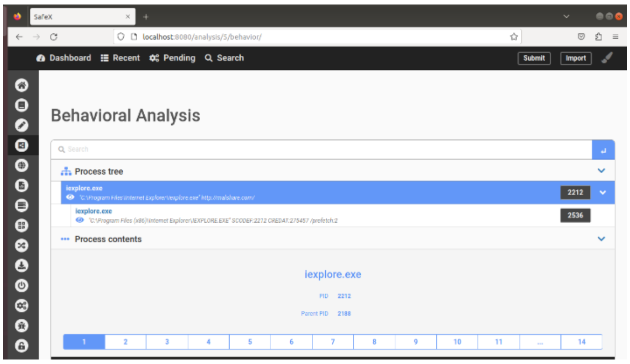
Category	Started	Completed	Duration	Routing	Logs
URL	Feb. 4, 2023, 8:57 a.m.	Feb. 4, 2023, 9:02 a.m.	314 seconds	none	Show Analyzer Log Show Cuckoo Log

Signatures

- Allocates read-write-execute memory (usually to unpack itself) (50 out of 319 events)
- Changes read-write memory protection to read-execute (probably to avoid detection when setting all RWX flags at the same time) (1 event)
- Potentially malicious URLs were found in the process memory dump (50 out of 493 events)
- Uses Windows utilities for basic Windows functionality (1 event)
- Resumed a suspended thread in a remote process potentially indicative of process injection (2 events)

Fig. 16. Events related to Specific File/ URL

The behavioral assessment:



Behavioral Analysis

Process tree

iexplore.exe "C:\Program Files\Internet Explorer\iexplore.exe" http://malshare.com/ 2212

iexplore.exe "C:\Program Files (x86)\Internet Explorer\IEXPLORE.EXE" SCODEF:2212 CREDAT:275457 /prefetch:2 2536

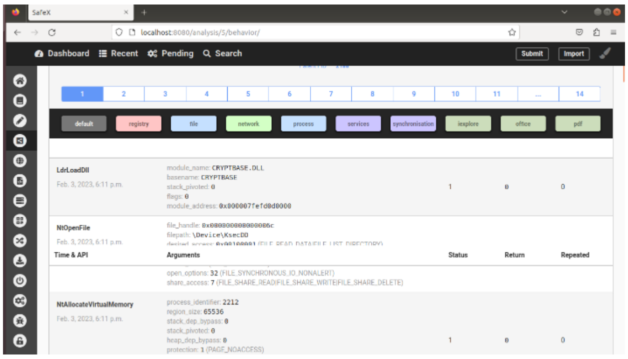
Process contents

iexplore.exe

PID 2212

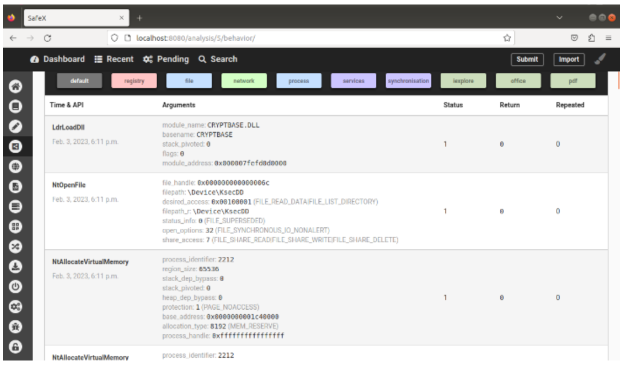
Parent PID 2188

Fig. 17. Behavioral Assessment



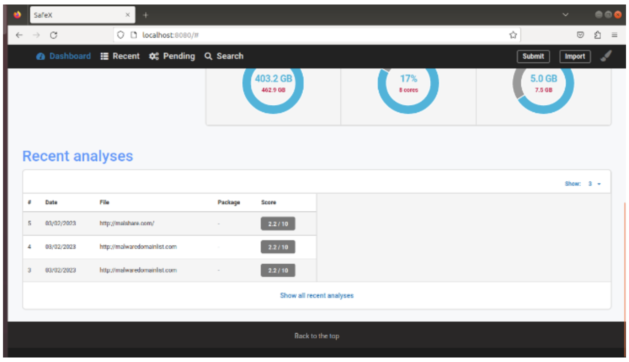
Time & API	Arguments	Status	Return	Repeated
LdrLoadDll Feb. 3, 2023, 6:11 p.m.	module_name: CRYPTBASE.DLL basename: CRYPTBASE stack_pivoted: 0 flags: 0 module_address: 0x000007fefd0d0000	1	0	0
NtOpenFile Feb. 3, 2023, 6:11 p.m.	file_handle: 0x000000000000006c filepath: \Device\KsecDD desired_access: 0x00100001 (FILE_READ_DATA|FILE_LIST_DIRECTORY) open_options: 32 (FILE_SYNCHRONOUS_IO_NONALERT) share_access: 7 (FILE_SHARE_READ|FILE_SHARE_WRITE|FILE_SHARE_DELETE)			
NtAllocateVirtualMemory Feb. 3, 2023, 6:11 p.m.	process_identifier: 2212 region_size: 65536 stack_dep_bypass: 0 stack_pivoted: 0 heap_dep_bypass: 0 protection: 1 (PAGE_NOACCESS)	1	0	0

Fig. 18. Behavioral Assessment -II



Time & API	Arguments	Status	Return	Repeated
LdrLoadDll Feb. 3, 2023, 6:11 p.m.	module_name: CRYPTBASE.DLL basename: CRYPTBASE stack_pivoted: 0 flags: 0 module_address: 0x000007fefd0d0000	1	0	0
NtOpenFile Feb. 3, 2023, 6:11 p.m.	file_handle: 0x000000000000006c filepath: \Device\KsecDD desired_access: 0x00100001 (FILE_READ_DATA|FILE_LIST_DIRECTORY) filepath_r: \Device\KsecDD status_info: 0 (FILE_SUPERSEDED) open_options: 32 (FILE_SYNCHRONOUS_IO_NONALERT) share_access: 7 (FILE_SHARE_READ|FILE_SHARE_WRITE|FILE_SHARE_DELETE)	1	0	0
NtAllocateVirtualMemory Feb. 3, 2023, 6:11 p.m.	process_identifier: 2212 region_size: 65536 stack_dep_bypass: 0 stack_pivoted: 0 heap_dep_bypass: 0 protection: 1 (PAGE_NOACCESS) base_address: 0x0000000001c40000 allocation_type: 8192 (MEM_RESERVE) process_handle: 0xffffffffffffffff	1	0	0
NtAllocateVirtualMemory	process_identifier: 2212			

Fig. 19. Behavioral Assessment - III



Recent analyses

#	Date	File	Package	Score
5	03/02/2023	http://malshare.com/	-	2.2 / 10
4	03/02/2023	http://malwaredomainlist.com	-	2.2 / 10
3	03/02/2023	http://malwaredomainlist.com	-	2.2 / 10

Show all recent analyses

Fig. 20. Recent Analysis

OUTCOME EXPLANATION:

After the investigation is finished, we may check the generated report to look at the malware's actions. The report will provide you with a rundown of the steps the malware made while running in the sandbox.

Follow these procedures to investigate the software's actions: The user uploads a suspicious file for analysis in file submission.

Virtual environment creation: It produces a virtual environment (VM) that is separate from the host system yet similar to a genuine system. The virtual machine is set up to mimic the actions of the target system where the suspicious file was supposed to run.

Execution of the suspicious file: The suspicious file is run in a virtual environment. logging every system-level activity, including system calls, registry modifications, file system changes, and network traffic.

Analysis of the recorded data to ascertain whether the file is maliciously using behavioral analysis. It generates a report of the observed behavior and compares the behavior of the file to recognized malicious patterns.

Report generation: Produces a thorough report of the behavior of the file, including network connections made, system changes made, and any suspicious activity found. Recommendations for mitigation and remediation are also included in the study.

You can learn more about how the virus functions and what it is attempting to do by looking at the actions the malware takes. Using this knowledge, practical methods for identifying and reducing malware threats can be developed.

ENSURE THE SAFETY OF ORGANIZATION DATA:

Implementing our software in an organization can help enhance the organization's security posture by providing a platform to analyze and detect potential malware threats. Here are some steps to implement it in an organization's structure:

Define the use case: Determine the specific use case for your organization. This could include analyzing suspicious files, detecting advanced malware threats, or testing the effectiveness of your security controls.

Define policies and procedures: Deploy the virtualization technology that you plan to use with. This could include VirtualBox, VMware, or KVM. Then, install the software on a dedicated server or a virtual machine. Configure by setting up analysis options, including the analysis environment, analysis packages, and the types of data that you want to capture and analyze. Develop policies and procedures, including guidelines for submitting files for analysis, handling the analysis results, and responding to any detected threats.

Integrate with other security tools: Provide training to your security team on how to use the software effectively. This should include training on submitting files for analysis, interpreting analysis results, and responding to any detected threats. Integrate with your existing security tools, such as antivirus software and SIEM platforms, to enhance your organization's security posture.[17]

Take action: Based on the analysis results, the organization can take appropriate actions to mitigate the impact of the malware. This may involve blocking network traffic to known malicious hosts, isolating infected machines, or cleaning infected systems.

Update security measures: Based on the analysis results, the organization can update its security measures to prevent future malware attacks. This may involve updating antivirus definitions, applying security patches, or implementing additional security controls. Monitor the performance regularly and update the software and configuration as needed to ensure that it continues to operate effectively and efficiently.

Overall, implementing it in an organization can help to improve the organization's security posture by providing a safe and controlled environment for analyzing and detecting malware. An organization's structure requires careful planning, policy development, and training to ensure that it is used effectively and efficiently.

V. CONCLUSION

It is a strong and adaptable automated malware analysis tool that is capable of efficiently identifying and analysing a variety of infections. It is a useful tool for recognising and comprehending the behaviour of malware due to its capacity to offer thorough analysis reports that include network traffic, system calls, and behavioural information. It can offer efficient malware detection when used in conjunction with machine learning methods. The suggested method can be used as a dependable and effective option for locating and reducing malware threats in practical situations.

We tested the performance of the suggested approach in terms of detection accuracy, false positives, and false negatives using a dataset of malware samples.

The findings show that the method is successful in identifying a variety of malware, including zero-day and targeted assaults, and that the addition of machine learning techniques greatly increases detection precision.

Overall, it is a potent open-source tool for dynamic analysis that automates malware analysis and adheres to best practises. It offers a secure setting for running malware samples and observing their behavior, which helps researchers comprehend the potential effects they might have on a target machine. Users can decide on the severity of the malware and take the necessary precautions thanks to its in-depth analysis reports.

VI FUTURE SCOPE

Future potential resides in its ongoing development and enhancement to stay up with the malware landscape's constant change. The following are some prospective directions for study and development:

enhancing the tool's capacity to recognise and counteract evasion strategies utilised by sophisticated malware. improving the tool's compatibility with other platforms, especially mobile ones. Creating more sophisticated behavioural analysis methods to recognise and classify novel malware variants. incorporating machine learning technologies to raise malware detection precision. integrating with other security platforms and solutions to improve an organisation's overall security posture. enhancing the tool's reporting and visualisation features to deliver more useful insights for malware analysis.

Advanced machine learning techniques could be incorporated to increase the precision of malware detection and categorization as new and complex malware continues to proliferate.

Sandboxing in the cloud: Integration with cloud-based services could make it easier for us to scale and offer affordable analysis resources, enhancing the efficiency and precision of malware analysis.

Integration with other security technologies: To offer a more complete security solution, integration with other security technologies such as intrusion detection systems (IDS) and security information and event management (SIEM) systems.

Support for new platforms: Although it now works with a number of different operating systems, adding support for new platforms like mobile operating systems would boost its usefulness and adaptability.

Faster threat response times would be made possible by improved reporting and visualisation capabilities, which would aid analysts in quickly understanding and acting on the findings of their analysis.

Overall, the future's potential is broad and varied, with room for advancement in a number of malware analysis and cybersecurity fields.

ACKNOWLEDGMENTS

We would like to acknowledge Dr. Meenakshi Sundaram B. for guiding us throughout this paper and helping us to come up with a sophisticated model by pushing us to what we can achieve. We are glad we got to work on this paper and make it come to life.

REFERENCES

[1] Daisuke INOUE, Hiroki NOGAWA, Katsunari YOSHIOKA, Koji Nakao, Masashi ETO, and Yuji HOSHIZAWA (2009). *Malware Sandbox Analysis for Secure Observation of Vulnerability Exploitation*

[2] Daisuke Inoue, Katsunari Yoshioka, Koji Nakao, Masashi Eto, Masaya Yamagata, Takahiro Kasama, & Tsutomu Matsumoto (2012). *Malware Sandbox Analysis with Efficient Observation of Herder's Behavior Information and Media Technologies*

[3] F. says: "Evolution of Malware Sandbox Evasion Tactics: A Retrospective Study," McAfee Blogs, 09-Sep-2019.

[4] Juwono, Joshua & Lim, Charles & Erwin, Alva (2015) *A Comparative Study of Behavior Analysis Sandboxes in Malware Detection*

[5] Katsunari YOSHIOKA, & Tsutomu MATSUMOTO. (2010). Multi-Pass Malware Sandbox Analysis with Controlled Internet Connection *IEICE Transactions on Fundamentals of Electronics, Communications, and Computer Sciences*

[6] Le, H., & Ngo, Q. (2020). *V-Sandbox for Dynamic Analysis of an IoT Botnet*

[7] Levy, I. (2020), Virtual Containers Are Replacing Sandboxing As A Technology Option. Infosecurity Magazine.

[8] "Norman Sandbox: Your Proactive IT Security Tool." *Norman Sandbox*, 2010, download01.norman.no/product_sheets/eng/SandBox_analyzer.pdf.

[9] Sikorski, M., & Honig, A. (2011). *Practical malware analysis* No Starch Press

[10] Xin Jiang, Mingzhe Liu, Kun Yang, Yanhua Liu, and Ruili Wang (2018). *A Security Sandbox Approach of Android Based on Hook Mechanism*

[11] Zoltan Balazs (2016) *Malware Analysis Sandbox Testing Methodology Le Journal de La Cybercriminalité & Des Investigations Numériques*

[12] A Survey of Malware Analysis Techniques and Tools" by Mihai Lazarescu (2013)

[13] Dynamic Malware Analysis: A Case Study" by A. Cichocki and R. M. Nowak (2012)

[14] Automated Malware Analysis and Classification" by Konstantinos Demertzis and Dimitrios Geneiatakis (2015)

[15] "A survey of dynamic analysis techniques for malware detection and analysis" by Dinesh Goyal, Hiren Kumar Deva Sarma, and Venkata Narasimha Inukollu (2015)

[16] A survey of static analysis techniques for malware detection and analysis" by Dinesh Goyal, Hiren Kumar Deva Sarma, and Venkata Narasimha Inukollu (2015)

[17] Malware detection using sandbox technology" by Minaxi Singh and Bhushan Trivedi (2016)