Netradyne Standard Operating Procedure Malware Analysis

v1.0

Internal and Confidential

TABLE OF CONTENTS

Document Control

Document ID	NDMA2022001
Document Name	Netradyne Standard Operating Procedure Malware Analysis
Document Status	Draft
Document Released Date	04/May/2022
Document Author	Netradyne Infosec Team
Document Content Contributors	Prathamesh Padoskar
Document Signatory	Saravanan Sankaran
Document Owner	Saravanan Sankaran
Document Version	V1.0
Information Classification	Internal

Document Edit History

Version	Date	Additions/Modifications	Prepared/Revised By
0.1	28/Apr/2022	Draft version	Prathamesh Padoskar
0.2	04/May/2022	Changes incorporated post initial review	Prathamesh Padoskar
1.0	25/May/2022	Initial Release	Infosec

Document Review/Approval

Date	Signatory Name	Organization/Signatory Title	Comments
25/May/2022	Saravanan Sankaran	Senior Director, Infosec & IT	

Distribution of Final Document

Name	Organization/Title

1 Overview

One of the most widespread threats to cyber security in today's world of unlimited Internet access is malware. In recent times, the malware being designed are with the ability to transform their code and to hide quietly in the systems of the unsuspecting users. Malware analysis is the process of performing analysis of the malware and understanding its actions and behaviour. It is of two types- static and dynamic analysis. Static analysis is performed by observing the source code of the malware and drawing conclusions based on it. Dynamic analysis is the analysis performed by executing the piece of code and noting its actions. Malware analysis is an important and relevant task, for the advanced forms of malware these days are often not even detectable by commonly available anti-virus software.

2 Scope

Test Malware and check for relevant IOC create by malware and put appropriate actions on it. Sandbox environment is created; malware can be tested on this environment. Based on analysis of malware we will get IOC, we can use those IOC and submit on various security tools to prevent it from happening again. You can use various tools through Sandbox machine to test malware.

3 Pre-requisites / Assumptions

Only authorized users have access to sandbox machine. Members are listed below.
Analysis is strictly performed on sandbox machines as they are out of our network.

No.	Name	Access Level
1	Gautam Kumar	Admin Access
2		Admin Access

4 Steps to access sandbox machine

Connect to VPN of vpn-india.netradyne.com.
Once connected to VPN please access terminal server 172.16.20.126 with your credentials.
Through terminal server please access sandbox machine 10.11.13.100 with your credentials.

5 Malware Analysis Important Terms

Malware analysis is the process of understanding the behavior and purpose of a suspicious file or URL. The output of the analysis aids in the detection and mitigation of the potential threat.
Before you get into what malware analysis is, you first need to understand what malware is. Malware or malicious software is a program or code that is designed to enable unauthorized access to a computer or network. Malware can come in many forms.

5.1 Types of Malwares

• **Viruses**: A self-replicating malicious program that attaches itself to legitimate files. In some cases, viruses do not damage the victim's computer, but is solely created for humor or to disrupt productivity.

• **Ransomware**: Malicious software created to encrypt the victim's data, preventing the victim from being able to access their files in exchange for money.

• **Rootkit**: A malicious software that allows threat actors to remotely connect to a compromised computer and control it without the victim's knowledge.

• **Trojan**: Malware that is hidden inside seemingly legitimate software and can cause damage to the victim's computer and/or steal data.

• **Worm**: A type of malware that is designed to self-replicate and spread to other computers consuming most of or all of the bandwidth on a victim's network.

• **Spyware**: A form of malware that covertly runs in the background of a victim's device collecting their data.

5.2 Types of Malware Analysis

- **Statical or Static property analysis:** This method involves analyzing the binary without executing it. Static analysis allows the analyst to extract metadata from the binary. This technique reveals limited but useful information about the binary that will aid in further analysis. Static analysis also involves analyzing the malicious code itself. The analyst will disassemble and/or debug the suspect binary and investigate the code to get a better understanding of the program's functionality.

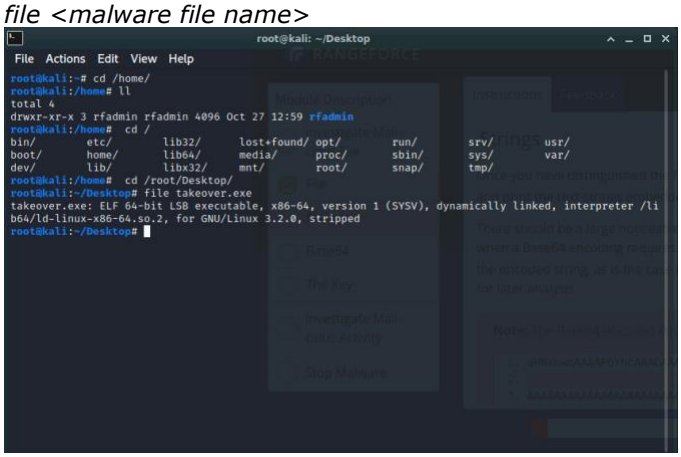
- **Dynamic or fully automated malware analysis:** Dynamic or behavioral analysis involves executing the binary to get a visual understanding of how the binary functions. It is highly recommended that you perform dynamic analysis in an isolated environment. This technique is useful but does not uncover all the functionalities of the malicious binary.

- **Manual Code reversing:** In this type involves reverse engineering of the malicious file. this process is made especially difficult because you must circumvent everything the malware does to prevent you from understanding it. code packing, code obfuscation, anti-debugging techniques. Reverse engineering is time consuming and complex. but it is the only way to have complete insight into piece of malware.

6 Basic tools and steps for malware analysis

6.1 using statical approach

In this scenario we have takeover.exe malware file and we will be performing analysis on this file.
1) Investigate malware file type: Determining if the binary has a file type of portable executable (PE) will help determine which operating system the malware is targeting. It is not recommended to rely on file extensions alone. We will be using file command to find file type.
file <malware file name>



Can conclude An ELF 64-bit LSB executable and linux executable.

2) Strings
Once you have distinguished the file type, you can go ahead and use strings to find and print the text strings embedded in the file.
There should be a large noticeable Base64 string embedded in the file. Sometimes, when a Base64 encoding requires padding, it adds one or two = characters to the end of the encoded string, as is the case in this file. Copy that whole string and save it into a file for later analysis.

Note: The Base64-encoded string's first and last lines look like this:

```
1. aHNxcwcAAAAP6YhcAAACAAAAAAAEABEA0AIBAAQAAADgAAAAAAAAK0FAAAAAAAApQUAAAAAAD/
2. ...
3. AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA==
```

Often, SHA-256 or other hashing algorithms are used to create a unique signature for a malware file. The sha256sum utility can be used for that.

command to pull strings: strings <file>
Command to generate hash: sha256sum <file>

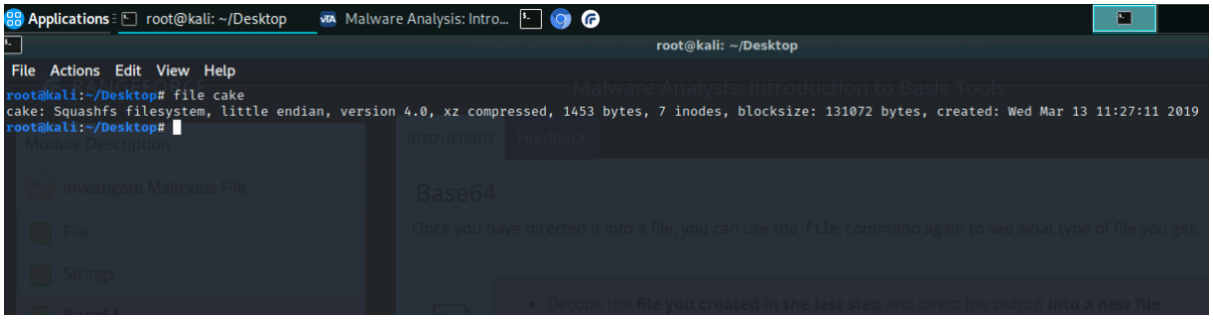
3) Base64: Now that you have created a new file with the Base64 encoding saved into it, it is time to decode it and see what it entails. The base64 utility is handy for these kinds of situations.

command: base64 -d <file>

If you just try to decode it, you get a bunch of gibberish. This could indicate that you are dealing with binary. Try directing the output into a file.

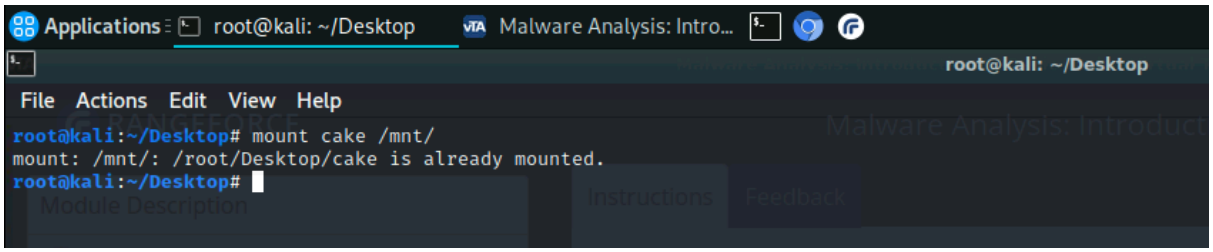
command: base64 -d <file> > <new file>

Once you have directed it into a file, you can use the file command again to see what type of file you get.

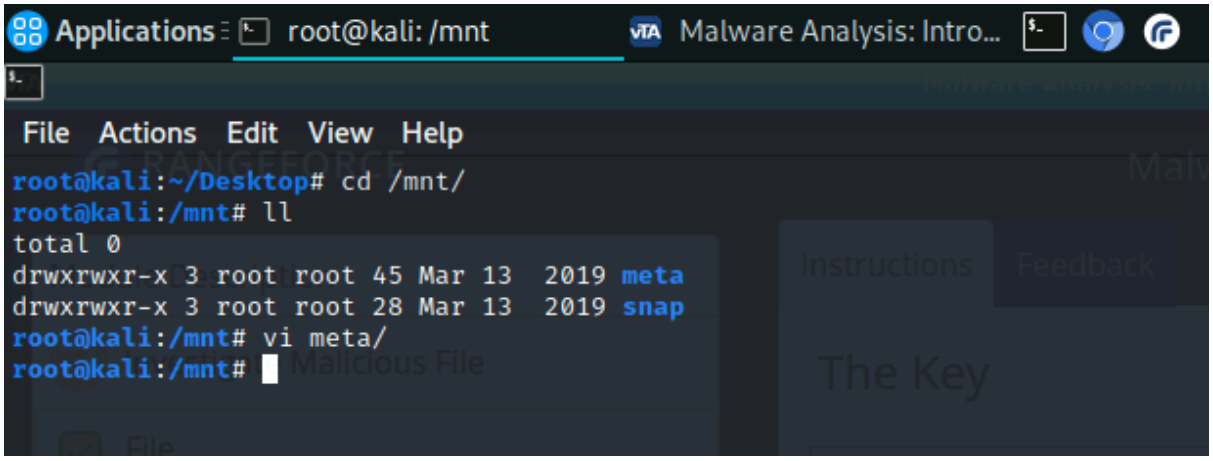


4) The Key: It seems that there is a compressed read-only file system encased in the file. You can

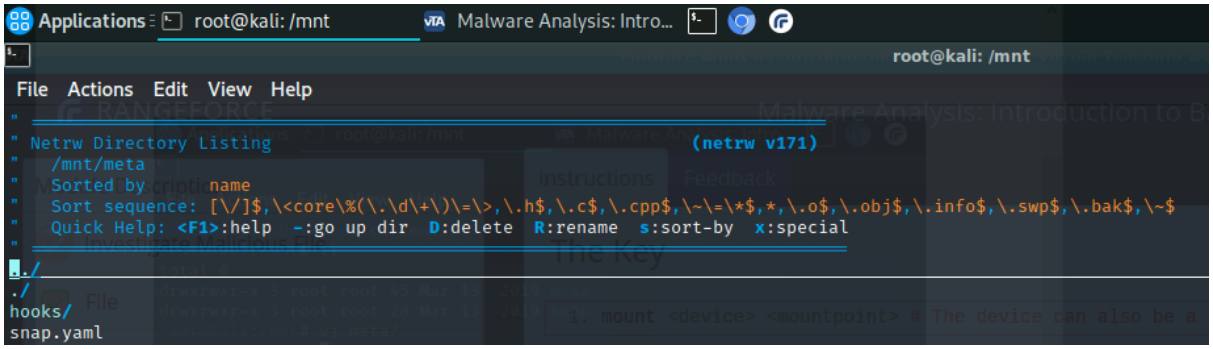
easily mount it to the OS file system using mount. The /mnt directory is usually used as a mountpoint.
mount <device> <mountpoint> # The device can also be a file
Now you have a bunch of new files to investigate. Try finding the key from the file content.

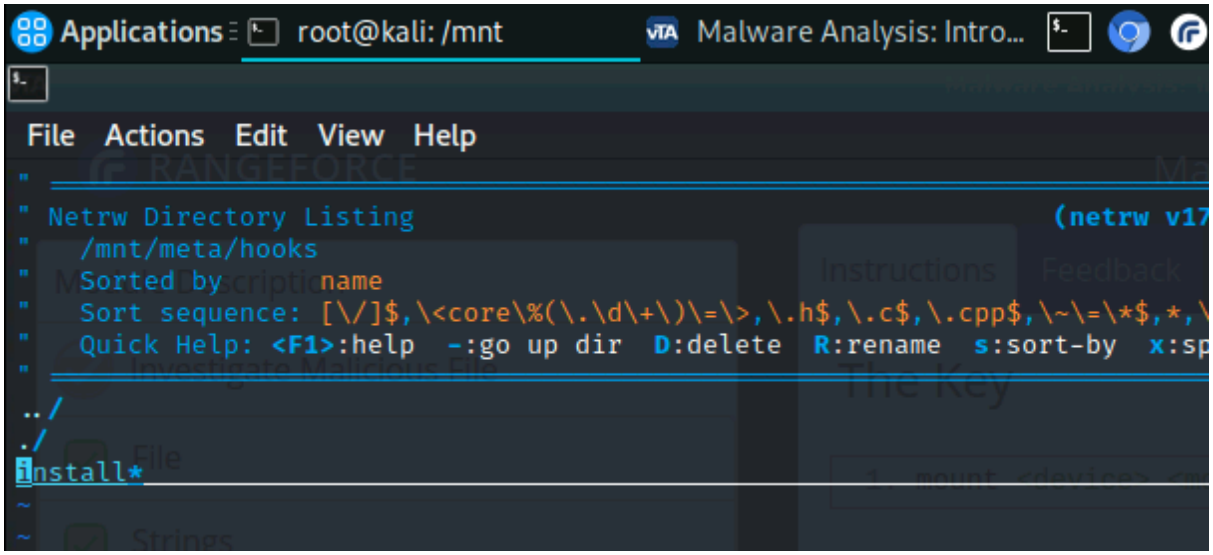


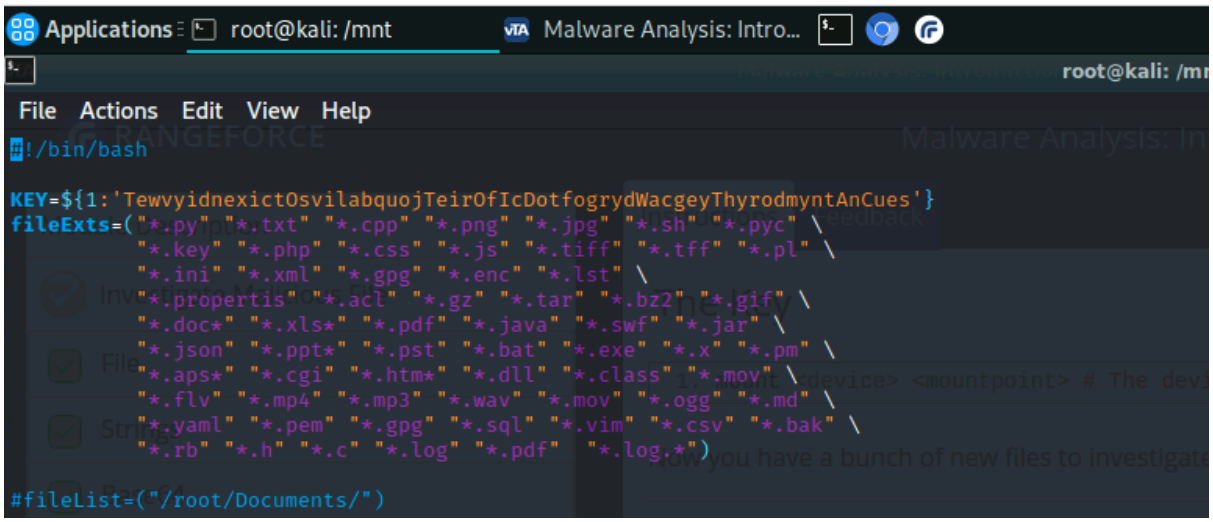
now go to /mnt the open meta.



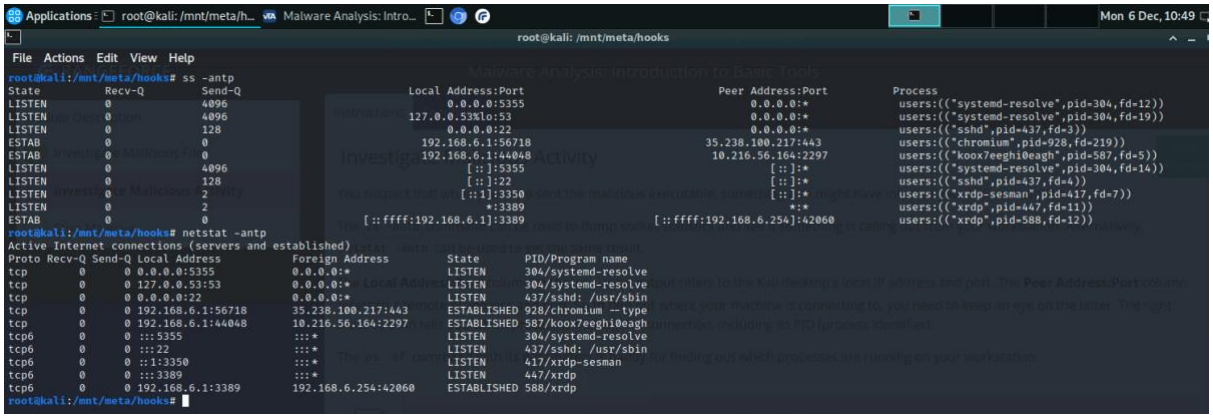
You can investigate in both files hooks and snap.yml. I found key in hooks / install*.



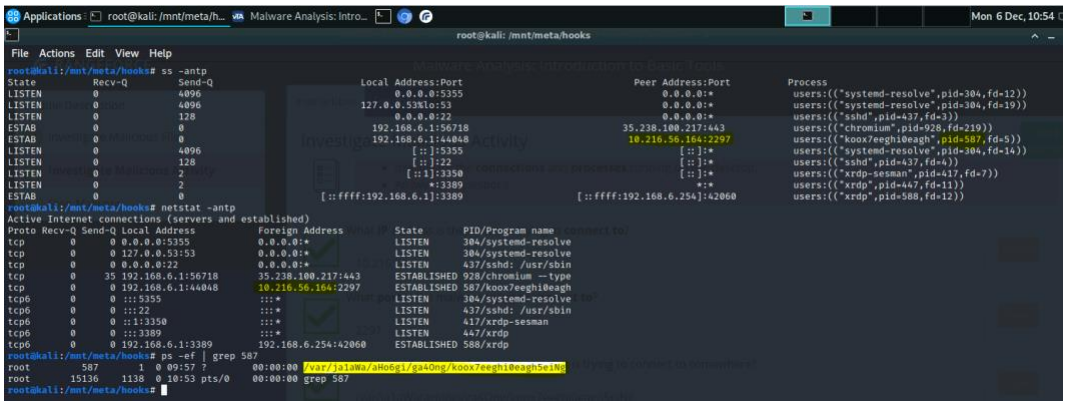




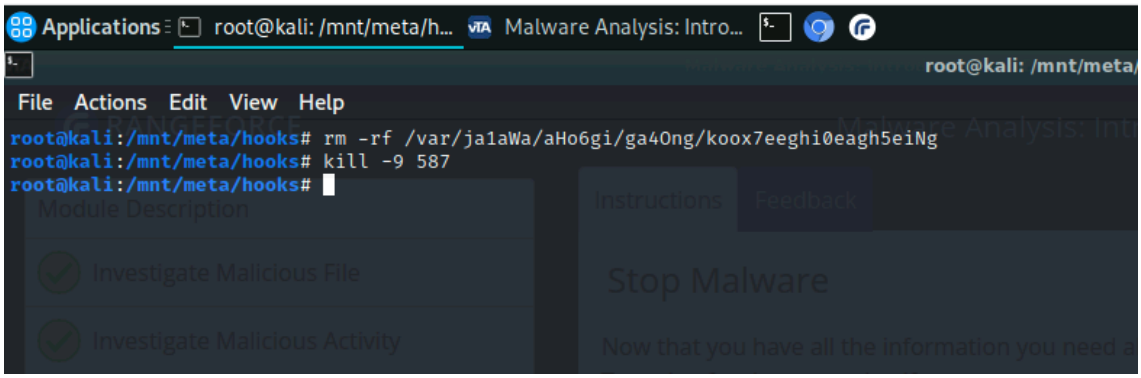
5)Investigate Malicious Activity: You suspect that when you were sent the malicious executable, something else might have infected your Kali workstation.
The ss -antp command can be used to dump socket statistics and see if something is calling out from your workstation. Alternatively, netstat -antp can be used to get the same result.
The Local Address:Port column in the ss -antp output refers to the Kali desktop's local IP address and port. The Peer Address:Port column refers to a remote IP address and port. So to find out where your machine is connecting to, you need to keep an eye on the latter. The right-most column tells you which process is making the connection, including its PID (process identifier).
The ps -ef command with its parameters is handy for finding out which processes are running on your workstation.



Now you can find for which Ip malware is trying to connect then port and full path of malware.



6) Stop Malware: Now that you have all the information you need about the malware, it's time to stop its activities. Just killing the process is not always enough. Try going for the source itself.
Command
To remove a file
rm <file>
To kill a process
kill -9 <pid>



6.2 using Dynamic approach

Dynamic or behavioral analysis involves executing a malware sample in a virtual environment and analyzing how it affects the target system. There are multiple tasks that need to be done during analysis.

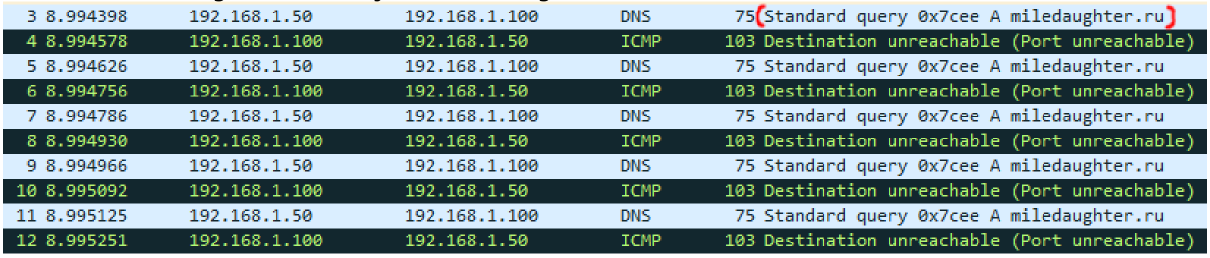
Task/Tools

- Simulating internet services: Using Flare-Fakenet-NG or INetSim to provide the malware a simulation of the services it requires to operate.

```
=== INetSim main process started (PID 21034) ===
Session ID:     21034
Listening on:   127.0.0.1
Real Date/Time: 2021-01-26 19:29:57
Fake Date/Time: 2021-01-26 19:29:57 (Delta: 0 seconds)
 Forking services...
  * dns_53_tcp_udp - started (PID 21036)
  * finger_79_tcp - started (PID 21048)
  * daytime_13_tcp - started (PID 21053)
  * ntp_123_udp - started (PID 21047)
  * syslog_514_udp - started (PID 21050)
  * time_37_udp - started (PID 21052)
  * daytime_13_udp - started (PID 21054)
  * quotd_17_tcp - started (PID 21059)
  * quotd_17_udp - started (PID 21060)
  * chargen_19_tcp - started (PID 21061)
  * chargen_19_udp - started (PID 21062)
  * irc_6667_tcp - started (PID 21046)
  * dummy_1_udp - started (PID 21064)
  * discard_9_tcp - started (PID 21057)
  * discard_9_udp - started (PID 21058)
  * echo_7_tcp - started (PID 21055)
  * dummy_1_tcp - started (PID 21063)
  * echo_7_udp - started (PID 21056)
  * tftp_69_udp - started (PID 21045)
  * http_80_tcp - started (PID 21037)
  * pop3s_995_tcp - started (PID 21042)
  * ftps_990_tcp - started (PID 21044)
  * https_443_tcp - started (PID 21038)
  * ftp_21_tcp - started (PID 21043)
  * pop3_110_tcp - started (PID 21041)
  * ident_113_tcp - started (PID 21049)
  * time_37_tcp - started (PID 21051)
  * smtps_465_tcp - started (PID 21040)
  * smtp_25_tcp - started (PID 21039)
 done.
Simulation running.
```

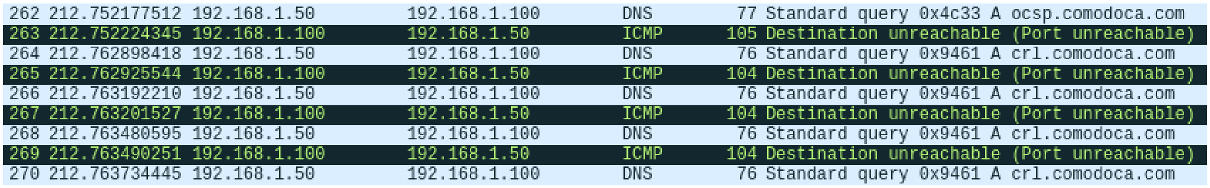
Monitoring the malware to understand how it behaves and its effects on the system is important during dynamic analysis. There are multiple forms of monitoring malware activity, which include network monitoring, process monitoring, registry monitoring, and file system monitoring.

- **Network monitoring**: Monitoring live network traffic to and from the system during malware execution. (Wireshark, Flare-Fakenet-NG, INetSim)
  In this below image we see trojan connecting to C2 server.



3	8.994398	192.168.1.50	192.168.1.100	DNS	75 Standard query 0x7cee A miledaughter.ru
4	8.994578	192.168.1.100	192.168.1.50	ICMP	103 Destination unreachable (Port unreachable)
5	8.994626	192.168.1.50	192.168.1.100	DNS	75 Standard query 0x7cee A miledaughter.ru
6	8.994756	192.168.1.100	192.168.1.50	ICMP	103 Destination unreachable (Port unreachable)
7	8.994786	192.168.1.50	192.168.1.100	DNS	75 Standard query 0x7cee A miledaughter.ru
8	8.994930	192.168.1.100	192.168.1.50	ICMP	103 Destination unreachable (Port unreachable)
9	8.994966	192.168.1.50	192.168.1.100	DNS	75 Standard query 0x7cee A miledaughter.ru
10	8.995092	192.168.1.100	192.168.1.50	ICMP	103 Destination unreachable (Port unreachable)
11	8.995125	192.168.1.50	192.168.1.100	DNS	75 Standard query 0x7cee A miledaughter.ru
12	8.995251	192.168.1.100	192.168.1.50	ICMP	103 Destination unreachable (Port unreachable)

  In below image trojan is attempting to connect suspicious urls



262	212.752177512	192.168.1.50	192.168.1.100	DNS	77 Standard query 0x4c33 A ocsp.comodoca.com
263	212.752224345	192.168.1.100	192.168.1.50	ICMP	105 Destination unreachable (Port unreachable)
264	212.762898418	192.168.1.50	192.168.1.100	DNS	76 Standard query 0x9461 A crl.comodoca.com
265	212.762925544	192.168.1.100	192.168.1.50	ICMP	104 Destination unreachable (Port unreachable)
266	212.763192210	192.168.1.50	192.168.1.100	DNS	76 Standard query 0x9461 A crl.comodoca.com
267	212.763201527	192.168.1.100	192.168.1.50	ICMP	104 Destination unreachable (Port unreachable)
268	212.763480595	192.168.1.50	192.168.1.100	DNS	76 Standard query 0x9461 A crl.comodoca.com
269	212.763490251	192.168.1.100	192.168.1.50	ICMP	104 Destination unreachable (Port unreachable)
270	212.763734445	192.168.1.50	192.168.1.100	DNS	76 Standard query 0x9461 A crl.comodoca.com

  Upon further investigation of the C2 server (miledaughter.ru) and the suspicious URLs (ocsp.comodoca.com, crl.comodoca.com) a few sources (howtoremove.guide, how-to-remove.com) claim that this trojan happens to be a browser hijacker that is delivered upon the installation of seemingly legitimate software. Although when visiting the comodoca.com, the site offers SSL certificates. As for the C2 server, after running a whois query it shows that the domain registrar is 101DOMAIN-RU and it was created May 3, 2020. ThreatCrowd also displays a graph showing the different variants of this malware and their origins.

- **Process Monitoring:** Examining the malware process activity during malware execution. (Process Hacker, Process Monitor, Noriben)

- **Registry monitoring:** Monitors whether registry keys have been modified/accessed and what registry data has been read/written to. (Noriben, Process Monitor)

- **File system monitoring:** Monitoring the file system activity in real-time during malware execution. (Noriben, Process Monitor)

  Below is Noriben's report after executing the malware. As you can see, there are multiple registry changes, but the most intriguing registry change is "[RegSetValue]

  Explorer.EXE:5436 > HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\ActivityDataModel\ReaderRevisionInfo\6540C835-B904-5629-60B1-FAD36E0062D1 = 1¬ 笁†腏瑯扡獨鑲獵慴据鑲≤摸勖ㄴ+†囫煥敵据≥摸烈䶂ꓤ†∠捡梯楶裓瑓㮡鑲≤摸∠靥/ᆢ帨匭篼卄Ⅹ坃恣坳箣⊖秾ᡫ菮〰㈶ꓭQ†咄汩整≲摸笠††∠獩俶摡樓瑬枋•›+††∠牯杩湩樓瑬枋敘≵摸ꓽ††† 瑚愕整樓瑬枋敘≵摸ꓽ††† 産敳稰瑦濮卮慍整樓瑬枋•›ᖷ†素紊;.

Running the registry value in Google Translate reveals that the text is Chinese. The program was able to translate a portion of the text to renewing the enemy's data, picking up the stubborn data, and production and treatment. There were no interesting data entries logged for under the Network Traffic and Unique Host tabs since the malware was connecting to a simulated network. You can view the full list of changed registries here.

```
-=] Sandbox Analysis Report generated by Noriben v1.8.4
-=] Developed by Brian Baskin: brian @@ thebaskins.com  @bbaskin
-=] The latest release can be found at https://github.com/Rurik/Noriben

-=] Execution time: 47.90 seconds
-=] Processing time: 0.48 seconds
-=] Analysis time: 0.69 seconds

Processes Created:
==================
[CreateProcess] Explorer.EXE:5436 > "%UserProfile%\Desktop\salford1\salford1.exe "	[Child PID: 2632]
[CreateProcess] svchost.exe:812 > "%WinDir%\SysWOW64\DllHost.exe /Processid:{776DBC8D-7347-478C-8D71-791E12EF49D8}"	[Child PID: 2496]

File Activity:
==================
[CreateFile] svchost.exe:1648 > %WinDir%\ServiceProfiles\LocalService\AppData\Local\FontCache	[File no longer exists]
[CreateFile] svchost.exe:1648 > %WinDir%\ServiceProfiles\LocalService\AppData\Local\FontCache	[File no longer exists]
[RenameFile] svchost.exe:1648 > %WinDir%\ServiceProfiles\LocalService\AppData\Local\FontCache\FontCache-S-1-5-18.dat => %WinDir%\ServiceProfiles\LocalService\AppData\Local\FontCache\~FontCache-S-1-5-18.dat
[RenameFile] svchost.exe:1648 > %WinDir%\ServiceProfiles\LocalService\AppData\Local\FontCache\FontCache-FontSet-S-1-5-18.dat => %WinDir%\ServiceProfiles\LocalService\AppData\Local\FontCache\~FontCache-FontSet-S-1-5-18.dat
[CreateFile] svchost.exe:1648 > %WinDir%\ServiceProfiles\LocalService\AppData\Local\FontCache\~FontCache-FontSet-S-1-5-18.dat	[File no longer exists]
[CreateFile] svchost.exe:1648 > %WinDir%\ServiceProfiles\LocalService\AppData\Local\FontCache\~FontCache-S-1-5-18.dat	[File no longer exists]

Registry Activity:
==================
[RegSetValue] System:4 > HKLM\System\CurrentControlSet\Services\bam\State\UserSettings\S-1-5-21-4279892692-2277359461-354179757-1001\SequenceNumber  =  44
[RegSetValue] Explorer.EXE:5436 > HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\ActivityDataModel\ReaderRevisionInfo\6540C835-B904-5629-60B1-FAD36E0062D1  =  1ㄱ	笀†䐀瑞扡犸鑲獮
慴据鑲≤摸勮ㄴ+†匢煥敵据≥摸烈㱏ꓲ†∠捡㭁䴢䜀榌婥籴鑲≤摸∠靥/ㄟ帗㴀䈝⺡Ⅹ坫䫸
```

坳箣⊖秾⺄禔〰㈶ㅂㅅǬ†咄汨整≲摸笠††∠獵僘慨樓鎏牞•›+††∠牯杩湩樓鎏牞敘≵摸
〠ꓤ†††猢慴整樓鎏牞敘≵摸〠ꓤ†††産敳稽瑣潩卮慴整樓鎏牞•›ᑯ†素紊;

27. [RegSetValue] Explorer.EXE:5436 > HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\ActivityDataModel\ReaderRevisionInfo\6540C835-B904-5629-60B1-FAD36E0062D1 = 1ㄱ 笁†膠瑯扡獨鐮獺
慴据鐮≤摸勖ㄴ+†匢熸敵据≥摸烈䝹ꓤ†∠捡椭湥祷瑯牯鐮≤摸∠䝹 /ᐟ帵區篼ㅐⵝ垳瑟
坳箣⊖秾⺄禔〰㈶ㅂㅅǬ†咄汨整≲摸笠††∠獵僘慨樓鎏牞•›+††∠牯杩湩樓鎏牞敘≵摸
〠ꓤ†††猢慴整樓鎏牞敘≵摸〠ꓤ†††産敳稽瑣潩卮慴整樓鎏牞•›ᑯ†素紊;
28. [RegSetValue] svchost.exe:812 > HKLM\System\CurrentControlSet\Services\bam\State\UserSettings\S-1-5-21-4279892692-2277359461-354179757-1001\\Device\HarddiskVolume2\Windows\System32\dllhost.exe = D1 18 56 1F F4 ED D6 01 00 00 00 00 00 00 00 00
29. [RegSetValue] ctfmon.exe:5312 > HKCU\SOFTWARE\Microsoft\Input\TypingInsights\Insights = 02 00 00 00 07 1D E8 C1 31 CC 83 60 A3 D6 D9 C1
30. ...
31.
32.
33. Network Traffic:
34. =================
35. [UDP] svchost.exe:2388 > 192.168.1.100:53
36. [UDP] svchost.exe:2388 > 169.254.38.149:53
37. [UDP] svchost.exe:2388 > ff02:0:0:0:0:0:1:3:5355
38. [UDP] System:4 > 192.168.1.100:137
39. [UDP] System:4 > 169.254.38.149:137
40. [UDP] 192.168.1.100:53780 > svchost.exe:7048
41.
42. Unique Hosts:
43. =================
44. 169.254.38.149
45. 192.168.1.100
46. ff02

7 Malware Sandboxing using Hybrid Analysis

Malware sandboxing is a technique utilized by cybersecurity professionals to analyze malicious and potentially dangerous software. The idea behind it is rather simple — create a secure "walled" environment (a virtual machine, which in this context is called a sandbox) and let the malware do whatever it was supposed to do. This way defenders can collect indicators of compromise (or IOCs for short) without the risk of infecting their systems.
here are many great automated sandbox solutions that allow you to submit your malware samples for analysis online. With the help of tools like Hybrid Analysis, Cuckoo and AnyRun, you can submit malicious files and URLs for analysis for free and get results within minutes.

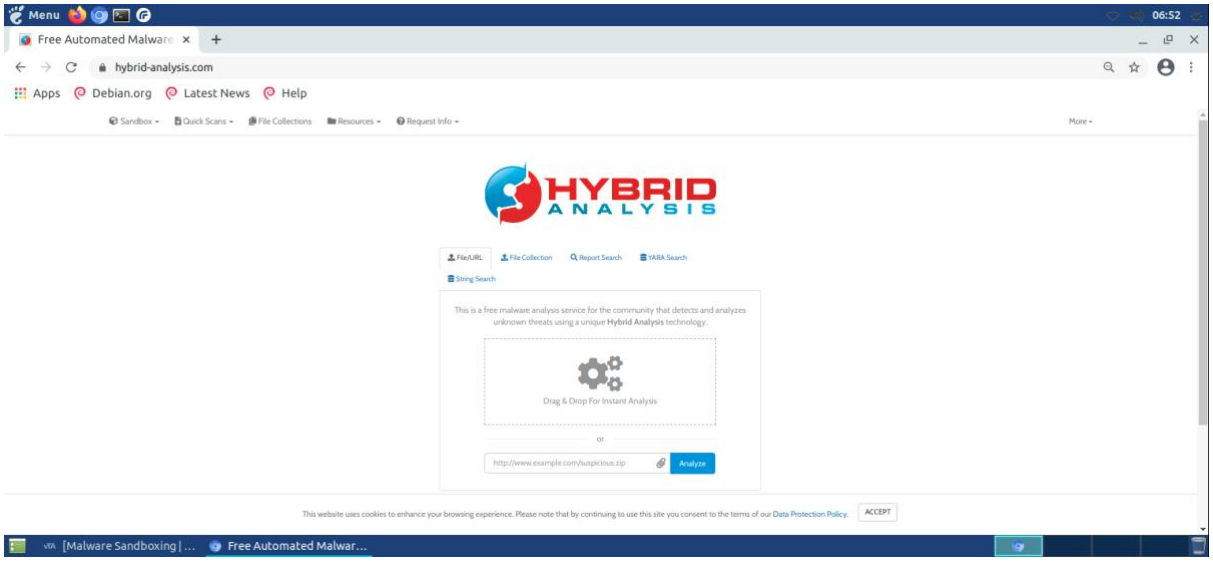
7.1 Submitting Malware for Analysis

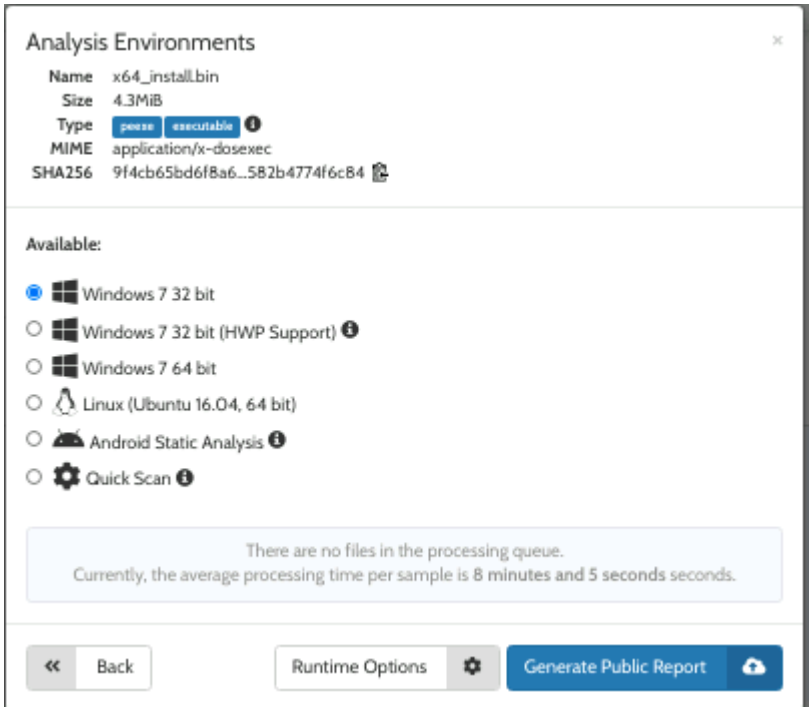
There are many sandbox solutions, both commercial and free. We will be using hybrid analysis for sandboxing. Hybrid Analysis allows you to analyze both files and URLs for malicious activity. It cross-references dozens of security intelligence providers to gather information on potential IOCs and related flavors of malware.

Steps for submitting malware

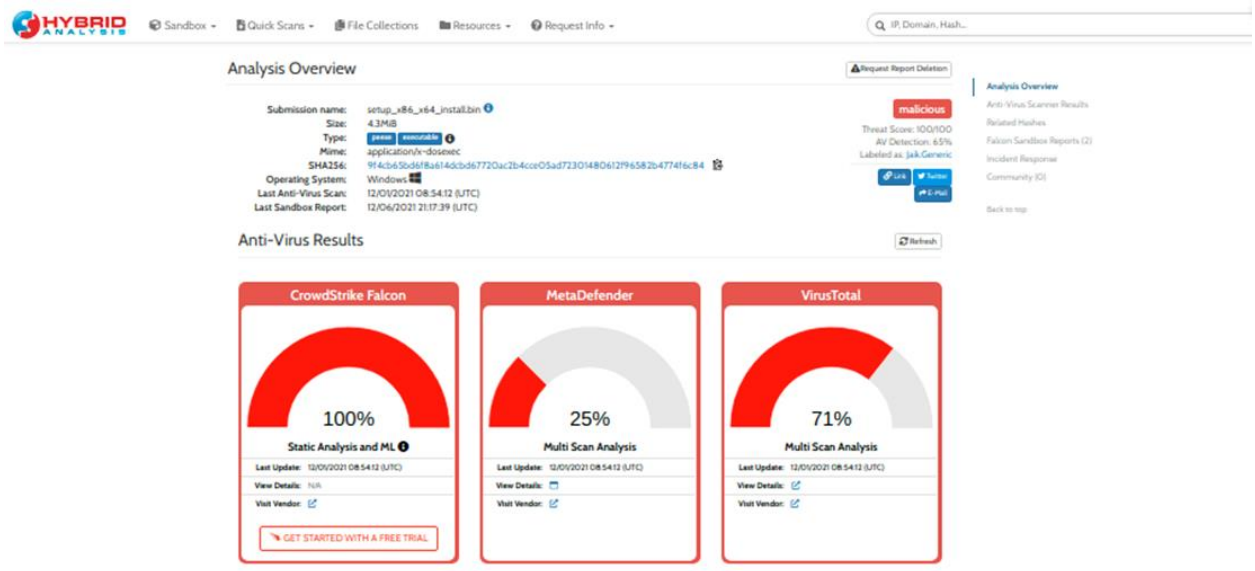
- Open Hybrid-analysis.com in your browser.
- Drag and drop malware sample
- Complete captcha and select analysis environment on which you want to test.
- Click to generate public report.

After submitting a file for analysis, the tool automatically redirects you to the Analysis Overview page. This is where you can find all the base information regarding the file, including:
file type,
SHA256 hash,
antivirus detection score,
general assessment of whether the file is malicious or not.





Results will appear as follows.

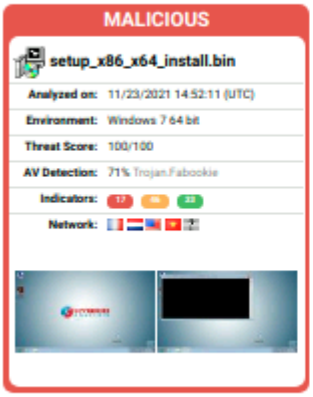


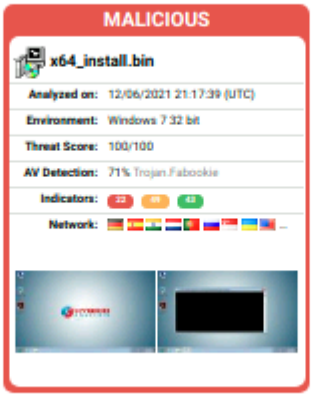
Related Hashes

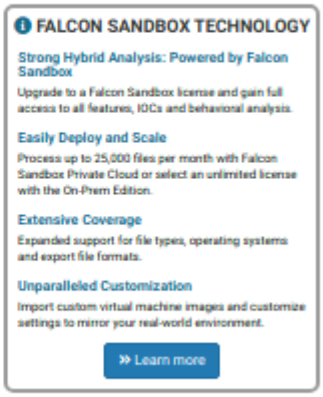
Files extracted during detonation

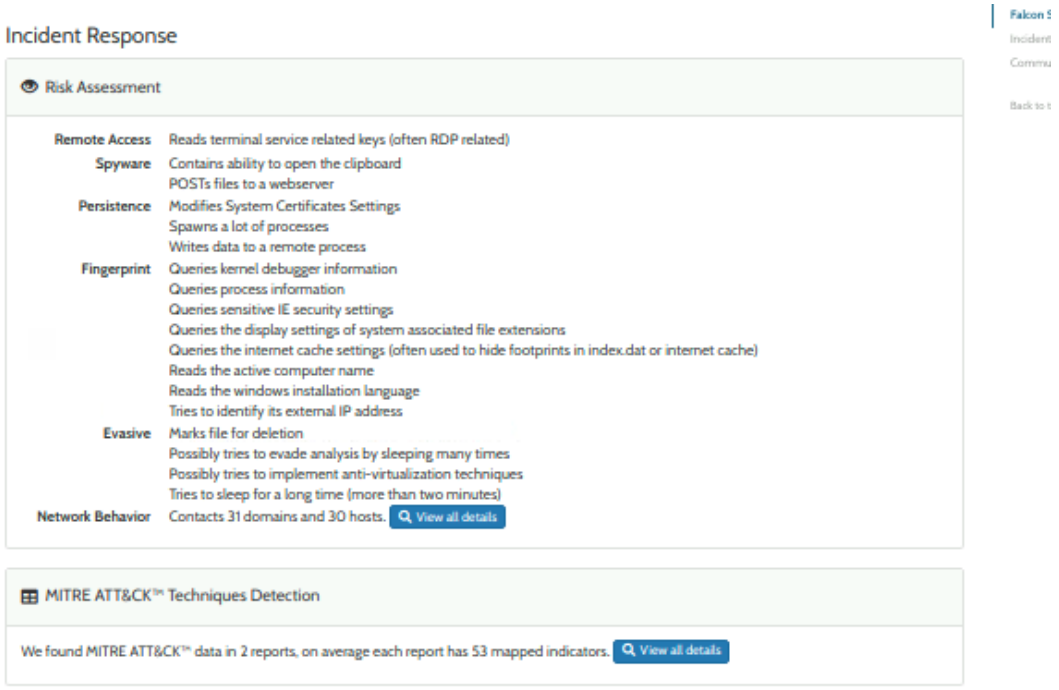
Name	Sha256	Verdict
file	ea8cce26f5348ef3395c7b4a713b28a7801cfc1a27b67bb860b82063c4276a1d	malicious
file	b5b2c802acacf54a31c2ad67b0d971d481db8887a939173b54ec2a933792daa9	malicious
file	339e4df29303e36ef5f8c9ef7381f9007a8b01b3cb3bf5ffcee56495c4e40747	malicious
file	d5959e437e58709c5e5e7a923efe7351b28bedef15cb00cd9fdb4e5e955b2af3	malicious
file	83664d9745f1f75b770b960a253e5efc0ff4ee06b72083fa8be2bbf801328d3e	malicious
SunO6964bbec839856df.exe	82c8f6980fe9b0de916fc1954a2e1db5101177001794f8fdf15a2e84656962e6b7	malicious
file	5da618d0d54f9251a1735057b27f9a5f88e2ddd44f53ce35ce69caaf678f26a8	malicious
file	93af7aa366e45b5d4c87a6273cd20a6657a729516831f486ca9478dffd44f83b	malicious
file	d838cfaf7bf97d6c3379e2c5daf269cc422a09df556de6ca08fef74b4906b3b6	malicious
file	282eb8e7745f9396a255f817e90afbdfe54a77c427c3050fd0ec638fb2f50dc3	malicious
file	6e14451c4001c83136bbe8d08d404b038aedb2a7dffa18cf45581b8cc0d78ccd	malicious
file	36cc42294d2cac9e45fa389f9a7afdf18cb5af6f68ed2d5e9563bd522f48bc4a	malicious
file	d5ccebea40a76ec2c82cac45cc208a778269e743f1a825ef881533b85d6c1d31	malicious
Udp_1_.exe	2975e8edd2d526652fd56cddf2e8fe293c341d6a08232fc0e4974a8678cf6c6b	malicious
file	af620f48d534f6db07e31fa18182cbf78bf4b9c9128657a779094cdbd8fe4a25	malicious
_shfoldr.dll	9884e9dfb4f8a873ccbd81f8ad0ae257776d2348d027d811a56475e02836d087	whitelisted
foradvertisingwwb_1_.exe	342038018ad6203f807f31a0ccafeaff69a4db81d297bf1d4b8e4c963f67ea50	malicious

Falcon Sandbox Reports



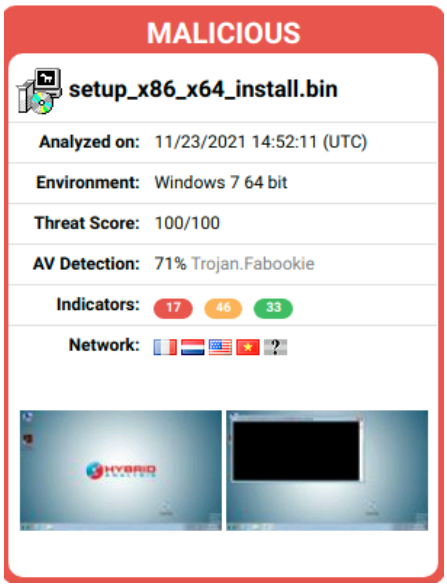






7.2 Examining the result

Once the sandbox analysis results appear in the Falcon Sandbox Reports section (located right below the antivirus section of the page), you will see a generated report, like so:

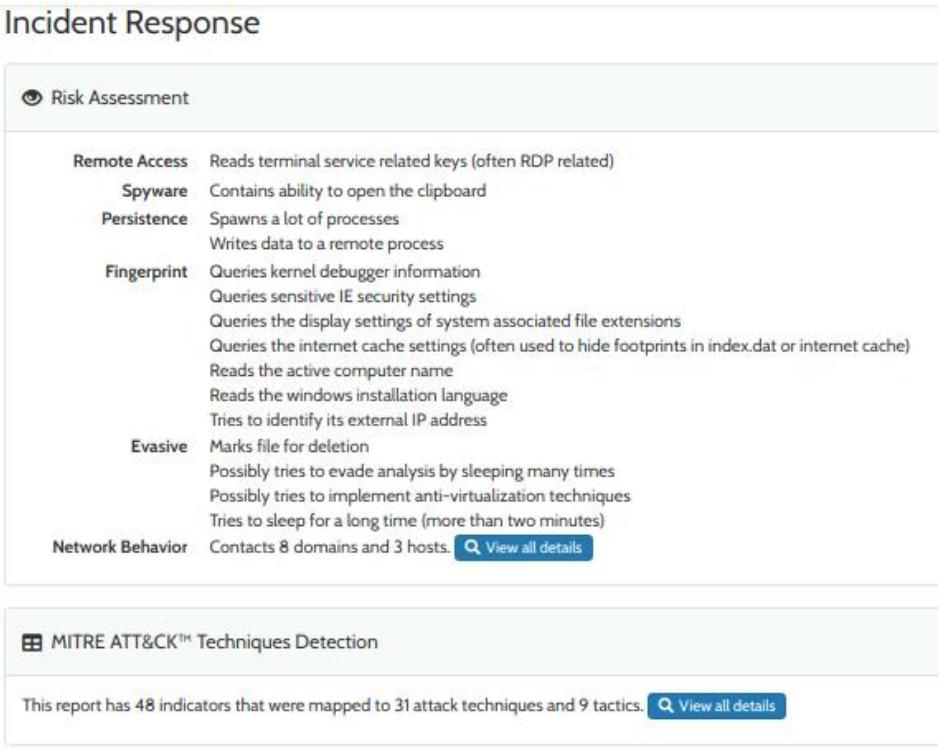


1) Incident Response

The first section of the report is titled Incident Response and consists of two parts:

- Risk Assessment that gives you the summary of what the malware is doing on the infected machine;
- MITRE ATT&CK mapping of the techniques used by the malicious file (you can click the View all details button to see the full mapping).

This section alone is enough to give you an idea of the TTPs used by the malware. Further sections of the report, however, are very useful for gathering more specific information, such as pinpointing the attacker's C2 servers and process/file names created and modified by the malware.

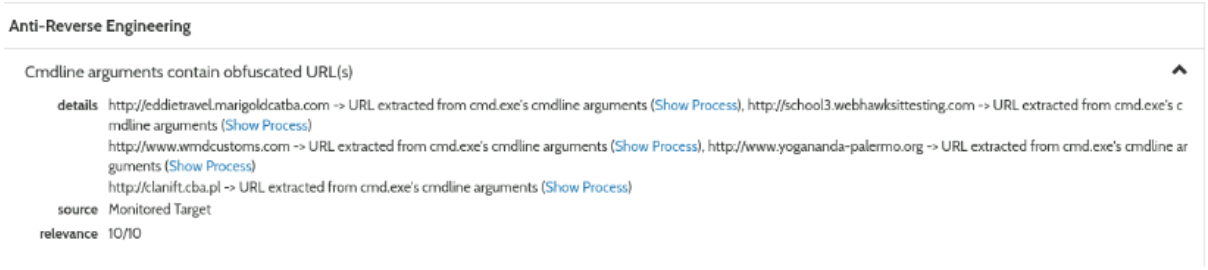


2) Indicators

In the indicators section of the report, you will find IOCs. This is split into two parts.
Malicious Indicators, Suspicious Indicators and informative.
The Malicious Indicators part includes all the activities that can be considered red flags right away, such as running PowerShell commands and pulling files from remote hosts. Note that all indicators in the list are collapsible and will give you more specific information if you click on them. For instance, the file that is being used as an example in this module has been detected to have an embedded VBA macro, which is represented in the following indicator:

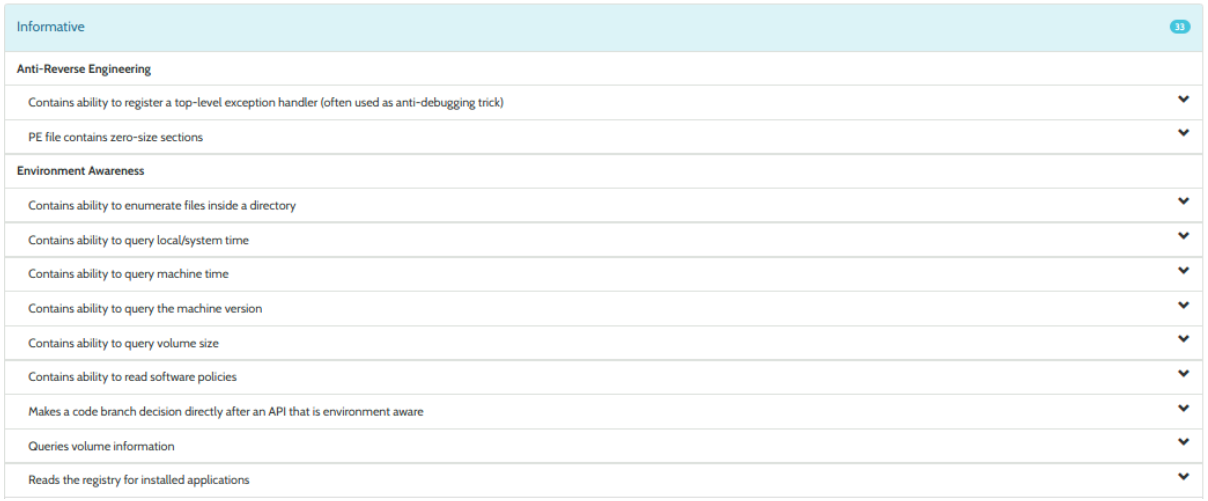
3) Suspicious Indicators:

The Suspicious Indicators part contains activities or techniques utilized by the malware that seem questionable, but the analysis tool is less certain regarding their maliciousness. Do keep in mind, however, that some of these techniques can seem quite obviously malicious to a human analyst, so don't just skip through this part of the report. For example, you can find clearly malicious indicators like the following one hidden in the Suspicious Indicators part:



4) Informative Indicators

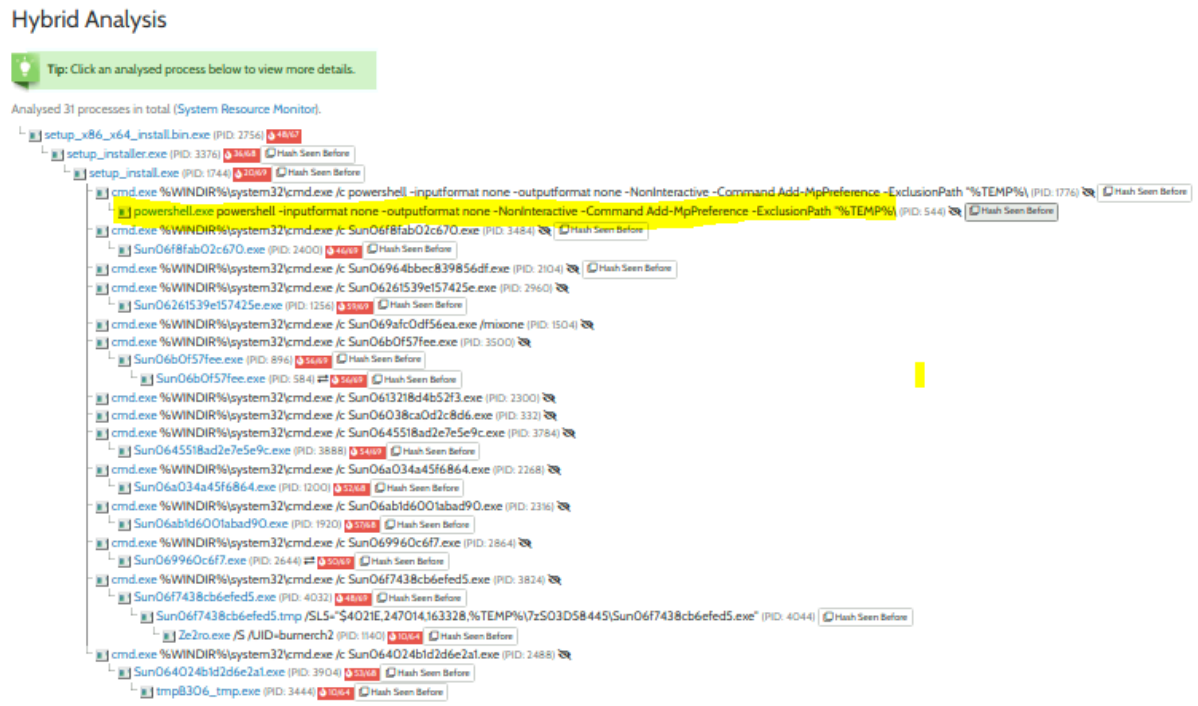
Informative part of the indicators section contains the list of all detected techniques and activities performed by the malware. This part can be especially extensive, so normally you would just briefly look through it because chances are the tool has caught and already marked all the notoriously bad indicators in the previous part of the indicators list



Hybrid Analysis:

The Hybrid Analysis section of the report can seem cryptic at first, but it provides a great insight into the malware's workflow. It presents all the processes ran by the malware in a tree view sorted in chronological order, so you can trace it every step of the way.

Here, the malware sample is trying to download some files from several remote hosts with the help of an obfuscated PowerShell command:

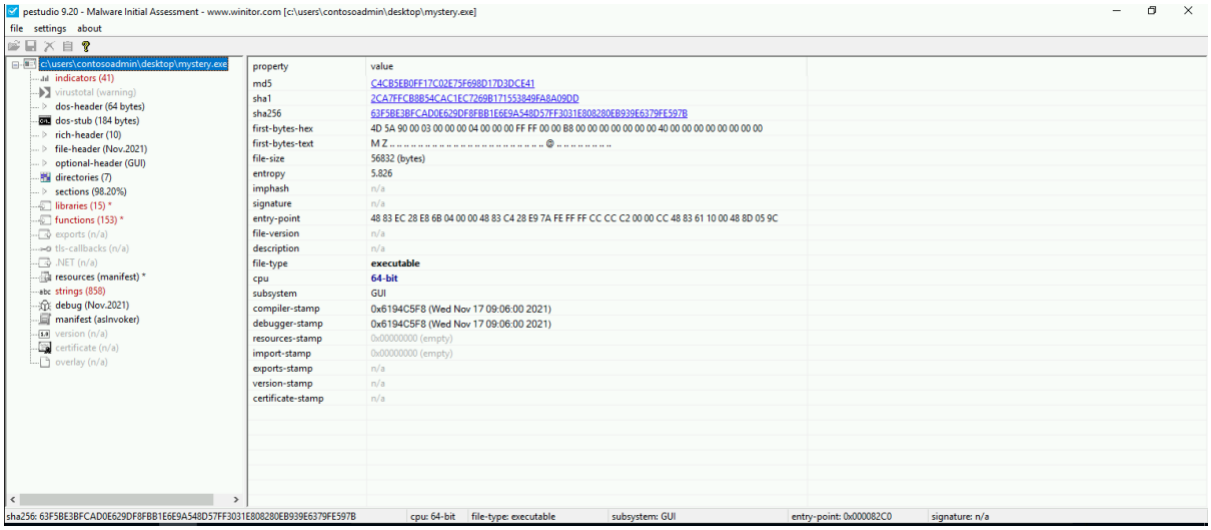


You can also click on every process to investigate it further. Hybrid Analysis records all the API calls, registry, and file manipulations committed by each process spawned by the malware.

8 Examining malware file using PE studio (Static Analysis)

8.1 Basic Information

When you first open a file in pestudio, you will see a window like this:
Open the mystery.exe file on your desktop using pestudio and look at some of the initially displayed information. To do this, launch pestudio using the shortcut on your desktop and either drag mystery.exe into the window or use the application menu to open it.



8.2 Hashes

When examining a potentially malicious file for the first time, it can be handy to start by grabbing the file's hash. You can think of a hash in this context as a fingerprint for the file. When you change even a single bit of data in a file, the hash of the file will be different. The results of three different algorithms displayed by pestudio (MD5, SHA1, SHA256) are effectively interchangeable for this purpose.

It is also possible to calculate a hash only for certain parts of the file, such as the headers or the imported libraries, also known as an imphash. This can be useful when you are trying to compare different files that you suspect are related. When malware authors release new versions, they might leave some parts of the file unchanged. Calculating hashes on specific parts of the file is something that pestudio also does for you.

You can use a file's hash to search online in VirusTotal, Hybrid Analysis, or any other similar databases or sandboxes. Pestudio will automatically search for the hash on VirusTotal, and if there is a match, pestudio will display the results directly.

By querying databases like this, you can sometimes quickly find out what you are dealing with, rendering further investigation unnecessary.

8.3 File Type:

Generally, pestudio will parse file type information out for you. For example, pestudio will attempt to detect if you are dealing with an executable and for which architecture it was compiled. However, it is good practice to be able to verify some of this information manually. Additionally, if pestudio cannot parse out any meaningful information, you might be able to see why by looking at the contents of the file directly.

For this purpose, pestudio conveniently shows you the first bytes of a file. The file you are currently examining is an executable, and you can see the start of the MZ header as you would expect. However, if you were to see a different signature or random data at the beginning of the file, then you would know to proceed accordingly.

For example, if someone hands you a file and says that it is malicious, but it looks like it is just random data, then maybe they have made a mistake, or perhaps the file contains encrypted data that is somehow used by some malicious application. This tells you what questions to ask going forward.

8.3 Compilation Time

The compilation timestamp can potentially be removed or faked by the developer but is still worth examining. This information can tell you how recently the sample was created — this can be important if you are dealing with an intrusion and are trying to understand if an attacker is still active. For instance, if the sample was compiled the day before, the attacker is likely actively attempting to use this tool.

The compilation time is also useful if you are examining multiple related malware samples and are trying to establish a timeline. If you have many associated samples, it can be easier to start your analysis with the older samples. They might be easier to understand if the malware has evolved to be more complex over time.

8.4 Entropy

Entropy, in simple terms, is a measure of randomness and is a good indicator for determining if the file is somehow compressed or packed. A packed executable or an encrypted data file will have a more random-seeming structure than a regular executable file containing CPU instructions, i.e., encrypted data will have higher entropy. Therefore, if you see those parts of the file or the entire file have high entropy, you could guess that compressed or encrypted data is stored within.

Note: Zero entropy means no randomness, e.g., a file filled with zeroes, and the maximum value of eight represents a completely random sequence of bits.

It is essential to determine if you are dealing with an executable that is somehow packed because then you know that you must get it unpacked before you can meaningfully analyze it any further. Until then, all the functionalities will be hidden from you for any static analysis. You can also get an indication of whether the executable is packed by examining the imports. This topic will be covered in the following steps.

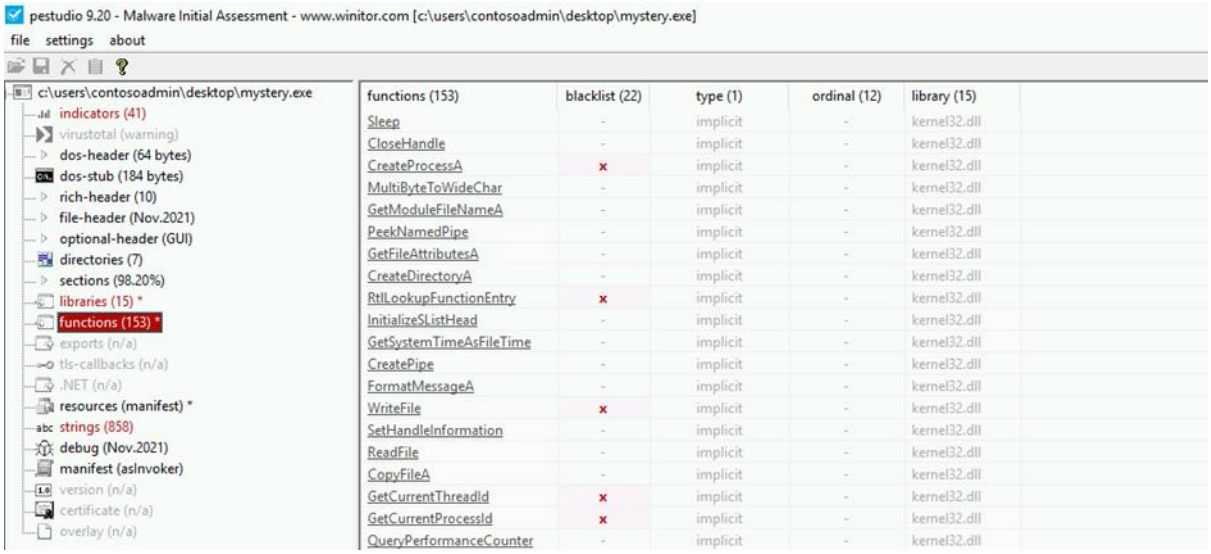
Note: Entropy is not always immediately telling since a packed executable can have similar entropy to one that has not been packed. Entropy is just one indicator.

8.5 Libraries and Functions

The information about libraries and functions is probably the most telling of what the malware might want to accomplish in broad terms.

Note: In pestudio, you will see a column called blacklist in various views. The items in this column are not inherently malicious. This only means that the item or functionality is commonly related to malware or is potentially dangerous in some way.

library (15)	blacklist (1)	type (1)	functions (153)	description
kernel32.dll	-	implicit	30	Windows NT BASE API Client DLL
user32.dll	-	implicit	1	Multi-User Windows USER API Client DLL
advapi32.dll	-	implicit	5	Advanced Windows 32 Base API
ws2_32.dll	x	implicit	14	Windows Socket 2.0 32-Bit DLL
msvcp140.dll	-	implicit	41	n/a
vcruntime140_1.dll	-	implicit	1	n/a
vcruntime140.dll	-	implicit	12	n/a
api-ms-win-crt-r...	-	implicit	18	n/a
api-ms-win-crt-s...	-	implicit	17	n/a
api-ms-win-crt-fi...	-	implicit	3	n/a
api-ms-win-crt-s...	-	implicit	4	n/a
api-ms-win-crt-h...	-	implicit	4	n/a
api-ms-win-crt-e...	-	implicit	1	n/a
api-ms-win-crt-...	-	implicit	1	n/a
api-ms-win-crt-l...	-	implicit	1	n/a



8.5.1 Libraries:

In the libraries view, you will see a list of libraries imported by this executable. This will give you an indication of what type of functionality the sample might have. There is a short description of the library in the description field, but you can always search online for more information or look ahead at the functions view which can sometimes be more descriptive since you can see what functions the library provides.
Note: The exports view is what you will want to look at if you analyze a DLL file. This view will tell you what functionality the DLL exposes.

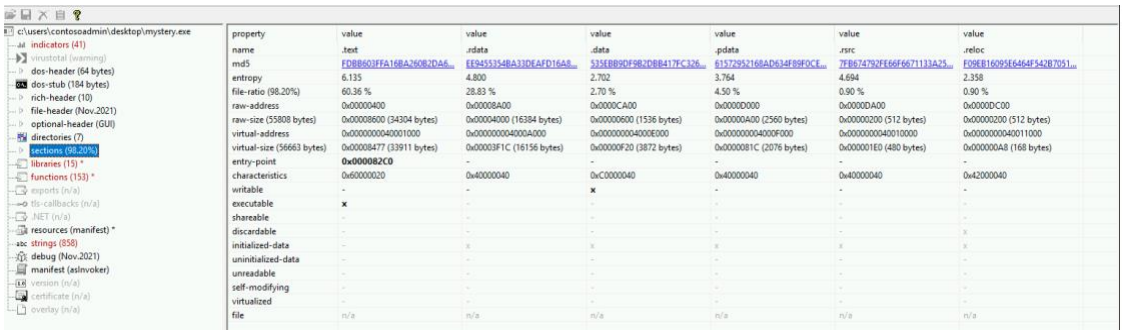
8.5.2 Functions:

The functions view lists the external functions potentially used by the program and is relatively self-explanatory. Pestudio will also mark the functions more commonly abused or potentially dangerous as denylisted.
You can see that as ws2_32.dll was marked as denylisted, all of the functions provided by that DLL are also marked as such. Additionally, in the functions view, you can see that this library implements networking functionality more easily.
Note: If you want more details on a particular function, you can search for it in Microsoft Docs.
If you only see a few functions listed here or only a few libraries imported, it could mean that the program is packed or is resolving the addresses of the libraries and functions by itself. For example, here is the list of functions shown for a UPX packed executable:

8.6 Other Information

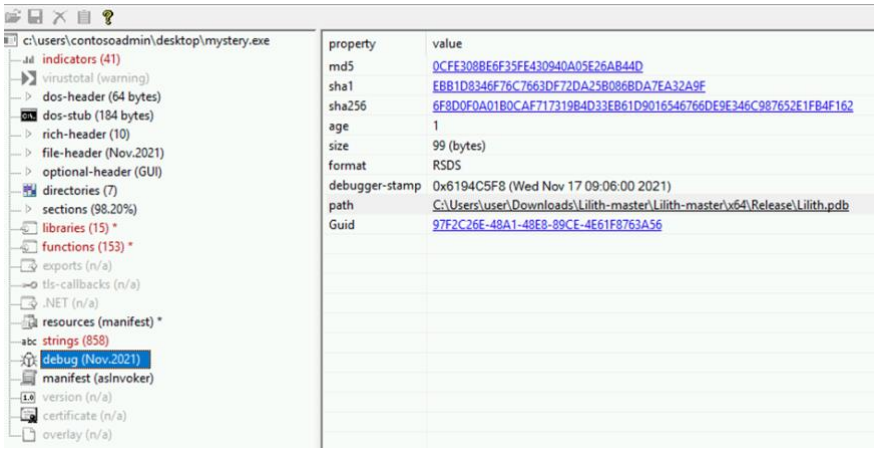
8.6.1 Headers and Sections

This information is usually not all that relevant for initial analysis. The relevant aspects here are parsed out by pestudio and displayed for you on the initial summary view. However, if you need to examine the file's headers or sections more closely, pestudio will also display that information in more detail as well.



8.6.2 Debug Information

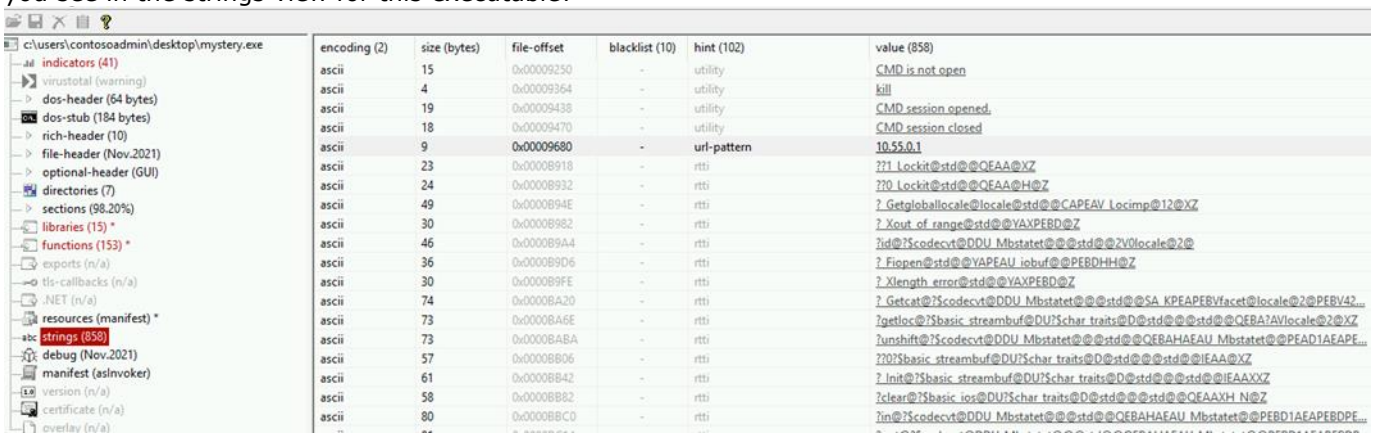
Specific information that is of particular interest in the debug view is the PDB string displayed in the path field. You don't need to understand how the PDB file is used. In this context, it is only important to understand that this path can tell you what the developer calls this program or project. This information, similarly, to the compilation timestamp, is useful when trying to understand the broader context of the sample you are analyzing.



8.6.3 Strings

Sometimes taking a quick look at the strings that the executable stores in the open can give some

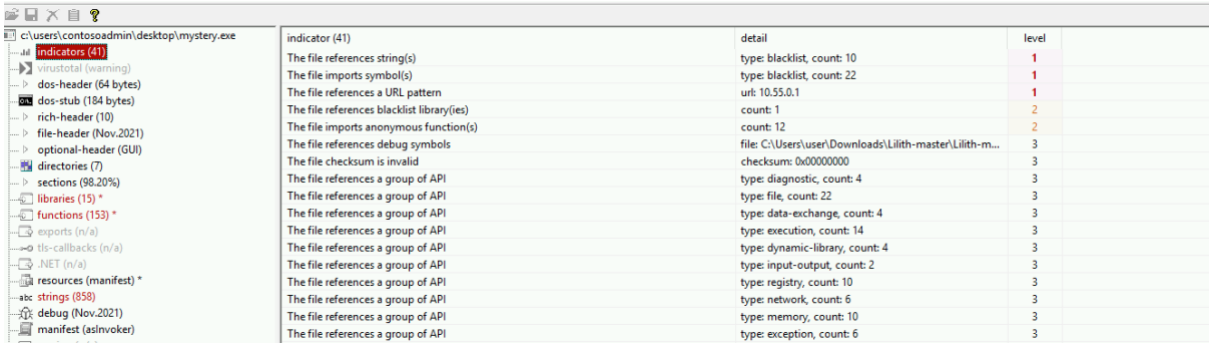
information about the nature of the file you are dealing with. For example, does the executable store strings that hint functionality (e.g., CMD session opened), or does it store anything that looks like a C2 address?
Usually, these types of strings are stored encrypted, but sometimes you can get lucky. What can you see in the strings view for this executable?



8.6.4 Indicators

The indicators view is a feature specific to pestudio. Pestudio will try to parse out features of an executable that might indicate that it is malicious. It can be useful to start by looking here and seeing anything interesting that pestudio has extracted. You should be able to see here that pestudio has parsed out the IP address and the PDB path that you can also find in the strings and debug views.

The indicators feature can also be useful in the sense that if there are a lot of red flags here, then there is a higher chance that you are dealing with something that is indeed malicious. However, keep in mind that if tools such as pestudio do not report anything suspicious, then that does not automatically mean that everything is fine. It might be, but it is your task to make that assessment.



9 Exception

Exception to this procedure must be approved through the Netradyne Exception Process.

10 References

10.1 Logging

A ticket should be raised for each and every malware analysis
Ticket can be raised through service desk+ portal

10.2 IOC

Every IOC found during analysis, we have to block on different
Please raise ticket for same through service desk+ portal

11 Appendix A: Document RACI Matrix

Role/Activity	Document Owner/Functional Area Lead	Document Contributor	ND Leadership	Functional Area Team	InfoSec	All ND Member(s)
Ensure document is kept current	**A**	**R**	**I, C**	**R, C**	**C**	**I**
Ensure stakeholders are kept informed	**A**	**R**	**-**	**R**	**C**	**-**
Ensure document contains all relevant information	**A**	**R**	**I, C**	**R, C**	**C**	**I**
Ensure document adheres to document governance policy	**A, R**	**R**	**I**	**R, C**	**R, C**	**I**
Provide SME advice	**I, R**	**A, R**	**I**	**R, C**	**I, C**	**I**
Gathering and adding document contents	**I**	**A, R**	**I, C**	**R, C**	**C**	**I**
Document Approval	**A**	**R**	**I, R**	**I**	**I, R**	**I**

Key

R	**Responsible**
A	**Accountable**
C	**Consulted**
I	**Informed**