Cuckoo Sandbox Lab

Technical Requirements:

Cuckoo Sandbox

File Link: https://drive.google.com/file/d/1Z9apAIet5YAfj3T-gPMTCb8_q0kgfbkq/view?usp=sharing

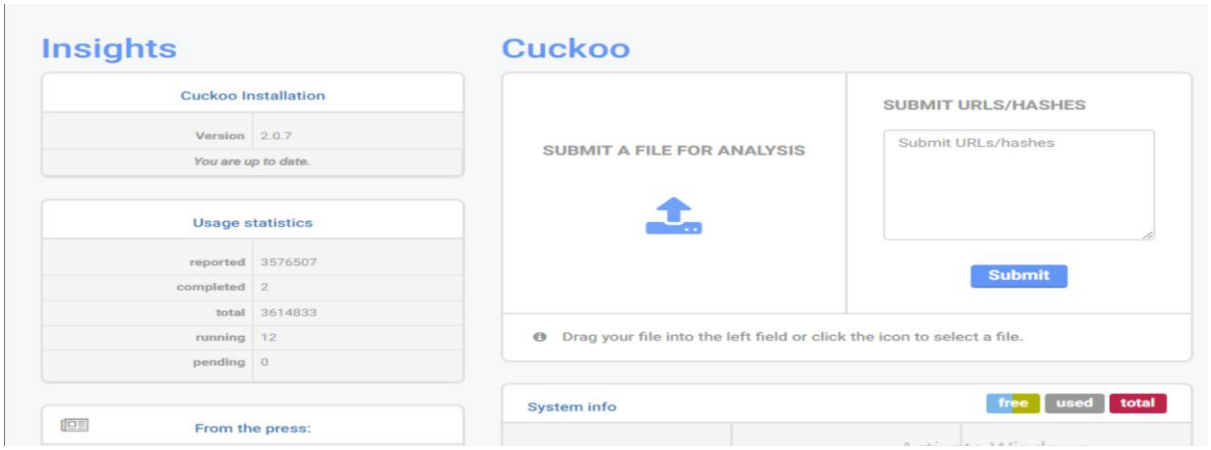
Don't extract the file.

Objective:

To load and analyse malware in the cuckoo sandbox.

Task 1: Upload the Malware Sample

- Open Cuckoo Sandbox in your browser.
- Locate the submit file window.
- Drag and drop the AgentTesla.exe (ZIP) file into the window.
- Click the Submit button to upload the file.

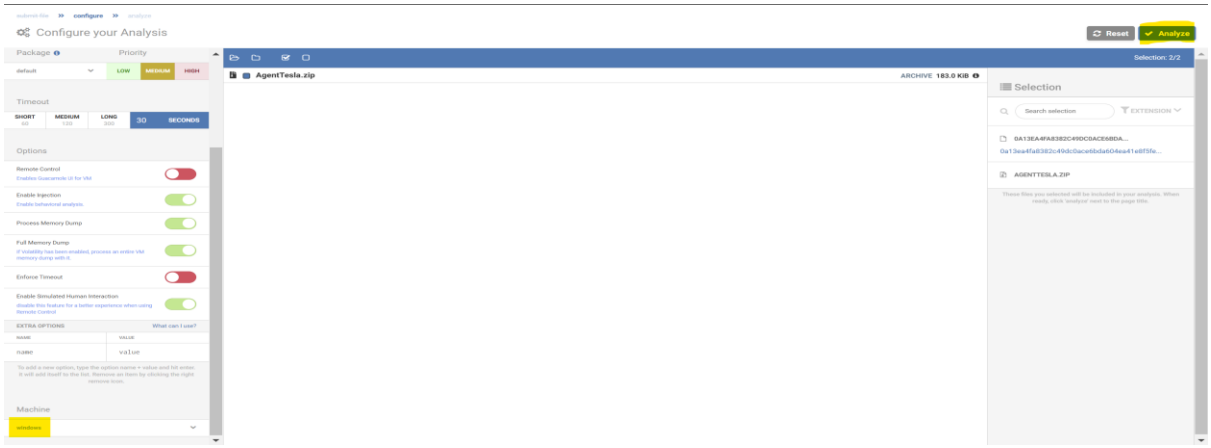


2. Select Analysis Options

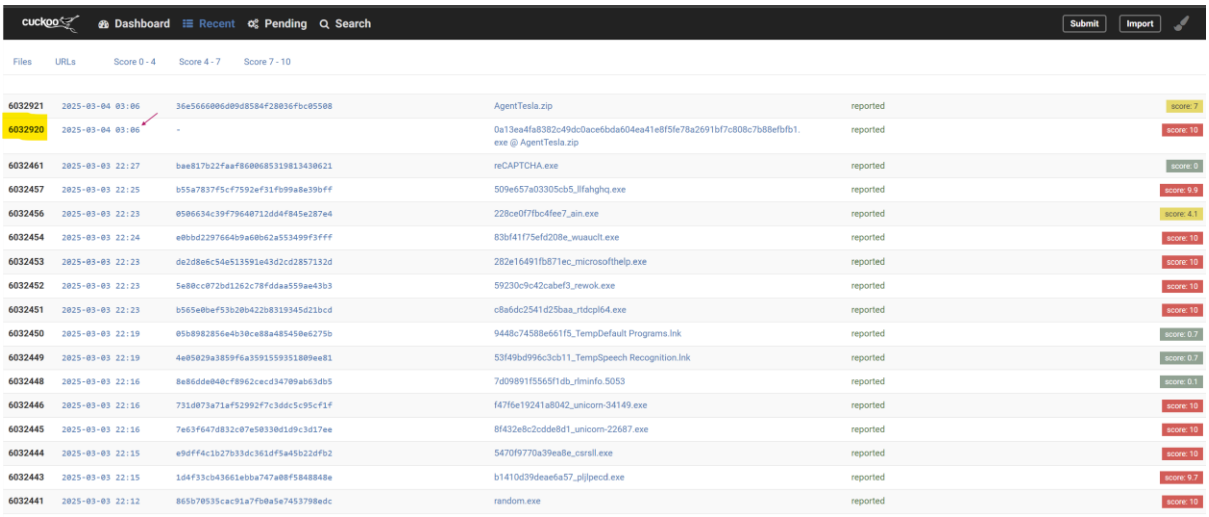
- After submission, you will be redirected to a new page with various analysis options.

Configure the following:

- **Network Routing**: (by default, It is selected as INTERNET. if not, manually select it)
- **Timeout**: Adjust if needed, default settings are usually sufficient.
- **Additional Options**: (Optional, leave as default for standard analysis)
- **Machine Selection**: From the dropdown, select Windows as the target system.
- **Analyze**: select it at the top right corner of the screen

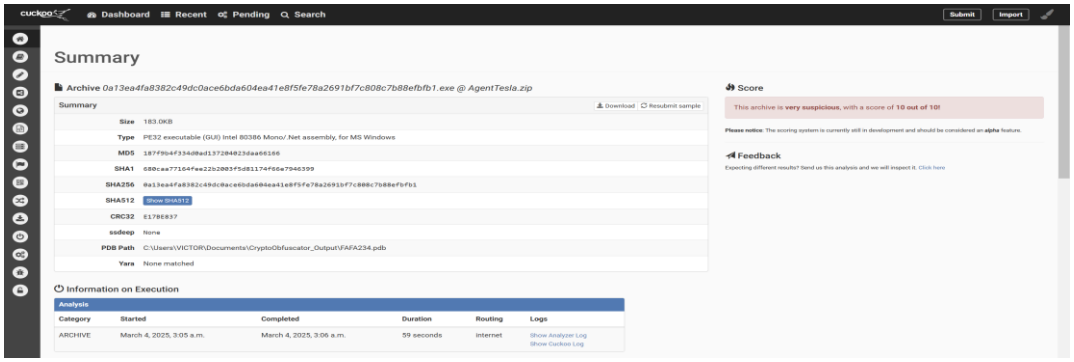


Cuckoo will analyze the executable, and you can track its progress on the right side of the screen next to the file entry. Since the file is in a ZIP format, Cuckoo needs to extract it before starting the actual analysis, which may take some extra time. Once the status changes to "Completed" and then "Reported, go to the “Recent" tab. Look for the extracted executable file, not the ZIP file, using the Task ID, then click the highlighted URL to open the report.



Task 2: Reviewing Executable summary.

Once you are redirected to the executable summary page, please notice the file score which shows that this executable has, and this would give an initial idea of how suspicious/malicious this file is.



Scroll down to the Signatures Section to review detailed insights about the file's behavior. The signatures are ordered by severity, with the most critical ones highlighted in red, indicating highly malicious activity.

For the current AgentTesla sample, one key observation is that it allocates read-write-execute memory, which suggests it may be injecting code or unpacking itself in memory. This behavior can be further analyzed by reviewing the file signatures to understand how the malware operates.

Answer the following questions by observing the Summary tab and the Signatures section.

Question 1: Does the malware use process injection techniques? If so, which APIs does it utilize?

Under Allocates read-write-execute memory in the signature section, APIs used NtProtectVirtualMemory and NtAllocateVirtualMemory.

Please observe the rest of the signatures marked in red to understand more about the executable being analysed.

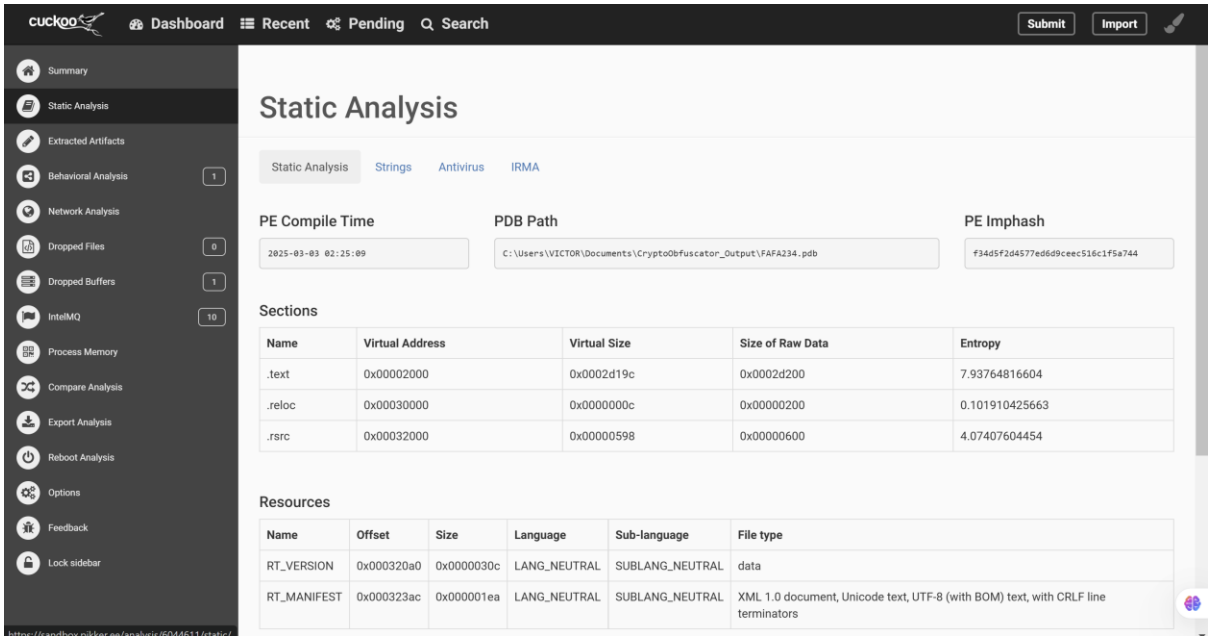
Question 2: How many antivirus engines flagged this file as malicious in the IRMA (Incident Response & Malware Analysis) scan?

The file was flagged as malicious by 11 antivirus engines in the IRMA scan

Question 3**:** Is the malware packed or obfuscated? If yes, what tool or method does it appear to use?

Yes, the malware is obfuscated using CryptoObfuscator, as indicated in the PDB path.

You can now select static analysis from the left pane to view file details view details such as Sections, imports and the strings that are found within this executable. We can also inspect different executable details such as its behavioural analysis response and dropped buffer details.



Answer the following questions by observing the static analysis tab.

Question 4: What does the PDB path reveal about the malware?

The PDB path suggests that the malware was built using CryptoObfuscator, a tool used to hide code and evade detection. The presence of "VICTOR" in the path may indicate the developer's system username.

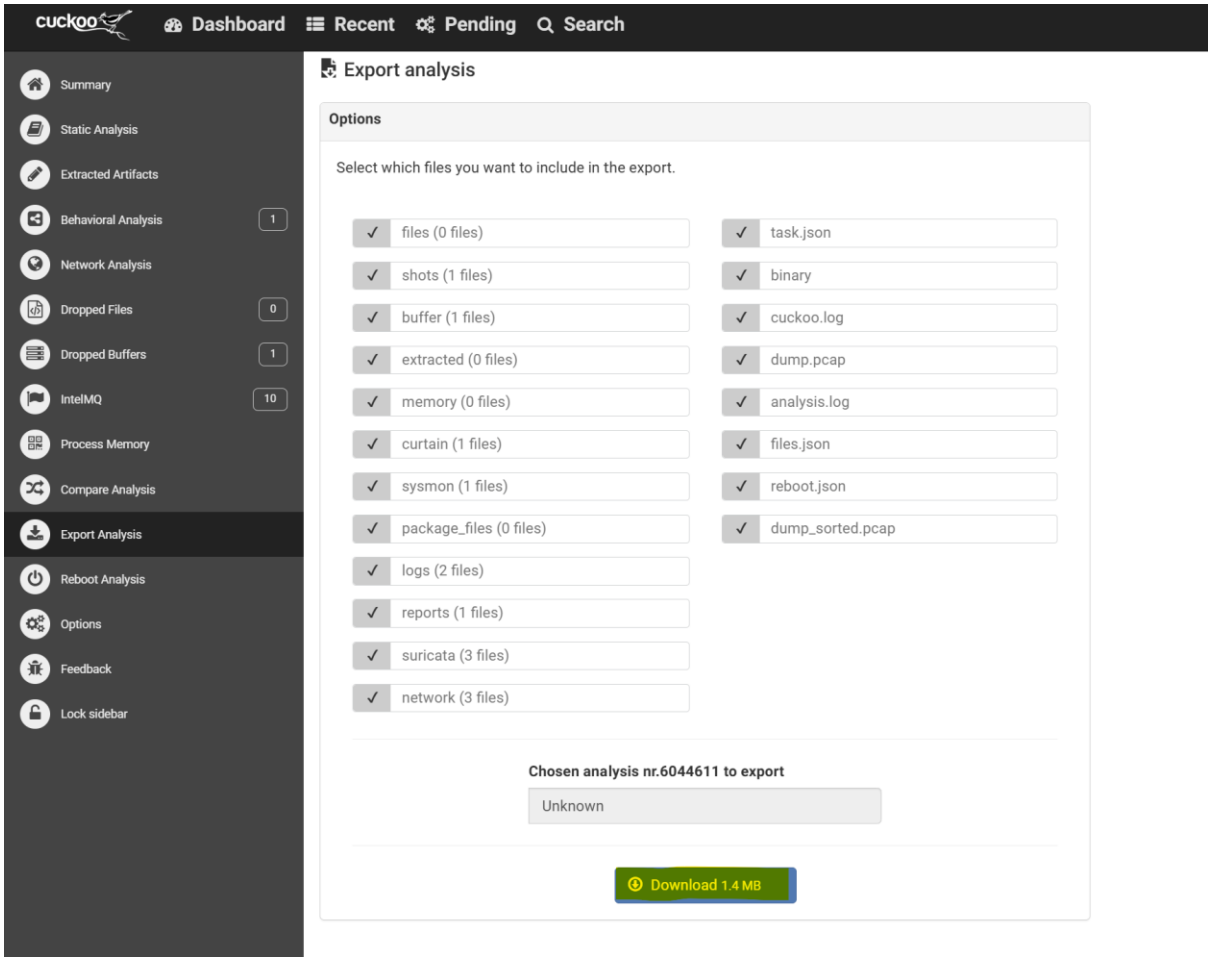
Question 5: The .text section of the executable has an entropy value of 7.93. What does this indicate?

An entropy value of 7.93 in the .text section indicates that the executable is likely packed or obfuscated, as high entropy suggests compressed or encrypted code.

Question 6: What is the overall behavior of the malware?

The malware manipulates memory permissions and injects code into other processes to run stealthily. It is obfuscated using CryptoObfuscator to avoid detection and allocates read-write-execute memory, likely to unpack itself or execute code within another process.

We can also export the analysis results by click on ‘Export Analysis’ and downloading the analysis files such as the logs, suricata files and pcap dumps.


cuckoo
Dashboard
Recent
Pending
Search
Summary
Static Analysis
Extracted Artifacts
Behavioral Analysis
1
Network Analysis
Dropped Files
0
Dropped Buffers
1
IntelMQ
10
Process Memory
Compare Analysis
Export Analysis
Reboot Analysis
Options
Feedback
Lock sidebar
Export analysis
Options
Select which files you want to include in the export.
files (0 files)
shots (1 files)
buffer (1 files)
extracted (0 files)
memory (0 files)
curtain (1 files)
sysmon (1 files)
package_files (0 files)
logs (2 files)
reports (1 files)
suricata (3 files)
network (3 files)
task.json
binary
cuckoo.log
dump.pcap
analysis.log
files.json
reboot.json
dump_sorted.pcap
Chosen analysis nr.6044611 to export
Unknown
Download 1.4 MB

End of Lab