Malware Simulation & Analysis Report

Author: Purna Kishore Kondapaneni
Project Title: Binary Executable Simulation for Malware Analysis

---

1. Objective

The goal of this project is to simulate a malware executable that performs malicious actions such as downloading a payload from the internet, creating and deleting local files, and modifying the Windows registry. The executable is then analyzed through static and dynamic techniques using various malware analysis tools, followed by reverse engineering and crafting of YARA rules for detection.

---

2. Malware Creation

2.1 C Code Description

The malware executable performs the following operations:

- Downloads a remote executable using HTTP.
- Creates and deletes two files in the local system.
- Writes a custom registry key-value pair.

2.2 C Code Sample (malware.c)

Here, Used C code, using Windows APIs, where it first downloads payload.exe from http://127.0.0.1/. Then, it creates test_file1.txt with specific content and test_file2.dat with binary data. Next, it creates a registry key Software\Evil app and sets a Config value to "malicious data". Finally, it attempts to delete both created files. In essence, it's a simple program demonstrating file download, creation, deletion, and registry modification on a Windows system.

2.3 Compilation on FLARE VM

gcc malware.c -o malware.exe -lwininet

1. Save the code as malware.c.
2. Open Command Prompt as an administrator.
3. Compile: gcc malware.c -o malware.exe -lwininet
4. Verify malware.exe is created.

2.4 Local Server Setup (Python) / Setting up the Simple HTTP Server (Simplified)

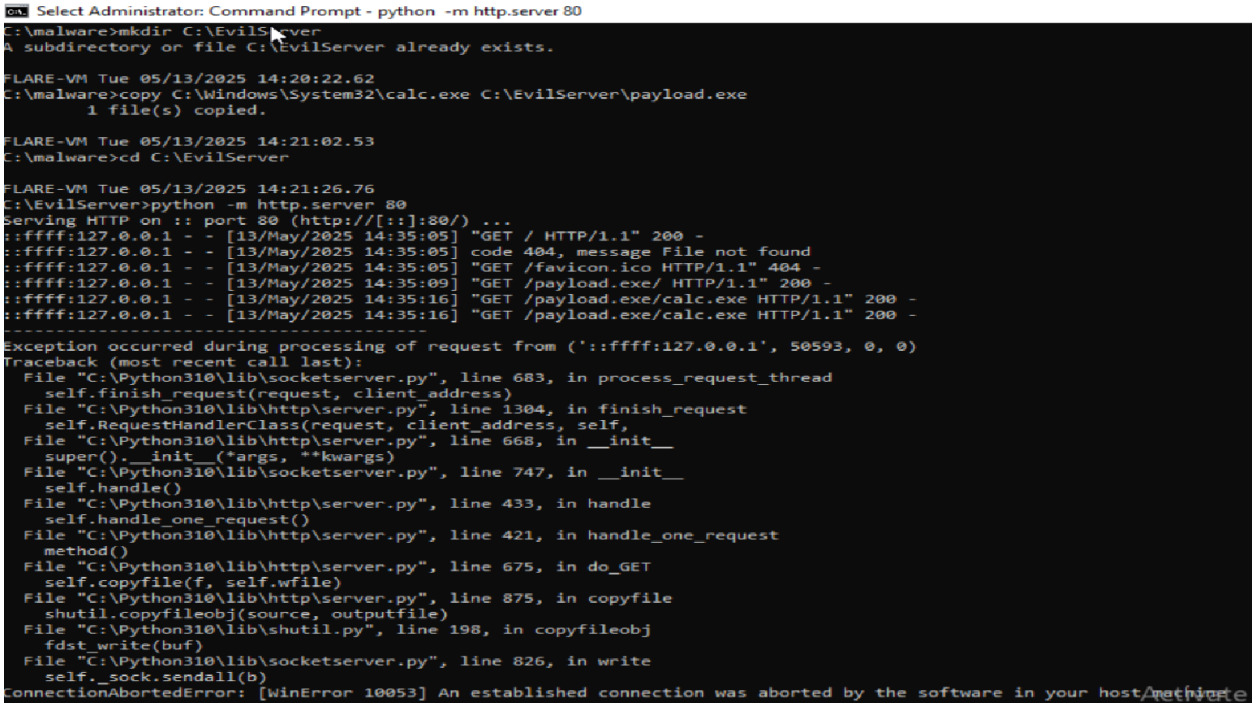
Preparing a Payload:

mkdir C:\\EvilServer

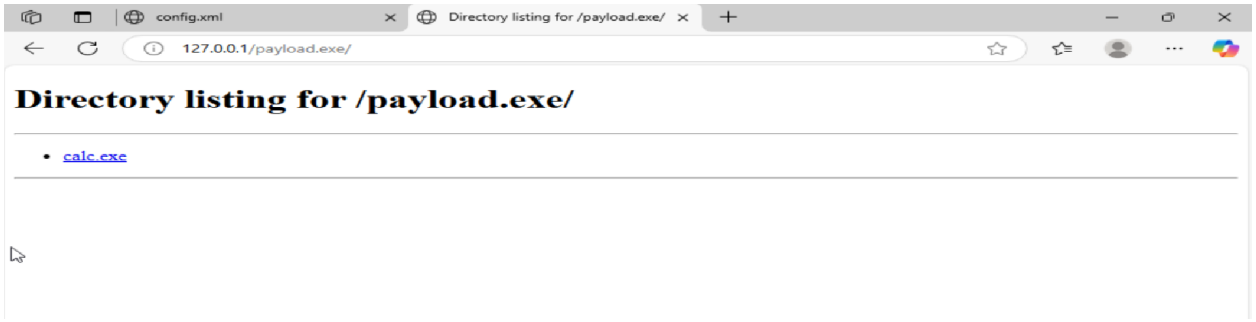
copy C:\\Windows\\System32\\calc.exe C:\\EvilServer\\payload.exe

Starting a Server:

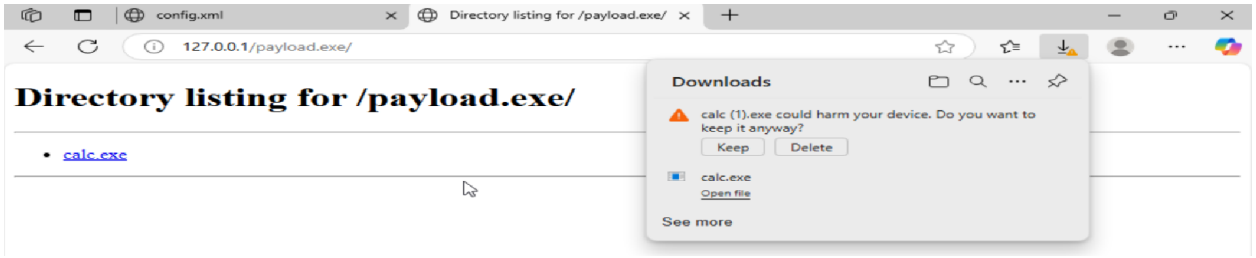
- In the same Command Prompt, navigate to C:\EvilServer:
- cd C:\EvilServer
- Run: python -m http.server 80

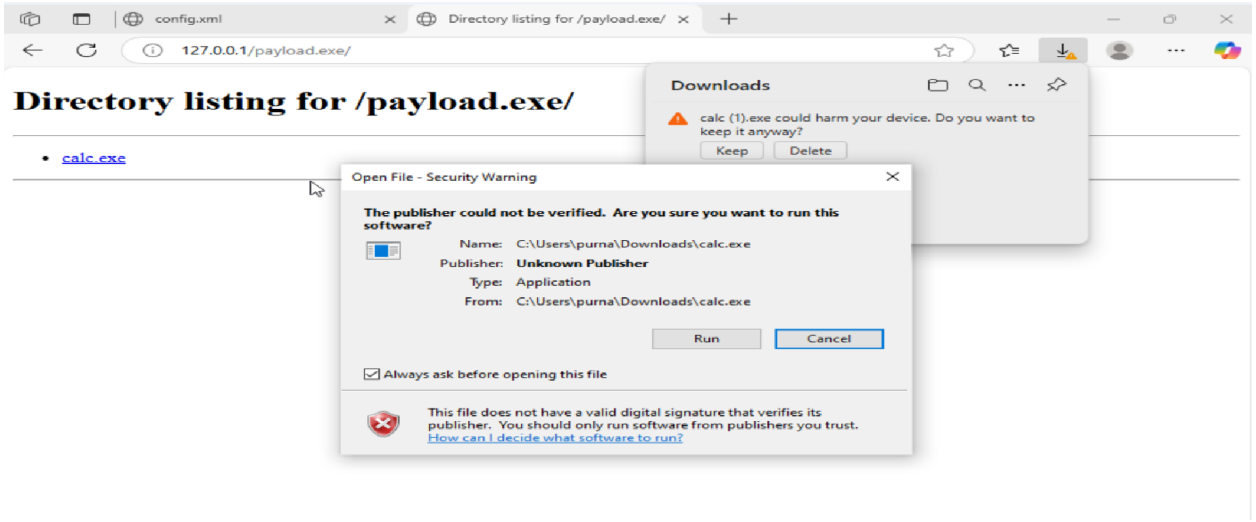


chose calc.exe as a benign payload for testing. Creating the C:\EvilServer directory and copying the executable.



When I clicked on the link, it was downloaded and able to open a calculator





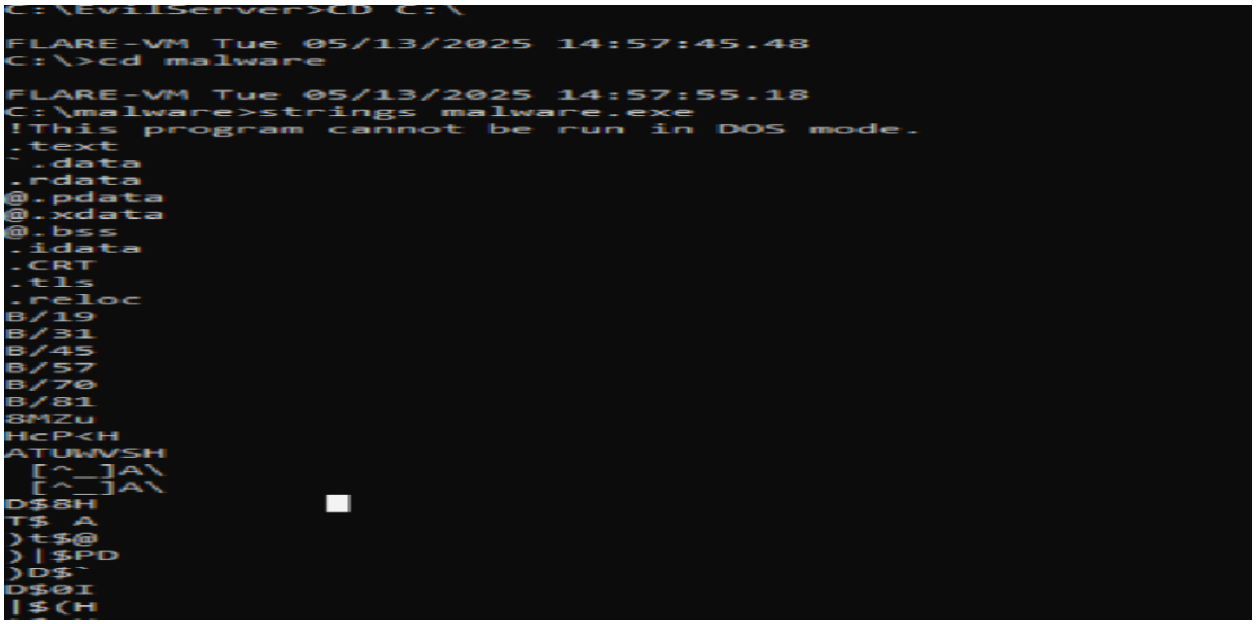
3. Static Analysis

3.1 Tools Used

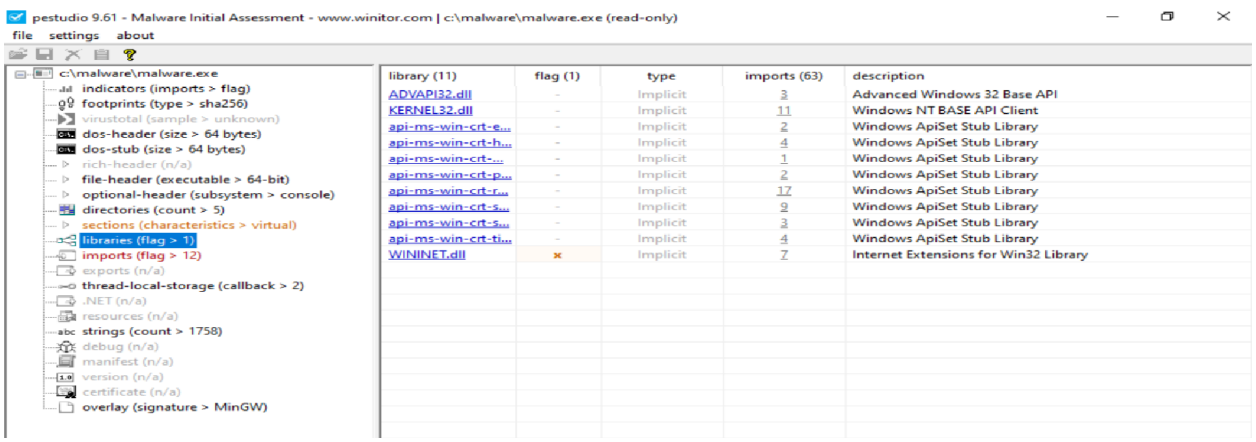
- Strings Utility
- PEStudio
- Dependency Walker
- Registry Editor
- Detect It Easy
- CFF Explorer

3.2 Observations

- **Strings:** URL, file paths, registry keys are visible as we defined in the C code.



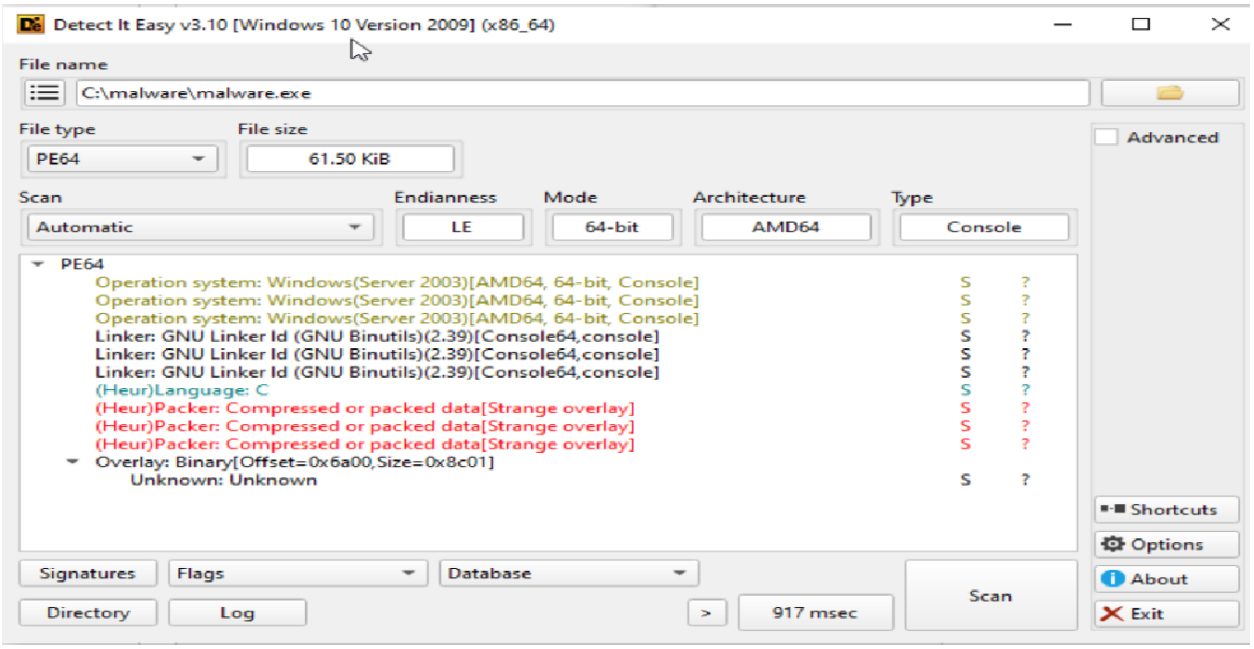
- **PEStudio:** No digital signature, no packing.



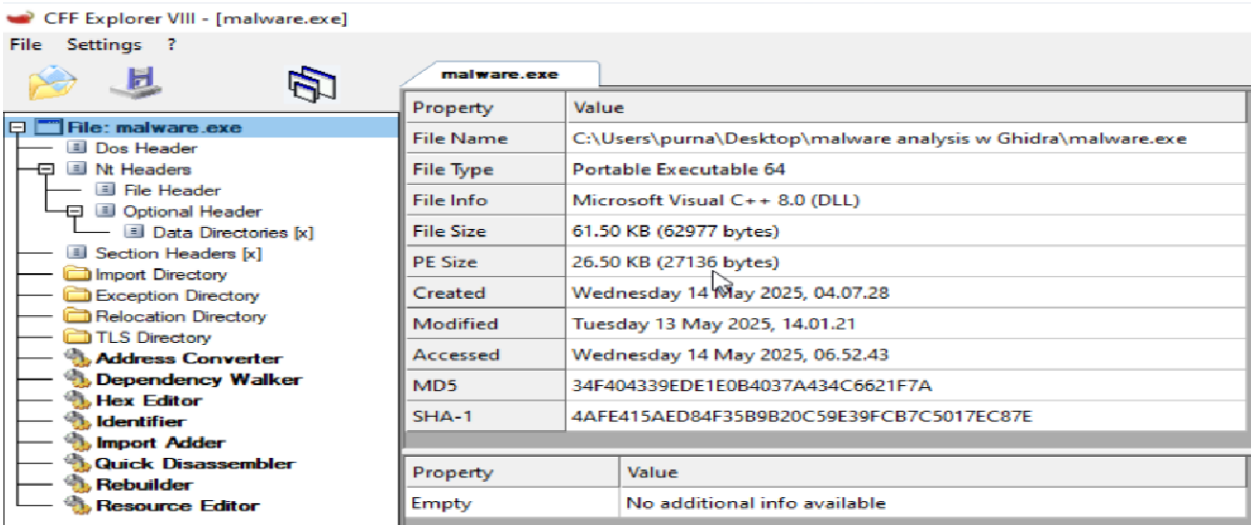
- **Dependency Walker:** Identifies WinAPI dependencies (Wininet.dll, Kernel32.dll). Dependency Walker is an application that builds a hierarchical tree diagram of other programs and all dependent modules. For each module found, it lists all the functions that are exported by that module, and which of those functions are actually being called by other modules.

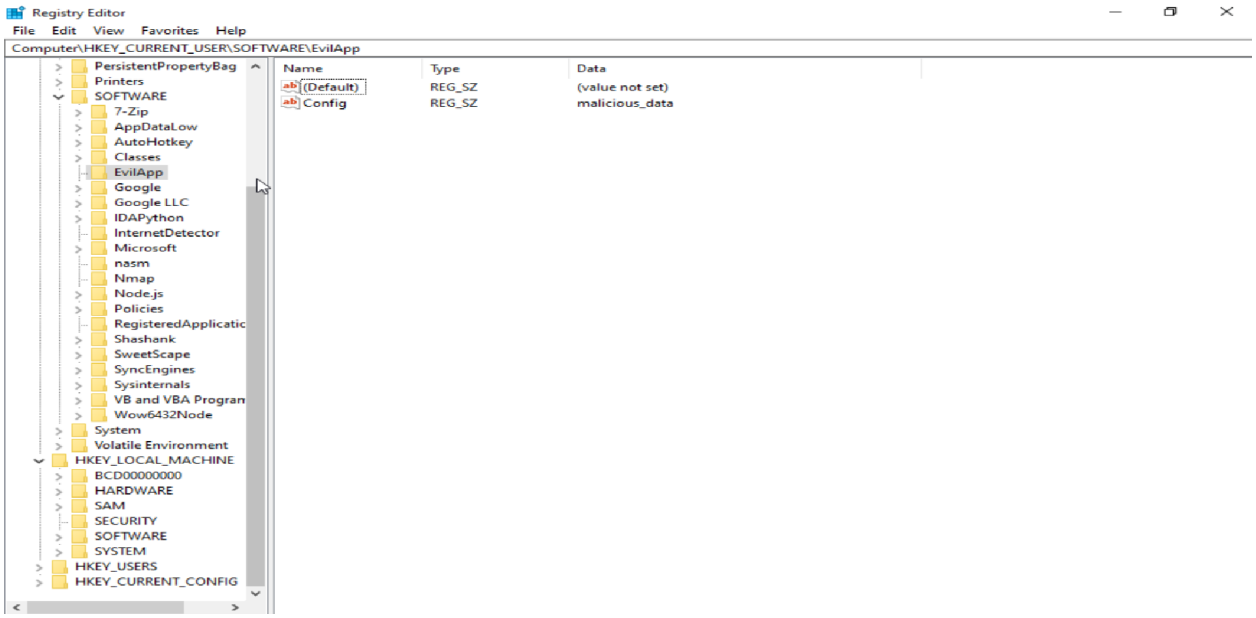
```
FLARE-VM Wed 05/14/2025  2:27:45.27
C:\malware>malware.exe;
Downloading file from http://127.0.0.1/payload.exe to C:\Users\Public\evil_payload.exe...
HttpSendRequestA failed: 12007
File download failed.
Creating local files...
Created: C:\Users\Public\test_file1.txt
Created: C:\Users\Public\test_file2.dat
Registry key created: HKEY_CURRENT_USER\Software\EvilApp
Registry value set: malicious_data = ╚QVS\@
Deleting local files...
Deleted: C:\Users\Public\test_file1.txt
Deleted: C:\Users\Public\test_file2.dat
```

- **Detect It Easy:** Recognized as PE64 executable.



- **CFF Explorer, HEX Editor:** Valid PE structure.



- **Registry Editor:** Custom key HKCU\\Software\\EvilApp with value Config = malicious_data.
- 

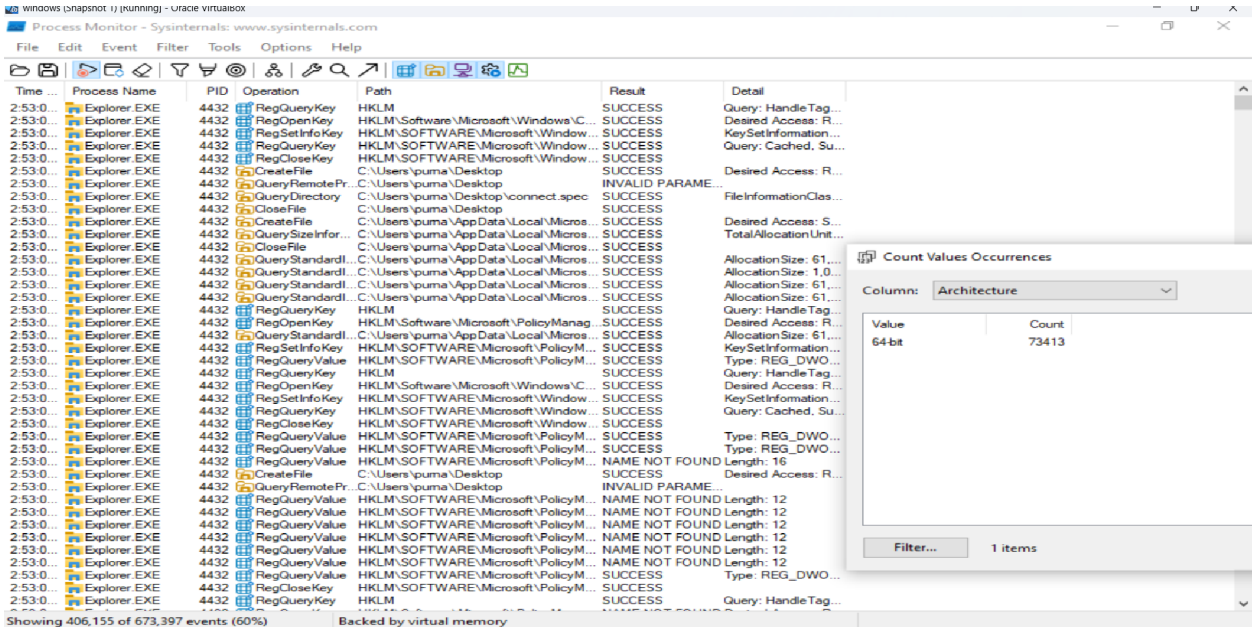
4. Dynamic Analysis

4.1 Tools Used

- Procmon
- Regshot
- virustotal.com

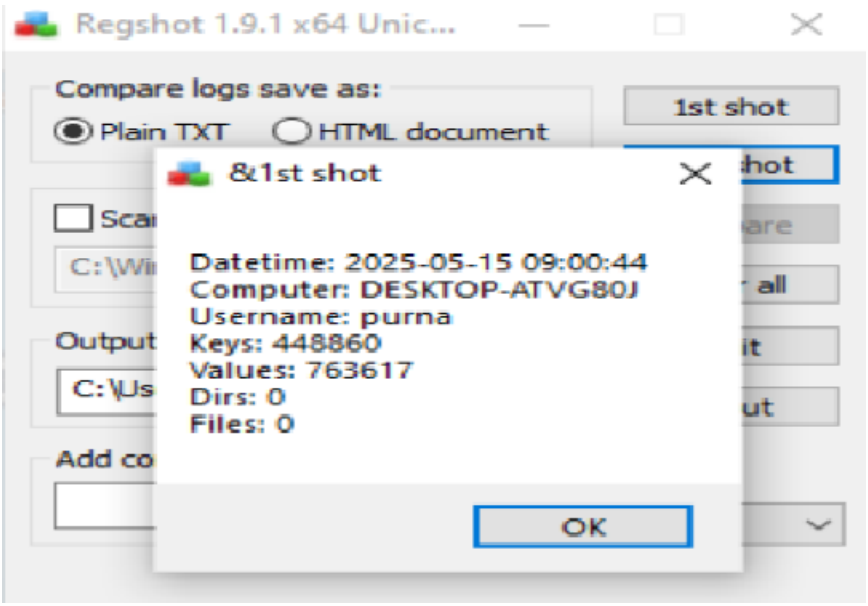
4.2 File and Registry Activity (Procmon)

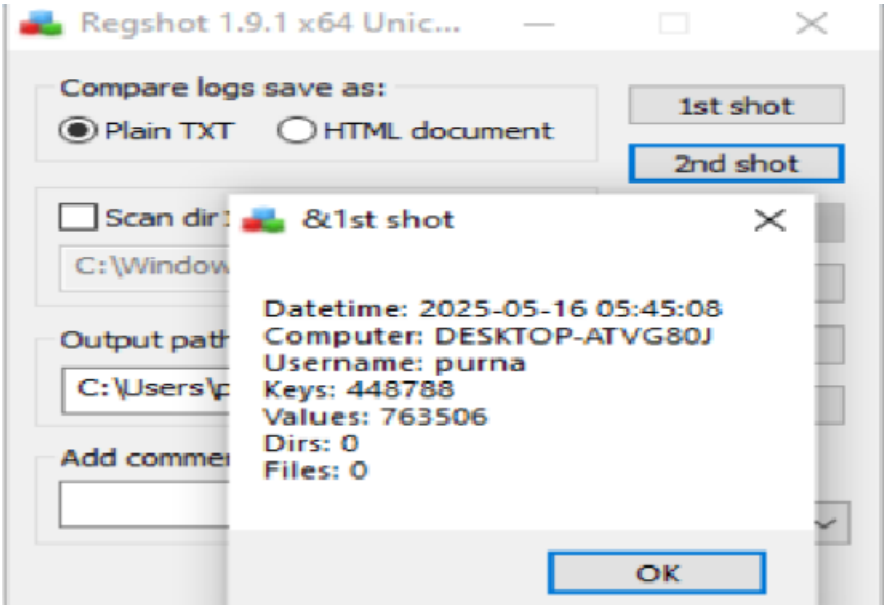
- File create/delete actions captured.
- Registry modification logged under HKCU\\Software\\EvilApp
- calc.exe downloaded and stored as evil_payload.exe



4.3 Regshot Comparison

- New registry key added: HKCU\\Software\\EvilApp
- No additional persistent changes

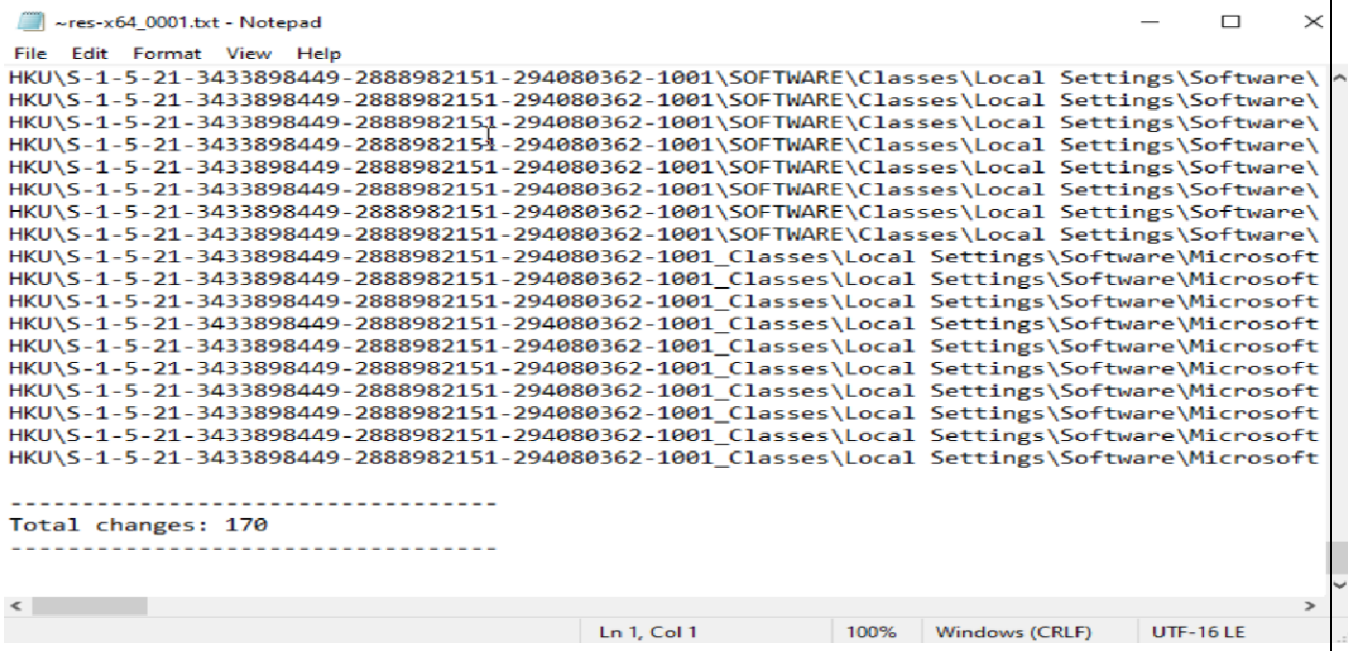




File Edit Format View Help

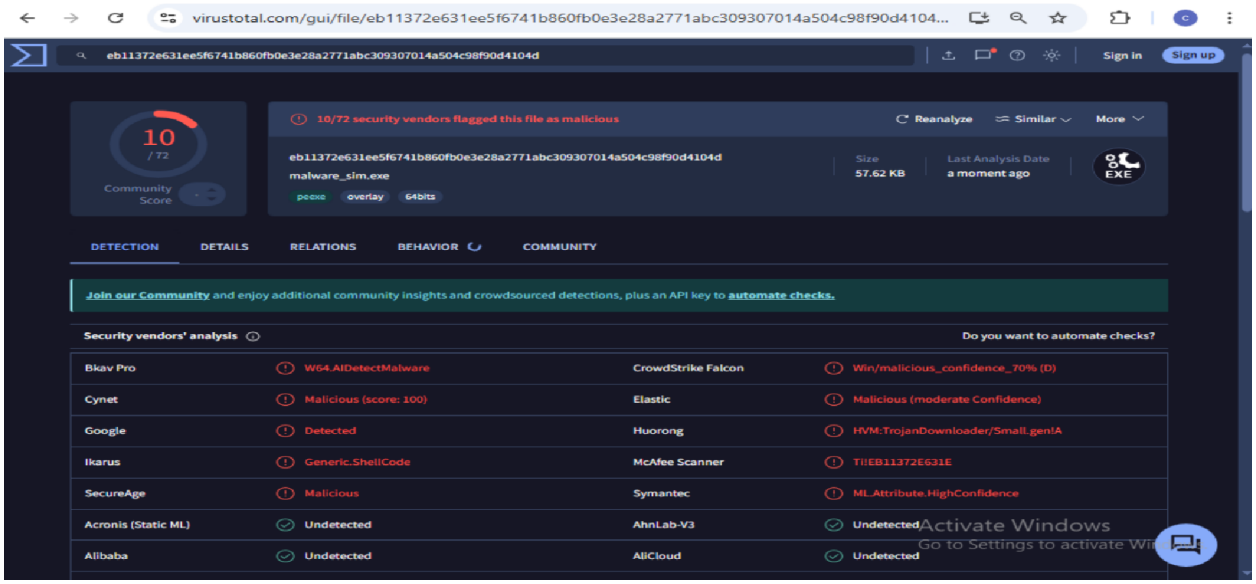
```
Regshot 1.9.1 x64 Unicode (beta r321)
Comments:
Datetime: 2025-05-14 07:04:37, 2025-05-14 07:05:42
Computer: DESKTOP-ATVG80J, DESKTOP-ATVG80J
Username: purna, purna

----------------------------------
Values modified: 14
----------------------------------
HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\VFUProvider\StartTime: 0x01DBC49E5E19699C
HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\VFUProvider\StartTime: 0x01DBC49E81DC6873
HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tasks\{857A9CCF-4082-4F7F-8AD1-87C165346D72}\DynamicInfo
HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tasks\{857A9CCF-4082-4F7F-8AD1-87C165346D72}\DynamicInfo
HKLM\SYSTEM\ControlSet001\Services\bam\State\UserSettings\S-1-5-21-3433898449-2888982151-294080362-1001\SequenceNumber: 0x000
HKLM\SYSTEM\ControlSet001\Services\bam\State\UserSettings\S-1-5-21-3433898449-2888982151-294080362-1001\SequenceNumber: 0x000
HKLM\SYSTEM\ControlSet001\Services\bam\State\UserSettings\S-1-5-21-3433898449-2888982151-294080362-1001\\Device\HarddiskVolum
HKLM\SYSTEM\ControlSet001\Services\bam\State\UserSettings\S-1-5-21-3433898449-2888982151-294080362-1001\\Device\HarddiskVolum
HKLM\SYSTEM\ControlSet001\Services\bam\State\UserSettings\S-1-5-21-3433898449-2888982151-294080362-1001\\Device\HarddiskVolum
HKLM\SYSTEM\ControlSet001\Services\bam\State\UserSettings\S-1-5-21-3433898449-2888982151-294080362-1001\\Device\HarddiskVolum
HKLM\SYSTEM\CurrentControlSet\Services\bam\State\UserSettings\S-1-5-21-3433898449-2888982151-294080362-1001\SequenceNumber: 0
HKLM\SYSTEM\CurrentControlSet\Services\bam\State\UserSettings\S-1-5-21-3433898449-2888982151-294080362-1001\SequenceNumber: 0
HKLM\SYSTEM\CurrentControlSet\Services\bam\State\UserSettings\S-1-5-21-3433898449-2888982151-294080362-1001\\Device\HarddiskV
HKLM\SYSTEM\CurrentControlSet\Services\bam\State\UserSettings\S-1-5-21-3433898449-2888982151-294080362-1001\\Device\HarddiskV
HKLM\SYSTEM\CurrentControlSet\Services\bam\State\UserSettings\S-1-5-21-3433898449-2888982151-294080362-1001\\Device\HarddiskV
HKLM\SYSTEM\CurrentControlSet\Services\bam\State\UserSettings\S-1-5-21-3433898449-2888982151-294080362-1001\\Device\HarddiskV
HKU\S-1-5-21-3433898449-2888982151-294080362-1001\SOFTWARE\Microsoft\Windows\CurrentVersion\ActivityDataModel\ReaderRevisionI
HKU\S-1-5-21-3433898449-2888982151-294080362-1001\SOFTWARE\Microsoft\Windows\CurrentVersion\ActivityDataModel\ReaderRevisionI
HKU\S-1-5-21-3433898449-2888982151-294080362-1001\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\UserAssist\{CEBFF5CD-ACE
0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 00 00 00
HKU\S-1-5-21-3433898449-2888982151-294080362-1001\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\UserAssist\{CEBFF5CD-ACE
0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0
```



4.4 Virus Total Scan

- Simulated sample recognized by several AV engines as "Malicious/Trojan"

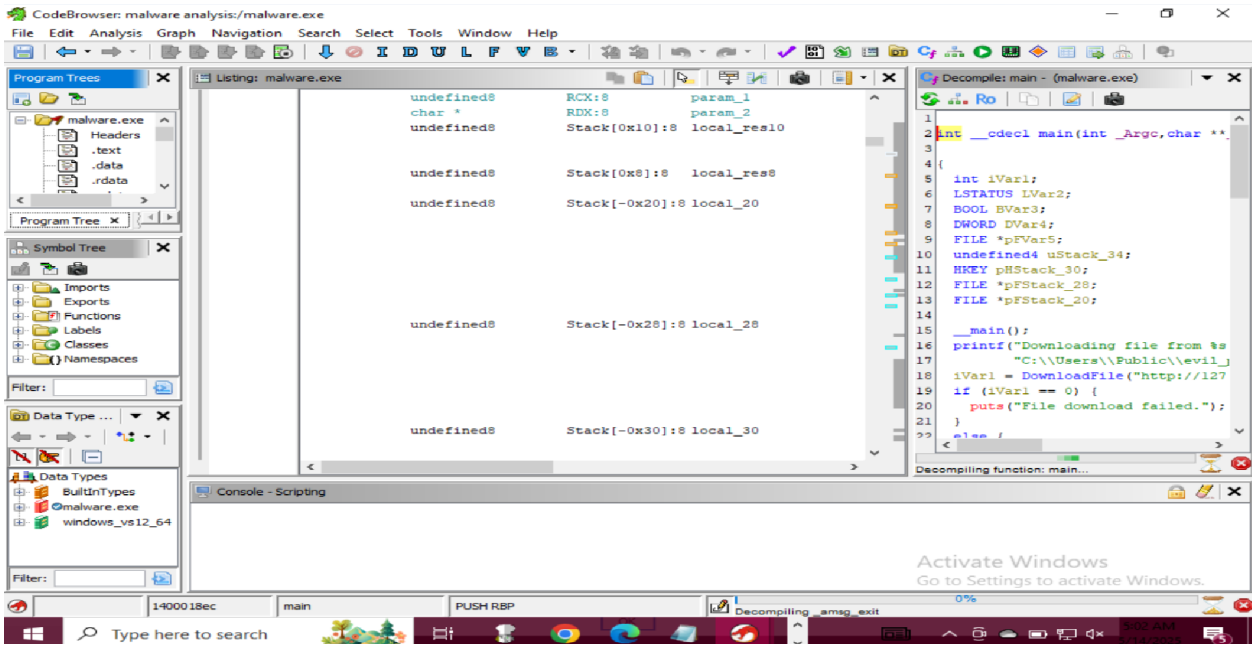


5. Reverse Engineering

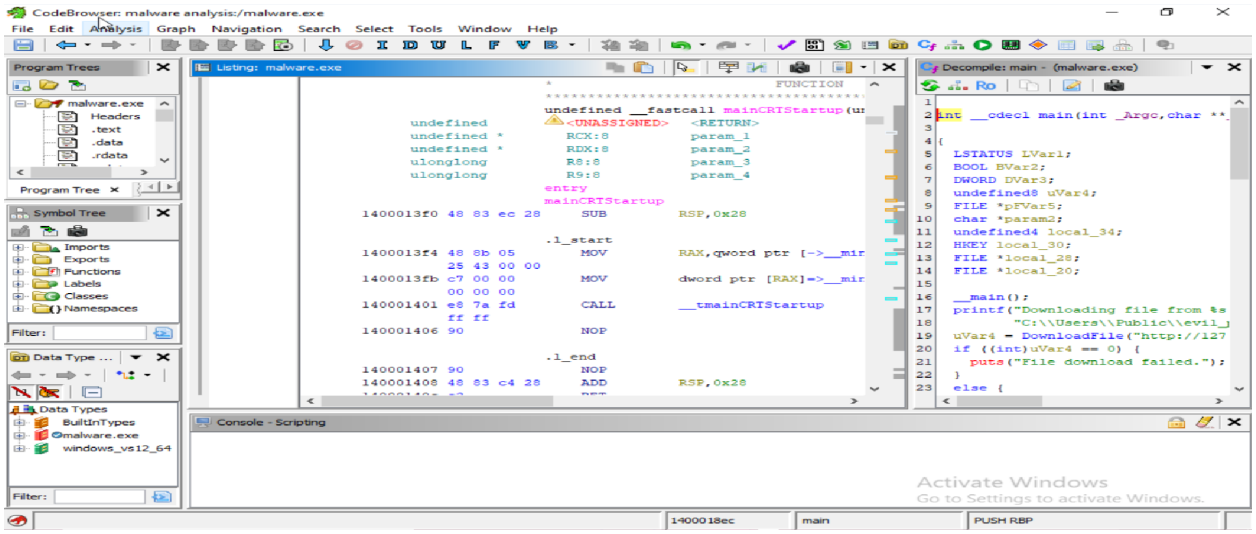
5.1 Ghidra Analysis (Disassembling Malware)

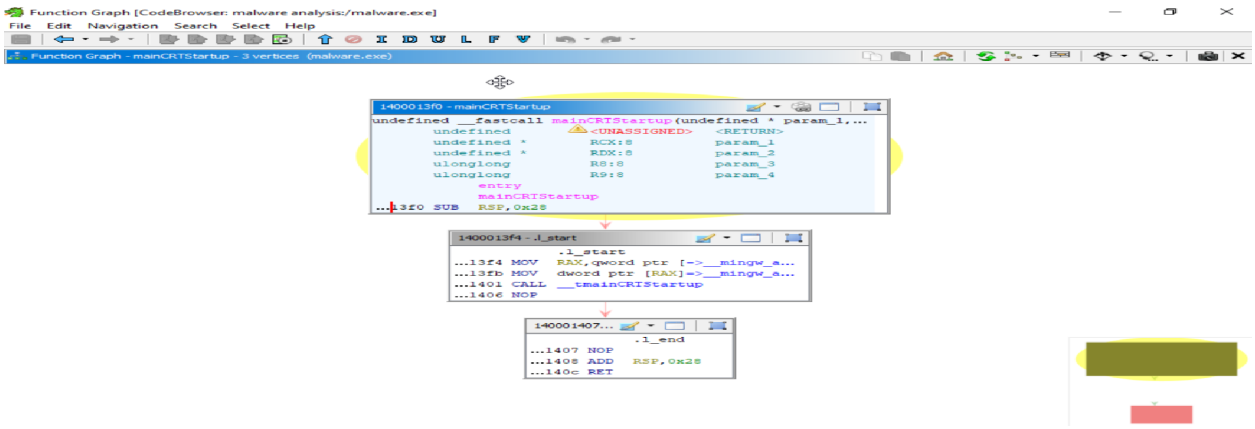
- Decompiled code matches behavior described in source
- Key functions: RegCreateKey, RegSetValueEx

Details of the imported file are shown here



This Screenshot shows the Ghidra windows





This Picture is showing the function graph in the listing window.

5.2. YARA Rules (Signature based Detections):

```
FLARE-VM Thu 05/15/2025  3:56:08.72
C:\malware>yara ..\YaraRules\MaliciousDownloader.yar malware.exe
```

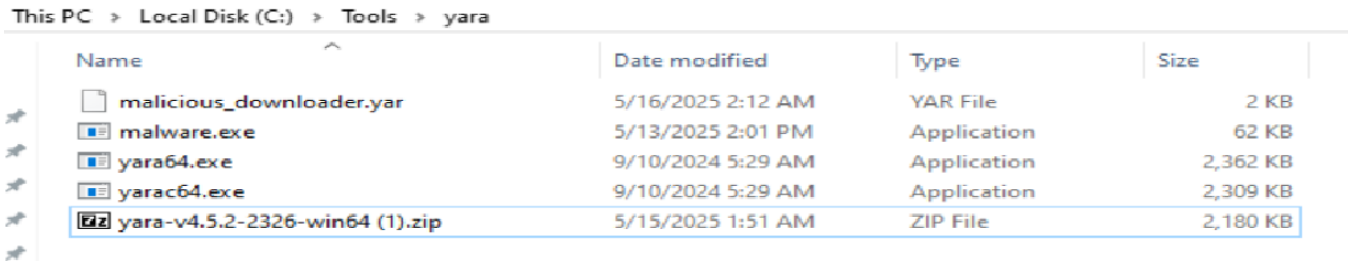
1st step: Downloaded the strings from .exe file (strings malware.exe > output.txt)

```
FLARE-VM 05/16/2025 02:18:06
PS C:\Windows\system32 > CD C:\Users\purna\malware.rep
FLARE-VM 05/16/2025 02:18:23
PS C:\Users\purna\malware.rep > strings malware.exe > output.txt
FLARE-VM 05/16/2025 02:18:53
```

*output.txt - Notepad
File Edit Format View Help

```
malicious_data
Registry value set: %s = %s
Failed to set registry value.
Failed to create/open registry key.
Deleting local files...
Deleted: %s
Failed to delete: %s (Error: %lu)
Argument domain error (DOMAIN)
Argument singularity (SIGN)
Overflow range error (OVERFLOW)
Partial loss of significance (PLOSS)
Total loss of significance (TLOSS)
The result is too small to be represented (UNDERFLOW)
Unknown error
_matherr(): %s in %s(%g, %g)  (retval=%g)
Mingw-w64 runtime failure:
Address %p has no image-section
  VirtualQuery failed for %d bytes at address %p
  VirtualProtect failed with code 0x%x
  Unknown pseudo relocation protocol version %d.
  Unknown pseudo relocation bit size %d.
%d bit pseudo relocation at %p out of range, targeting %p, yielding the value %p.
runtime error %d
GCC: (x86_64-posix-seh-rev0, Built by MinGW-Builds project) 13.2.0
RegCloseKey
RegCreateKeyExA
RegSetValueExA
DeleteCriticalSection
DeleteFileA
EnterCriticalSection
GetLastError
InitializeCriticalSection
LeaveCriticalSection
SetUnhandledExceptionFilter
Sleep
TlsGetValue
```

2nd step: With the above strings downloaded, able to create the Yara rules and saved as malicious_downloader.yar and extracted the latest yara-win64 zip file.




```
FLARE-VM 05/16/2025 02:15:15
PS C:\Users\purna\malware.rep > CD C:\Tools\yara
FLARE-VM 05/16/2025 02:17:47
PS C:\Tools\yara > .\yara64.exe malicious_downloader.yar malware.exe
>>
malicious_downloader_mingw malware.exe
```

As the Yara Rule matches, we got the output of the name of the rule and file name (As yara64.exe starts the Yara scanning engine, it loads all rules from malicious_downloader.yar., and it analyzes malware.exe using those rules).

Final Notes:

✓ **Component**	✓ **Observed Behaviors**
✓ Downloader	✓ Successful download via WinINet
✓ File system	✓ File creation and deletion
✓ Registry	✓ New key/values in HKCU
✓ Static detection	✓ No obfuscation; flagged as suspicious
✓ Dynamic detection	✓ File, registry, and HTTP activity confirmed
✓ Reverse engineering	✓ All malicious logic visible in decompiled code

✓ **Component**	✓ **Observed Behaviors**
✓ YARA rule	✓ Matches based on unique strings and APIs

6. Conclusion

This simulated malware was effective in demonstrating the behavior of typical downloader malware, incorporating real-world tactics like file manipulation, registry editing, and payload retrieval. The exercise enabled hands-on analysis using static and dynamic tools, and helped in crafting effective detection techniques using YARA rules. This forms a foundational understanding for identifying and analyzing real-world threats.

The simulated malware is very basic, lacking complexity, obfuscation, and evasion techniques of real threats. The analysis was likely shallow, focusing on immediate effects without deep behavioral or network inspection. The reliance on localhost and a controlled VM environment also simplifies the scenario compared to real-world infections. Finally, the YARA rule is likely basic and might not be robust against variations.