# Lab 2 – Malware Execution Monitoring Using ProcWatch, ProcMon and Macro Analysis

https://github.com/spectr33/MalwareAnalysis/blob/main/emotet.zip

Link for downloading ProWatch.

https://drive.google.com/file/d/1-WhOEL83h8OJ3XeBAMZdwMD0OKyFrzNh/view?usp=sharing

**Objective:**

- Learn how to detonate malware and monitor for dropped files and any changes in directories and their executions using ProcWatch and ProcMon.
- Learn how to analyse malicious macros using olevba to extract embedded VBA code and detect suspicious behaviour.
- Identify obfuscation techniques used in malicious macros, such as string manipulation, hex encoding, and Base64 encoding.

**Requirements-**

Flare VM

Microsoft Office: Microsoft prompts for sign-in or a product key. Click on "Product Key", then close the product key tab using the "Close Button" or "Close Icon" (X) in the top-right corner.

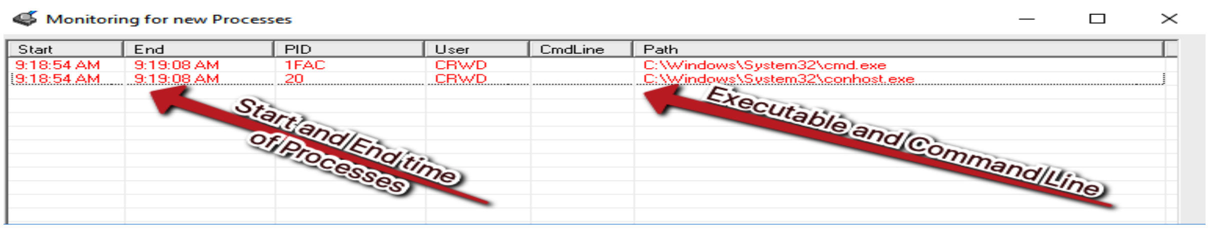
## Detonating your malware

In malware analysis, dynamic analysis is one of the most effective ways to observe how malware behaves when executed. By monitoring its activity in real-time, analysts can track process creation, file modifications, registry changes, and system interactions, helping to uncover its true functionality. This approach provides a clearer picture of how the malware operates and what it attempts to do on the system.

While static analysis, such as examining macros, can offer insights without execution, dynamic analysis allows us to see the malware's actual behavior in action. By using the right tools, we can detect hidden activities, identify malicious intent, and better understand the impact of the malware on the system.

## Monitoring for processes

In executing malware, it's important to realize that the binary file or scripted malware dropper that we are presented with as an initial vector of infection is rarely all there is to see. Often, the malware will create additional processes or executables that are not necessarily immediately apparent to the end user. Malware, as a rule, often performs many tasks that are invisible to the targeted user unless you are actively looking for these actions. To this end, there are several tools that are conducive to discovering these actions. The tool we will examine is **ProcWatch** and **ProcMon** in FLARE VM.


Monitoring for new Processes

| Start | End | PID | User | CmdLine | Path |
|---|---|---|---|---|---|
| 9:18:54 AM | 9:19:08 AM | 1FAC | CRWD | | C:\Windows\System32\cmd.exe |
| 9:18:54 AM | 9:19:08 AM | 20 | CRWD | | C:\Windows\System32\conhost.exe |

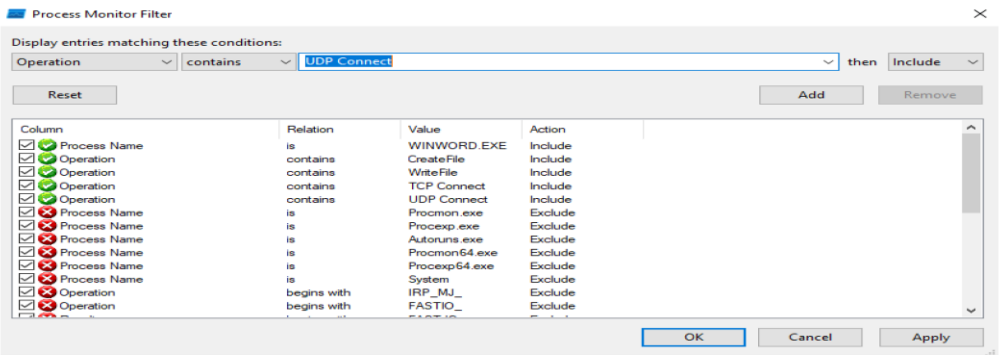
ProcWatch will monitor for new processes as they execute on the system and will inform us of their command-line arguments, as well as the user that ran them, and the start and end time of the processes. It's important to note that it will monitor for all new processes, not just ones related to malware, and as such, is prone to collecting noise from Windows' normal background processes.

Let's take a look at a sample piece of malware – an Emotet malicious document:

Proceed with the following steps –

- ProcMon creates a lot of noise, so set filters to track only winword.exe before running the emotet document. Additionally, you can filter for specific actions like CreateFile, WriteFile, and TCP/UDP connections to focus on potential malicious activity.

- Example Filters in ProcMon:



Display entries matching these conditions:
Operation | contains | UDP Connect | then | Include

| Column | Relation | Value | Action |
|---|---|---|---|
| ✓ Process Name | is | WINWORD.EXE | Include |
| ✓ Operation | contains | CreateFile | Include |
| ✓ Operation | contains | WriteFile | Include |
| ✓ Operation | contains | TCP Connect | Include |
| ✓ Operation | contains | UDP Connect | Include |
| ✓ Process Name | is | Procmon.exe | Exclude |
| ✓ Process Name | is | Procexp.exe | Exclude |
| ✓ Process Name | is | Autoruns.exe | Exclude |
| ✓ Process Name | is | Procmon64.exe | Exclude |
| ✓ Process Name | is | Procexp64.exe | Exclude |
| ✓ Process Name | is | System | Exclude |
| ✓ Operation | begins with | IRP_MJ_ | Exclude |
| ✓ Operation | begins with | FASTIO_ | Exclude |

- Start monitoring with the ProcWatch/ProcMon interface and open the Emotet document.
- Once the document has been opened, it may ask permissions for 'enable content' and you may proceed by enabling this option.

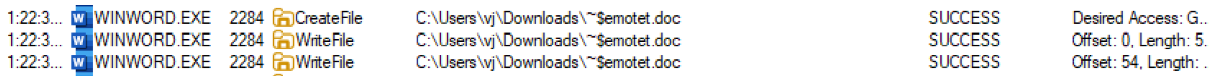
Question 1: After the document has been opened, have you noticed any new files dropped in any directories (including current directory).
A hidden emotet.doc file should be created in the same directory which is malicious.

- After the document has been opened, view the ProcWatch interface once again, followed by clicking on the 'Enable Editing' option which will further execute any malicious instructions embedded within the document.

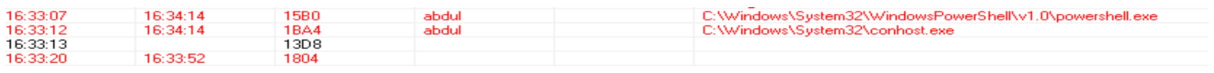
Question 2: Can you report any processes that were triggered under the current user in ProcWatch right after the above step.

A powershell.exe program would've been run along with the conhost.exe program. Some noise can be reported but these two programs are consistent in be run right after malware execution.

You may search for similar findings in your monitoring interfaces.



```
1:22:3...  WINWORD.EXE  2284  CreateFile  C:\Users\wj\Downloads\~$emotet.doc   SUCCESS  Desired Access: G..
1:22:3...  WINWORD.EXE  2284  WriteFile   C:\Users\wj\Downloads\~$emotet.doc   SUCCESS  Offset: 0, Length: 5.
1:22:3...  WINWORD.EXE  2284  WriteFile   C:\Users\wj\Downloads\~$emotet.doc   SUCCESS  Offset: 54, Length: .
```



```
16:33:07   16:34:14   15B0   abdul      C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
16:33:12   16:34:14   1BA4   abdul      C:\Windows\System32\conhost.exe
16:33:13              13D8
16:33:20   16:33:52   1804
```

Optional Question 3:
While analyzing Emotet execution in ProcMon or Wireshark (Ethernet-Adapter), did you observe any TCP or UDP connections initiated from the system?

**Extracting Macros for Analysis:**

- Emotet.doc relies on “macro execution” to run malicious commands. To analyze these macros, we will use olevba, a tool that extracts and displays VBA code from Word documents.

Steps to analyse emotet Macros:

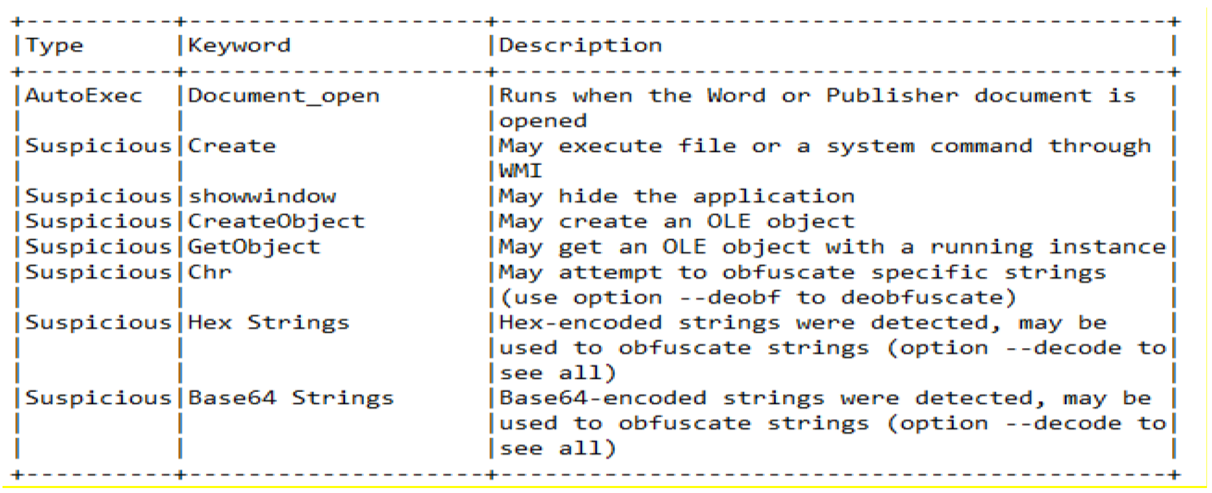
1. Open Command Prompt as Administrator.
2. Navigate to the folder where emotet.doc is located.
3. Run the following command to extract macros

Commands:

- olevba emotet.doc > emotet_macros.txt
- notepad emotet_macros.txt

Question 4: Based on the extracted VBA code, Provide a screenshot for the suspicious indicator table. identify at least three techniques used by the macro to evade detection and execute potentially malicious code?

Answer:



```
+----------+--------------------+---------------------------------------------+
|Type      |Keyword             |Description                                  |
+----------+--------------------+---------------------------------------------+
|AutoExec  |Document_open       |Runs when the Word or Publisher document is  |
|          |                    |opened                                       |
|Suspicious|Create              |May execute file or a system command through |
|          |                    |WMI                                          |
|Suspicious|showwindow          |May hide the application                     |
|Suspicious|CreateObject        |May create an OLE object                     |
|Suspicious|GetObject           |May get an OLE object with a running instance|
|Suspicious|Chr                 |May attempt to obfuscate specific strings    |
|          |                    |(use option --deobf to deobfuscate)          |
|Suspicious|Hex Strings         |Hex-encoded strings were detected, may be    |
|          |                    |used to obfuscate strings (option --decode to|
|          |                    |see all)                                     |
|Suspicious|Base64 Strings      |Base64-encoded strings were detected, may be |
|          |                    |used to obfuscate strings (option --decode to|
|          |                    |see all)                                     |
+----------+--------------------+---------------------------------------------+
```

Example answer:

Auto-Execution (AutoExec & Document_Open)

The macro runs automatically when the document opens, executing commands without user action.

Obfuscation (String Manipulation & Encoding)

The macro hides malicious commands using Chr, Hex encoding, or Base64 encoding.

System Interaction (CreateObject & GetObject)

The macro executes system commands by creating objects like WScript.Shell or PowerShell.

Question 5: Since Emotet uses macros for execution, what mitigations can be implemented in Windows to prevent the execution of such .doc-based attacks?

Use Protected view and disabled macros in Microsoft Word.