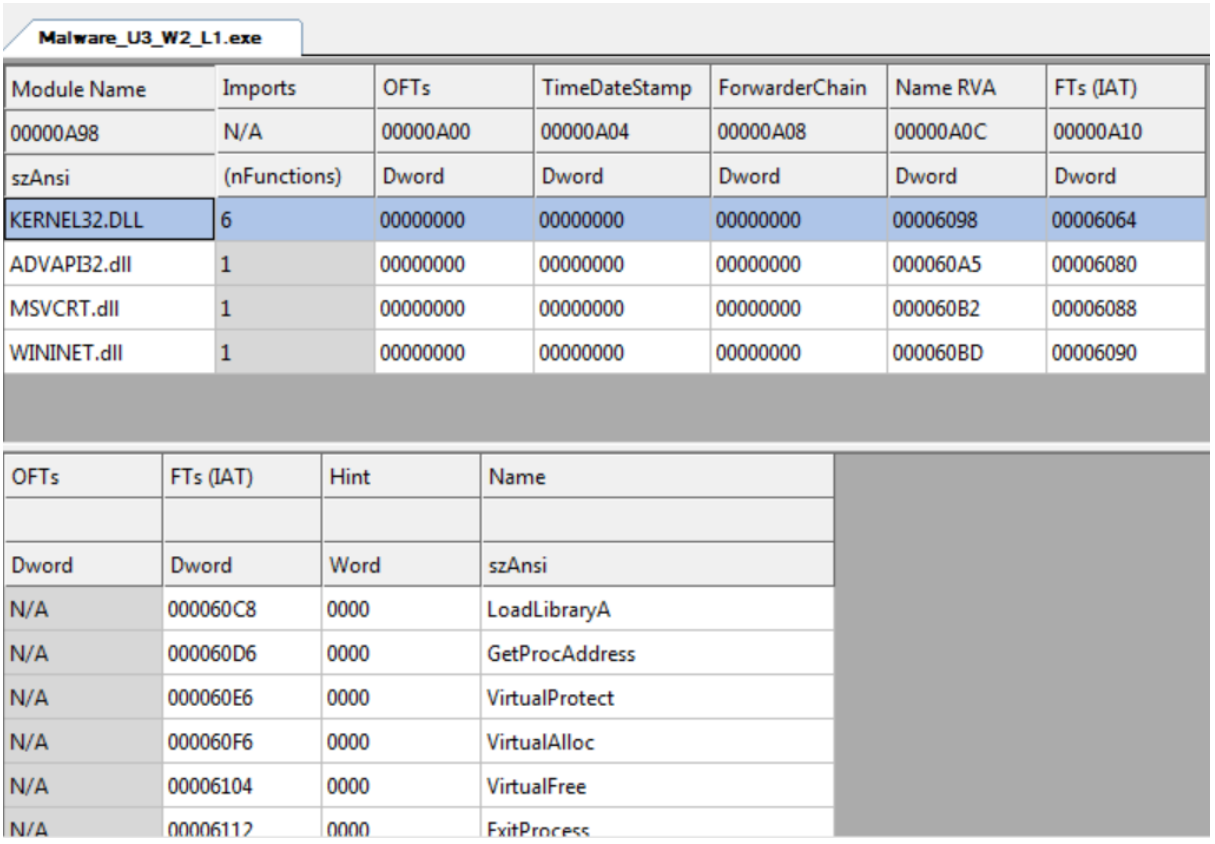
Basic Static Malware Analysis

We'll use **CFF Explorer** to run a basic static analysis on the malware **Malware_U3_W2_L1.**

First thing first, we're going to open the file and select the "import directory" section in order to see the libraries imported by the malware.



Malware_U3_W2_L1.exe

Module Name	Imports	OFTs	TimeDateStamp	ForwarderChain	Name RVA	FTs (IAT)
00000A98	N/A	00000A00	00000A04	00000A08	00000A0C	00000A10
szAnsi	(nFunctions)	Dword	Dword	Dword	Dword	Dword
KERNEL32.DLL	6	00000000	00000000	00000000	00006098	00006064
ADVAPI32.dll	1	00000000	00000000	00000000	000060A5	00006080
MSVCRT.dll	1	00000000	00000000	00000000	000060B2	00006088
WININET.dll	1	00000000	00000000	00000000	000060BD	00006090

OFTs	FTs (IAT)	Hint	Name
Dword	Dword	Word	szAnsi
N/A	000060C8	0000	LoadLibraryA
N/A	000060D6	0000	GetProcAddress
N/A	000060E6	0000	VirtualProtect
N/A	000060F6	0000	VirtualAlloc
N/A	00006104	0000	VirtualFree
N/A	00006112	0000	ExitProcess

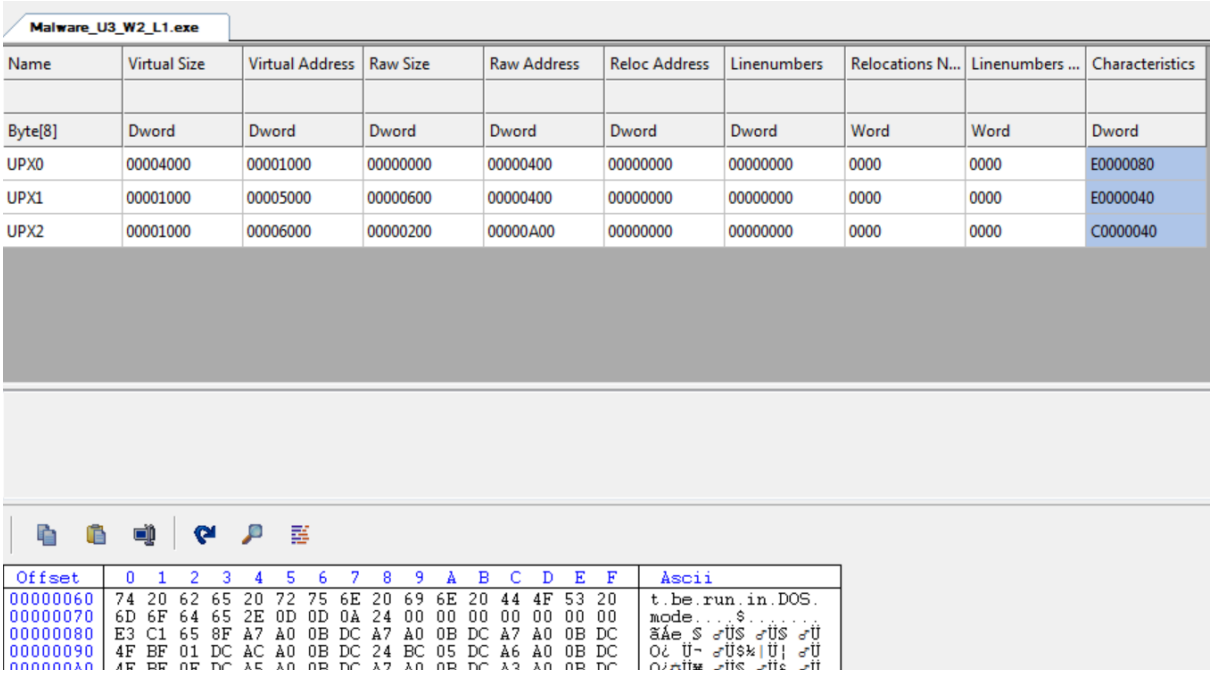
As we can see, the libraries imported are four:

- **Kernel32.dll**: Manages core system functions like process creation, memory management, and file operations.
- **Advapi32.dll**: Handles advanced system services, including security, user accounts, and registry management.

- **MSVCRT.dll**: The Microsoft C Runtime Library, which provides standard C library functions for programs written in C and C++.
- **Wininet.dll**: A Windows library that provides functions for accessing Internet protocols such as HTTP and FTP.

This four libraries manage key foundations of software, so we can infer that the malware is quite advanced, We can see that it imports the "**LoadLibraryA**" and the "**GetProcAddress**" functions from the Kernel32.dll library. These functions are often found when a malware imports libraries in runtime, which helps in hiding what libraries are imported.

Next, we'll head over the section header tab.



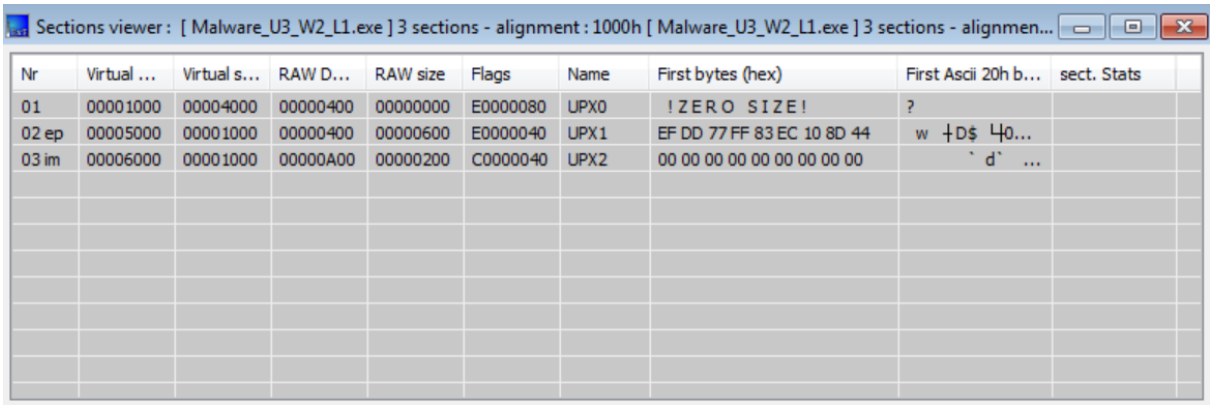
Here we can see that the malware is divided into three main section: UPX0, UPX1 and UPX2.

These are fake names, used by the malware to hide the true names of the sections in order to slow the analysis process down.

We cannot understand the true nature of these sections with a basic static analysis but, as we stated above, we can infer that this malware is requires a lot of studying through other tools and methods,

We can see all of this info with another tool: **ExeInfoPE.**

We'll load the malware and use the button **Sections Viewer**



We find the same informations we retrieved using CFF Extractor.

Federico Biggi