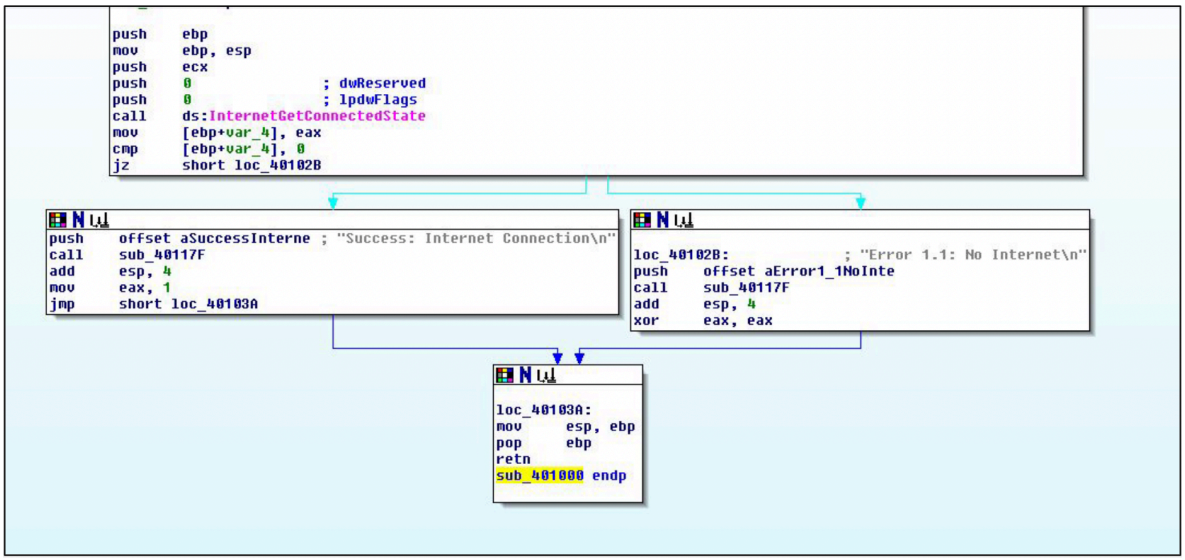
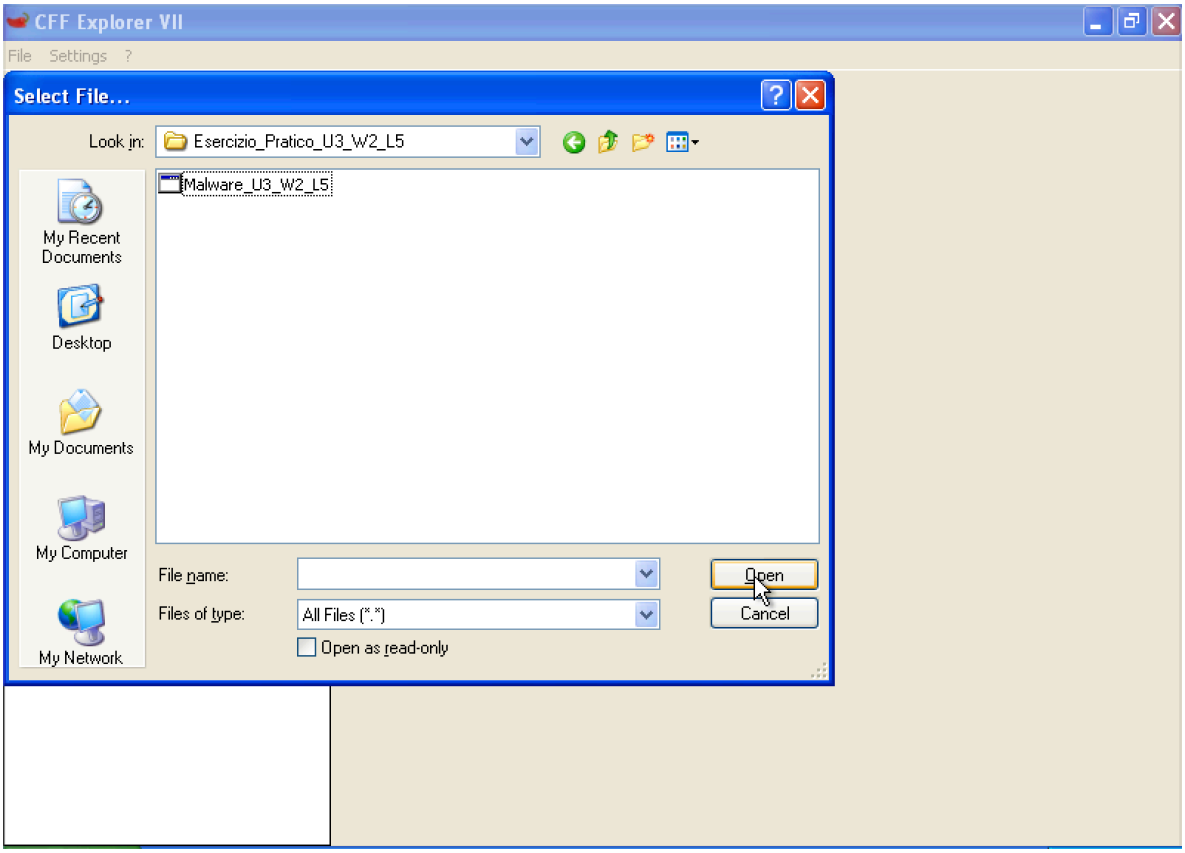
REPORT MALWARE ANALISYS

Traccia:

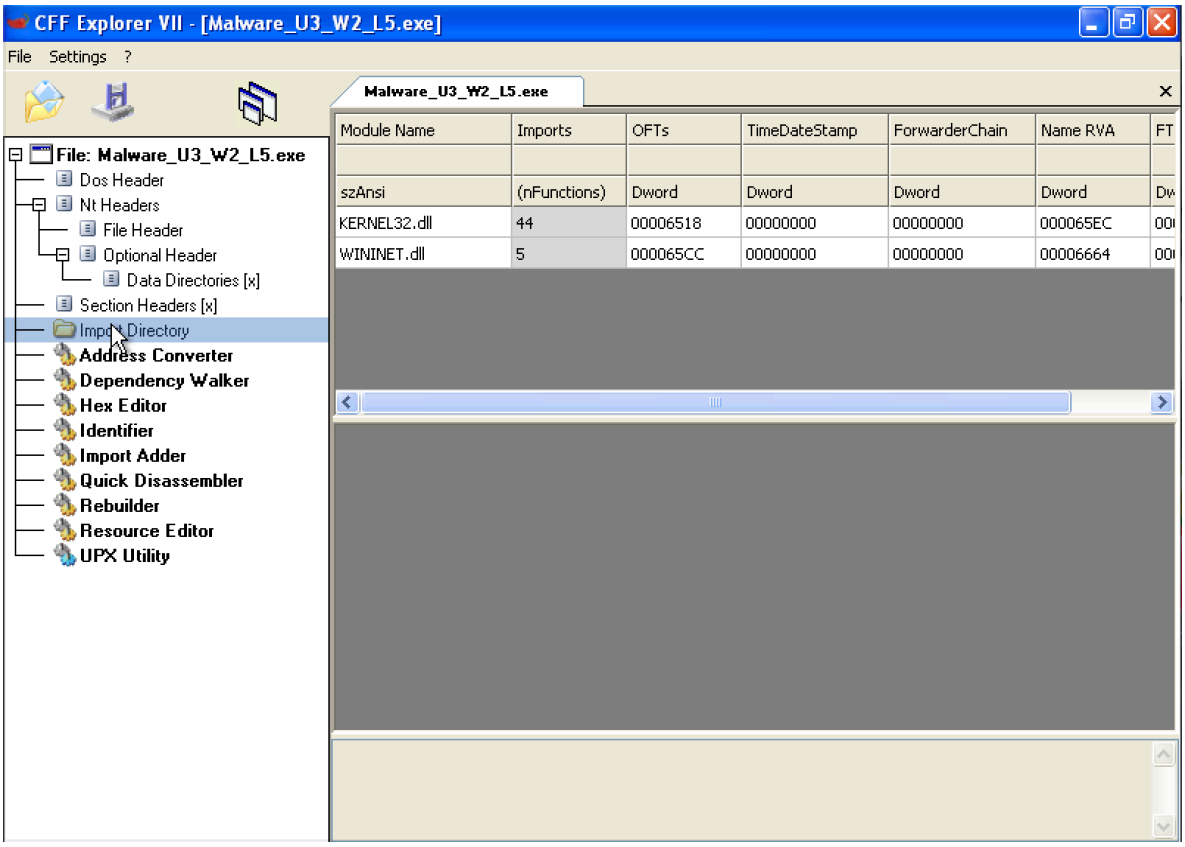
1. Analizzare file Malware_U3_W2_L5 presente nella nostra macchina virtuale, ed analizzare quali librerie e sezioni sono importate dal file eseguibile.
2. Data questa figura: analizzare e descrivere i costrutti noti ed analizzare eventuale comportamento.



Per prima cosa andiamo ad aprire il programma “CFF Explorer” presente sulla nostra macchina virtuale, ed andiamo a cercare e aprire il file Malware_U3_W2_L5.



Una volta all'interno del file andiamo a cercare quali sono le librerie importate dal Malware nella sezione "Import Directory".
Come possiamo vedere questo Malware importa due tipi di librerie:
-KERNEL32.dll, contenente 44 funzioni;
-WINNET.dll, contenente 5 funzioni;



La libreria KERNEL32.dll è un tipo di libreria piuttosto comune, contenente le principali funzioni principali per poter interagire col sistema operativo. Di fatto le sue 44 funzioni ce lo dimostrano.

000065E4	000065E4	0296	Sleep
00006940	00006940	027C	SetStdHandle
0000692E	0000692E	0156	GetStringTypeW
0000691C	0000691C	0153	GetStringTypeA
0000690C	0000690C	01C0	LCMapStringW
000068FC	000068FC	01BF	LCMapStringA
000068E6	000068E6	01E4	MultiByteToWideChar
00006670	00006670	00CA	GetCommandLineA
00006682	00006682	0174	GetVersion
00006690	00006690	007D	ExitProcess
0000669E	0000669E	029E	TerminateProcess
000066B2	000066B2	00F7	GetCurrentProcess
000066C6	000066C6	02AD	UnhandledExceptionFilter
000066E2	000066E2	0124	GetModuleFileNameA
000066F8	000066F8	00B2	FreeEnvironmentStringsA
00006712	00006712	00B3	FreeEnvironmentStringsW
0000672C	0000672C	02D2	WideCharToMultiByte

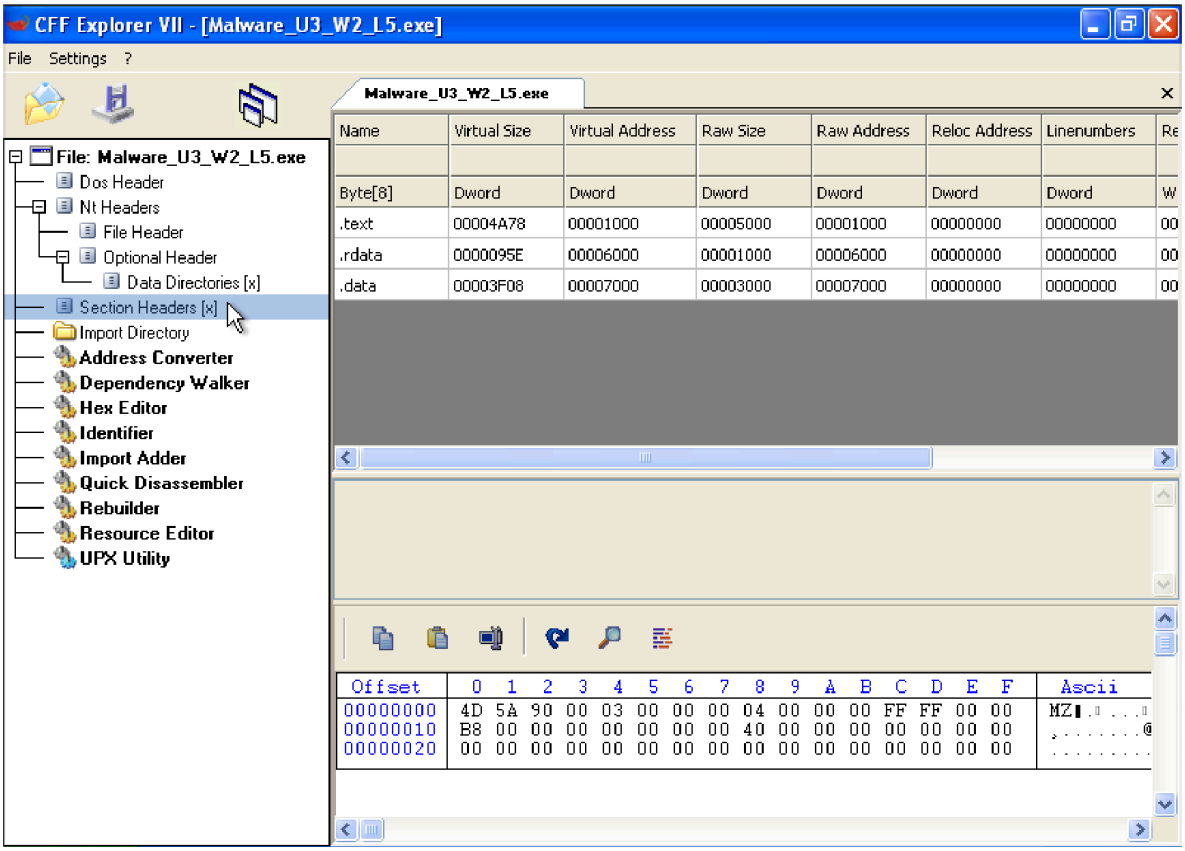
00006742	00006742	0106	GetEnvironmentStrings
0000675A	0000675A	0108	GetEnvironmentStringsW
00006774	00006774	026D	SetHandleCount
00006786	00006786	0152	GetStdHandle
00006796	00006796	0115	GetFileType
000067A4	000067A4	0150	GetStartupInfoA
000067B6	000067B6	0126	GetModuleHandleA
000067CA	000067CA	0109	GetEnvironmentVariableA
000067E4	000067E4	0175	GetVersionExA
000067F4	000067F4	019D	HeapDestroy
00006802	00006802	019B	HeapCreate
00006810	00006810	02BF	VirtualFree
0000681E	0000681E	019F	HeapFree
0000682A	0000682A	022F	RtlUnwind
00006836	00006836	02DF	WriteFile
00006842	00006842	0199	HeapAlloc
0000684E	0000684E	00BF	GetCPInfo

0000685A	0000685A	00B9	GetACP
00006864	00006864	0131	GetOEMCP
00006870	00006870	02BB	VirtualAlloc
00006880	00006880	01A2	HeapReAlloc
0000688E	0000688E	013E	GetProcAddress
000068A0	000068A0	01C2	LoadLibraryA
000068B0	000068B0	011A	GetLastError
000068C0	000068C0	00AA	FlushFileBuffers
000068D4	000068D4	026A	SetFilePointer
00006950	00006950	001B	CloseHandle

Mentre per quanto riguarda la libreria WININET.dll contiene le funzioni per l'implementazione di alcuni protocolli di rete come HTTP, FTP, NTP, e le funzioni ad essa associate sono:

00006640	00006640	0071	InternetOpenUrlA
0000662A	0000662A	0056	InternetCloseHandle
00006616	00006616	0077	InternetReadFile
000065FA	000065FA	0066	InternetGetConnectedState
00006654	00006654	006F	InternetOpenA

Per analizzare le sezioni di cui è composto il file, andiamo nella sezione "Section Headers"



Riscontrando che il malware è composto da Headers di tipo:

- .text
- .rdata
- .data

Malware_U3_W2_L5.exe

Name	Virtual Size	Virtual Address	Raw Size	Raw Address	Reloc Address	Linenumbers	Relocations ...	Linenumber...	Characteristics
Byte[8]	Dword	Dword	Dword	Dword	Dword	Dword	Word	Word	Dword
.text	00004A78	00001000	00005000	00001000	00000000	00000000	0000	0000	60000020
.rdata	0000095E	00006000	00001000	00006000	00000000	00000000	0000	0000	40000040
.data	00003F08	00007000	00003000	00007000	00000000	00000000	0000	0000	C0000040

Ora passiamo ad analizzare la figura contenete il codice in Assembly.
Da un attenta analisi possiamo dire che il codice sia composto da 6 costrutti noti e sono:

- push ebp
  mov ebp, esp

Con queste due stringhe di comando andiamo a creare lo Stack.

- push ecx
  push 0 ;dwReserved
  push 0. ;lpdwFlags
  call ds:InternetGetConnectedState

Attraverso questi tre parametri viene richiamata la funzione “InternetGetConnectedState”, la quale serve per controllare se il dispositivo è

connesso alla rete.

- cmp [ebp+var_4], 0
  jz short loc_40102B

Queste due stringhe rappresentano un “if” le quali controllano che il risultato sia 0, ed in caso contrario salta fino all’indirizzo di memoria scritto per continuare il codice.

- push offsetaSuccessInterne ;”Success: Internet Connection\n”
  call sub_40117F

Tramite questa chiamata possiamo ipotizzare che essa richiami la funzione printf per stampare la stringa Success: Internet Connection.

- push offset aError1_1NoInternet ;”Success: Internet Connection\n”
  Call sub_40117F

Tramite questa chiamata possiamo ipotizzare che la funzione printf venga chiamata per stampare la stringa “Error 1_1: NoInternet”.

- mov esp, ebp
  pop ebp

Attraverso queste due stringhe di comando si rimuove lo Stack.