

Analisi del malware

Controllare quali sono le librerie e le funzioni importate è fondamentale per capire lo scopo del malware infatti per rispondere ai primi due punti richiesti dal progetto utilizziamo il tool CFF EXPLORER che ci aiuta a controllare le funzioni importate ed esportate da un malware .

Il primo passaggio che facciamo è aprire CFF EXPLORER, clicchiamo sull' icona a forma di cartella e scegliamo il file da caricare in questo caso "**Malware_U3_W2_L5**"



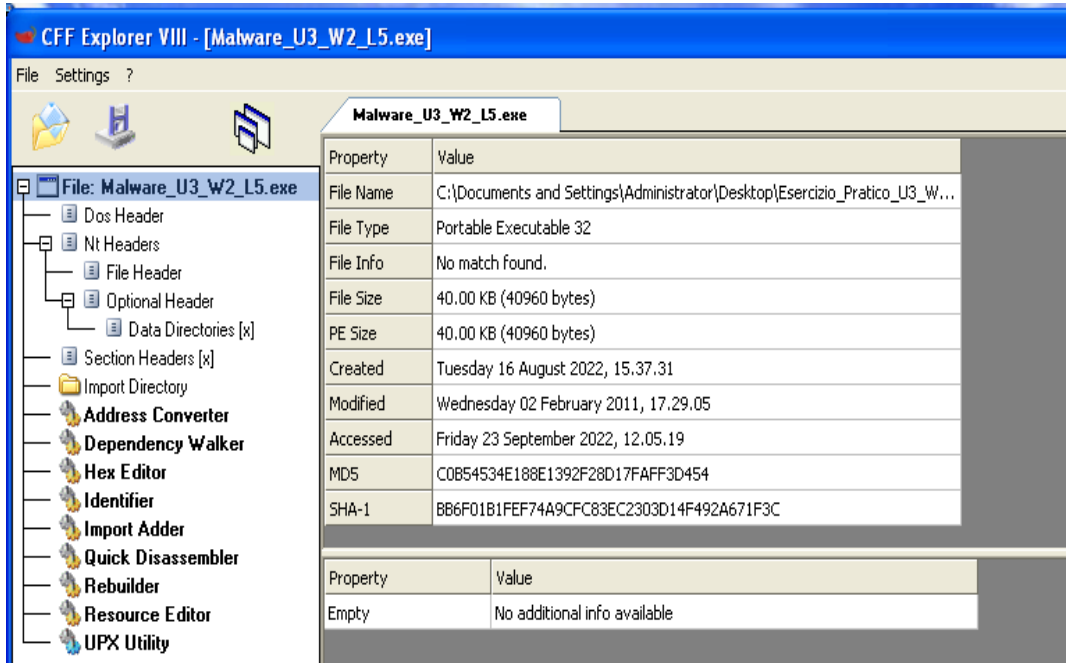
Una volta aperto il file,vediamo una descrizione generale del **malware**(img.1) .

Quello che ci interessa a noi e vedere le librerie e le funzioni importate , per questo ci digiriamo nella parte "**Import directory**" del menu a sinistra(img.2). Nella parte a destra ci vengono mostrate le informazioni sulle librerie importate. In questo caso abbiamo :

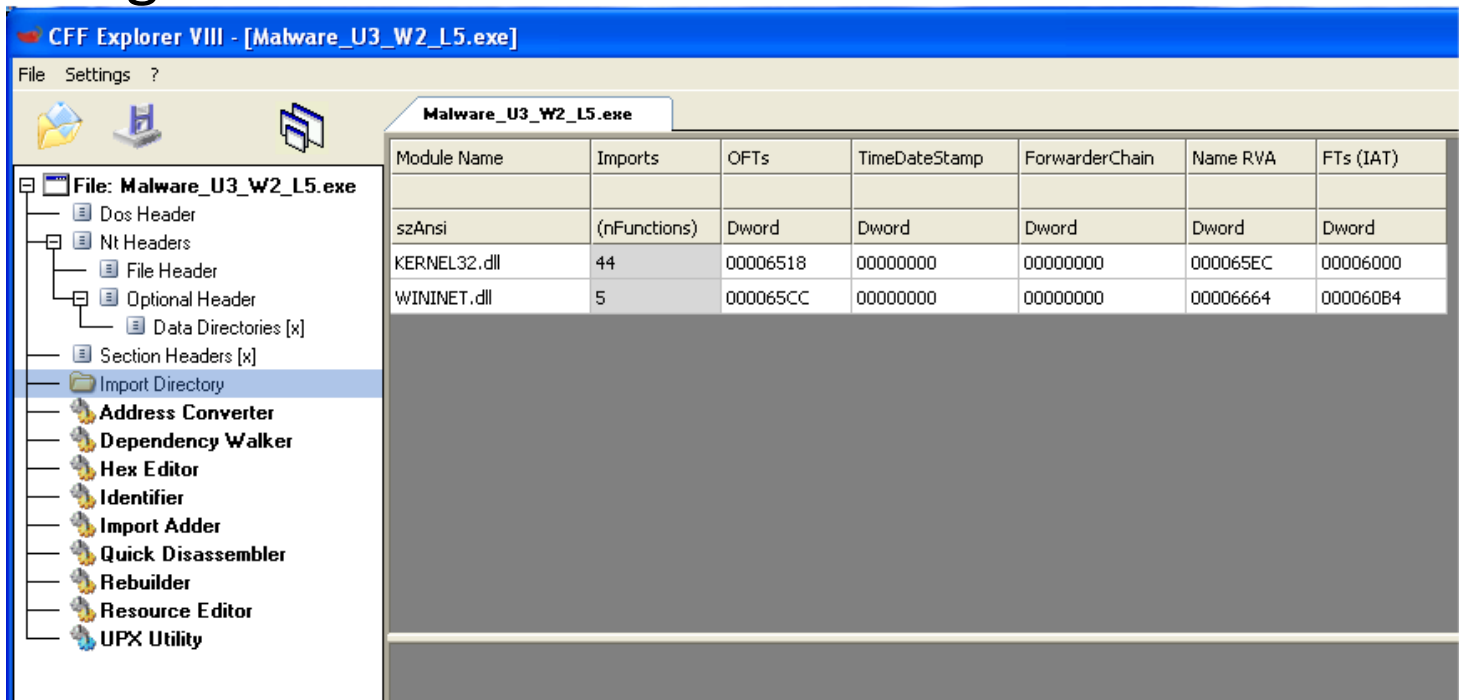
KERNEL32.dll - libreria comune che contiene funzioni principali per la manipolazione dei file ,la gestione della memoria.

WININET.dll - libreria che contiene le funzioni per l'implementazione di alcuni protocolli di rete come HTTP, NTP, FTP

• Img.1



• Img.2



Il secondo punto ci chiede le sezioni di cui si compone il malware , per questo sempre dal menu a sinistra andiamo nella sezione "Section headers",il quale ci mostra le informazioni circa le sezioni di cui si compone l'eseguibile, in questo caso è fromato da tre sezioni .

.TEXT - contiene le righe di codice che la CPU eseguirà una volta avviato il software

.RDATA - include le informazioni circa le librerie e funzioni importate

.DATA - contiene dati/variabili globali del programma eseguibile.

CFF Explorer VIII - [Malware_U3_W2_L5.exe]

File Settings ?

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

File: Malware_U3_W2_L5.exe

- Dos Header
- Nt Headers
 - File Header
 - Optional Header
 - Data Directories [x]
- Section Headers [x]
- Import Directory
- Address Converter
- Dependency Walker
- Hex Editor
- Identifier
- Import Adder
- Quick Disassembler
- Rebuilder
- Resource Editor
- UPX Utility

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	Ascii
00000000	4D	5A	90	00	03	00	00	00	04	00	00	00	FF	FF	00	00	MZÿ...
00000010	B8	00	00	00	00	00	00	00	40	00	00	00	00	00	00	00@.....
00000020	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000030	00	00	00	00	00	00	00	00	00	00	00	00	E8	00	00	00ë.....
00000040	0E	1F	BA	0E	00	B4	09	CD	21	B8	01	4C	CD	21	54	68	!!? ... ! !Th
00000050	69	73	20	70	72	6F	67	72	61	6D	20	63	61	6E	6E	6F	is program.canno
00000060	74	20	62	65	20	72	75	6E	20	69	6E	20	44	4F	53	20	t.be.run.in.DOS.
00000070	6D	6F	64	65	2E	0D	0D	0A	24	00	00	00	00	00	00	00	mode....\$.....
00000080	49	29	7E	26	0D	48	10	75	0D	48	10	75	0D	48	10	75	I)~&.H!u.H!u.H!u
00000090	3B	6E	1B	75	0C	48	10	75	8E	54	1E	75	03	48	10	75	;n!u H!u T!u H!u
000000A0	3B	6E	1A	75	20	48	10	75	0D	48	10	75	0A	48	10	75	;n!u.H!u.H!u.H!u

Linguaggio Assembly

```
push    ebp
mov     ebp, esp
push    ecx
push    0          ; dwReserved
push    0          ; lpdwFlags
call    ds:InternetGetConnectedState
mov     [ebp+var_4], eax
cmp     [ebp+var_4], 0
jz      short loc_40102B
```

```
push    offset aSuccessInterne ; "Success: Internet Connection\n"
call    sub_40117F
add     esp, 4
mov     eax, 1
jmp     short loc_40103A
```

```
loc_40102B:          ; "Error 1.1: No Internet\n"
push    offset aError1_1NoInte
call    sub_40117F
add     esp, 4
xor     eax, eax
```

```
loc_40103A:
mov     esp, ebp
pop     ebp
retn
sub_401000 endp
```

Creazione dello stack

Chiamata di funzione. I parametri sono passati sullo stack tramite le istruzioni push

Ciclo IF

```
push    ebp
mov     ebp, esp
push    ecx
push    0          ; dwReserved
push    0          ; lpdwFlags
call    ds:InternetGetConnectedState
mov     [ebp+var_4], eax
cmp     [ebp+var_4], 0
jz      short loc_40102B
```

Possibili output del ciclo IF, Se il valore di ritorno (return) della funzione è diverso da 0, allora vuol dire che c'è una connessione attiva.

Con il "push of set "ci dice che la connessione a internet è attiva

```
push offset aSuccessInterne ; "Success: Internet Connection\n"
call sub_40117F
add esp, 4
mov eax, 1
jmp short loc_40103A
```

Con il "push of set aError1"ci dice che la connessione non c'è

```
loc_40102B: ; "Error 1.1: No Internet\n"
push offset aError1_1NoInte
call sub_40117F
add esp, 4
xor eax, eax
```

Chiude il ciclo e ripulisce lo stack

```
loc_40103A:
mov esp, ebp
pop ebp
retn
sub_401000 endp
```

La funzionalità implementata

Il malware chiama la funzione "internetgetconnectedstate" e ne controlla con un «if» il valore di ritorno. Se il valore di ritorno (return) della funzione è diverso da 0, allora vuol dire che c'è una connessione attiva. Possiamo dunque dedurre che il malware in questione è un downloader perché tenta la connessione a internet per poi scaricare altri malware .