

A decorative graphic in the top-left corner consisting of a blue parallelogram and a light green parallelogram, both tilted at an angle. The background is a dark navy blue with faint, lighter blue diagonal stripes.

Esercizio S10 L5



Traccia:

Con riferimento al file Malware_U3_W2_L5 presente all'interno della cartella «Esercizio_Pratico_U3_W2_L5» sul desktop della macchina virtuale dedicata per l'analisi dei malware, rispondere ai seguenti quesiti:

Quali librerie vengono importate dal file eseguibile?

Quali sono le sezioni di cui si compone il file eseguibile del malware?

Con riferimento alla figura in slide 3, risponde ai seguenti quesiti:

Identificare i costrutti noti (creazione dello stack, eventuali cicli, costrutti).

Ipotizzare il comportamento della funzionalità implementata.

Svolgimento della parte prima della traccia:

Tutto il materiale necessario per svolgere l'esercizio di oggi è contenuto nella macchina virtuale fornita all'inizio della settimana, perciò utilizzeremo tale macchina.









Sul desktop della macchina è presente una cartella denominata MALWARE che contiene il malware che andremo ad analizzare.



Svolgimento della parte prima della traccia:

All'interno della cartella sono presenti altre sottocartelle che contengono vari malware, per l'esercizio di oggi analizzeremo il malware contenuto nella cartella Esercizio_Pratico_U3_W2_L5.

Nome	Ultima modifica	Tipo
 Build_Week_Unit_3	17/01/2024 17:48	Cartella di file
 Esercizio_Pratico_U3_W2_L1	06/02/2024 11:24	Cartella di file
 Esercizio_Pratico_U3_W2_L2	17/01/2024 17:48	Cartella di file
 Esercizio_Pratico_U3_W2_L5	06/02/2024 11:24	Cartella di file
 Esercizio_Pratico_U3_W3_L2	17/01/2024 17:48	Cartella di file
 Esercizio_Pratico_U3_W3_L3	17/01/2024 17:48	Cartella di file



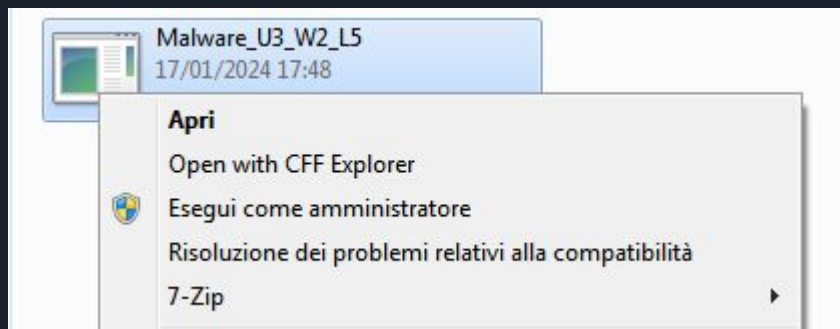
Malware_U3_W2_L5

17/01/2024 17:48

40,0 KB

Svolgimento della parte prima della traccia:

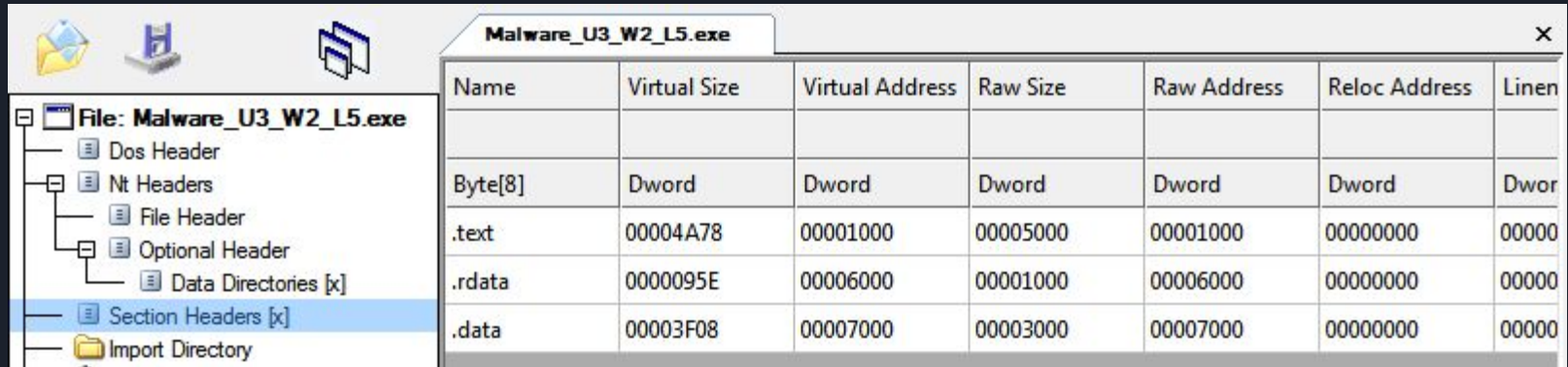
Apriamo il file con CFF Explorer, un software gratuito di PE editing e process viewing creato da Erik Pistrelli per l'analisi dei file .exe



Viene chiesto di scoprire quali librerie vengono importate dal file eseguibile e quali sono le sezioni di cui si compone il file eseguibile del malware, per recuperare queste informazioni esamineremo le sezioni Section Header e Import Directory del software.

Svolgimento della parte prima della traccia:

Apriamo la sezione Section Header e vediamo cosa ci viene mostrato:



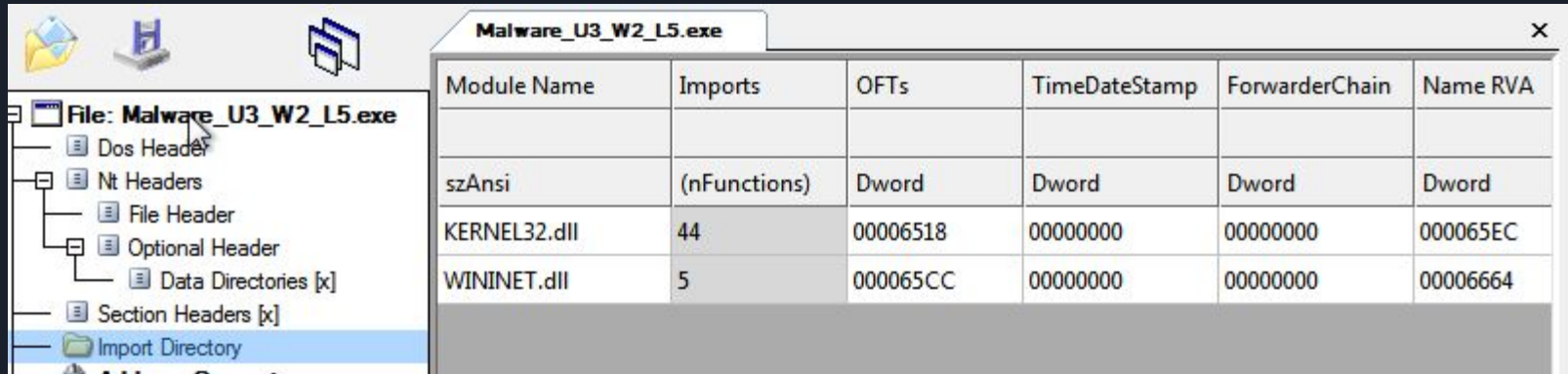
Name	Virtual Size	Virtual Address	Raw Size	Raw Address	Reloc Address	Linen
Byte[8]	Dword	Dword	Dword	Dword	Dword	Dwor
.text	00004A78	00001000	00005000	00001000	00000000	00000
.rdata	0000095E	00006000	00001000	00006000	00000000	00000
.data	00003F08	00007000	00003000	00007000	00000000	00000

Da questa schermata è possibile osservare che il malware in questione è composto dalle seguenti tre sezioni:

- .text
- .rdata
- .data

Svolgimento della parte prima della traccia:

Apriamo la sezione Import Directory del software e vediamo cosa ci viene mostrato:

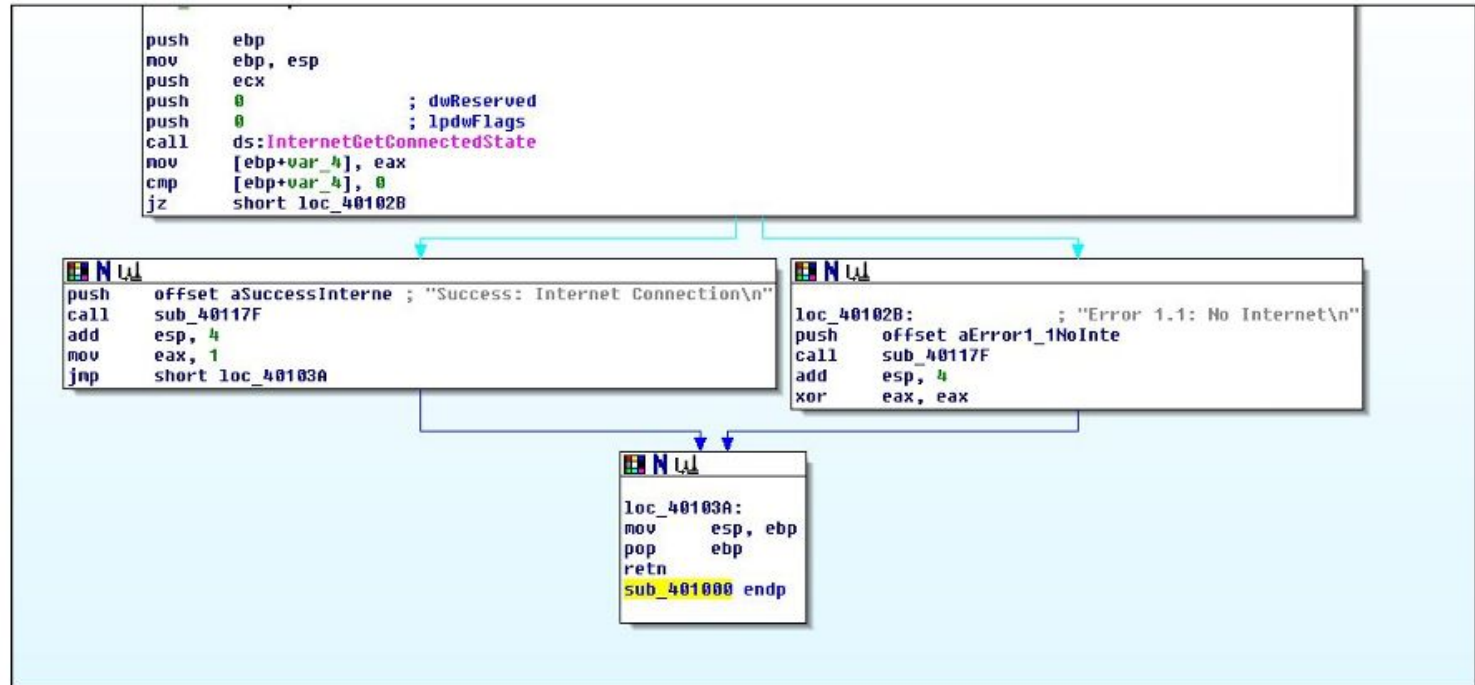


Module Name	Imports	OFTs	TimeDateStamp	ForwarderChain	Name RVA
szAnsi	(nFunctions)	Dword	Dword	Dword	Dword
KERNEL32.dll	44	00006518	00000000	00000000	000065EC
WININET.dll	5	000065CC	00000000	00000000	00006664

Da questa schermata possiamo vedere come il malware carichi due librerie, KERNEL32.dll e WININET32.dll, ed in particolare il malware richiama 44 funzioni diverse da KERNEL32.dll e 5 funzioni da WININET.dll

Svolgimento della seconda parte di traccia:

Figura 1



Svolgimento della seconda parte di traccia:

La prima parte di codice contiene le istruzioni che creano lo stack

```
push    ebp
mov     esp, ebp
```

Questa sezione del codice contiene le istruzioni che contraddistinguono il costrutto condizionale <IF>

```
cmp     [ebp+var_4], 0
jz      short loc_401028
```

In questa sezione finale del codice è contenuto il set di istruzioni che elimina lo stack

```
mov     esp, ebp
pop     ebp
```



Svolgimento della seconda parte di traccia:

Possiamo osservare che il codice mostrato richiama la funzione `InternetGetConnectionState`, che è una funzione che controlla se sia presente una connessione internet attiva, a seguito fa partire un controllo condizionale. Se trova una connessione stampa a schermo il seguente messaggio “Success: Internet Connection” mentre se la connessione non viene rilevata stampa a schermo il seguente messaggio “Error 1.1: No Internet”.