

**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:

### 1. Quali librerie vengono importate dal file eseguibile?

Il sistema operativo Windows utilizza principalmente il formato **PE (Portable Executable)** per i file eseguibili. Questo formato include informazioni essenziali per il sistema operativo al fine di gestire il codice del file, come le librerie. Le **librerie**, o moduli, contengono un insieme di funzioni. Quando un programma necessita di una funzione, richiama una libreria in cui è definita tale funzione. Questo processo è noto come importazione di una libreria.

Le librerie e le funzioni possono essere importate in tre modi diversi dagli eseguibili:

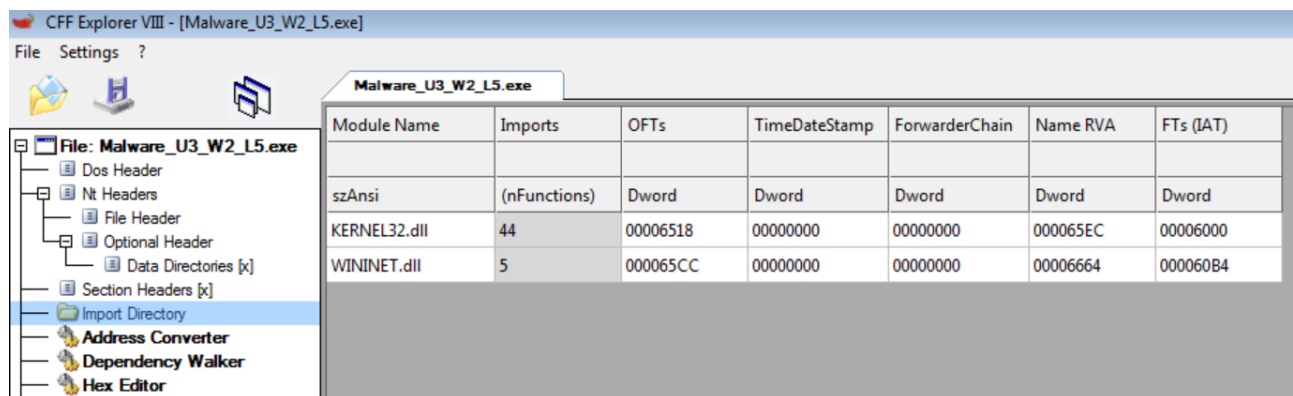
**Importazione statica:** in questo caso, l'eseguibile incorpora completamente il contenuto della libreria nel proprio codice. Questo approccio aumenta la dimensione del file e complica l'analisi statica avanzata, poiché è difficile distinguere il codice della libreria dal codice dell'eseguibile.

**Importazione a tempo di esecuzione (runtime):** in questo caso, l'eseguibile richiama la libreria solo quando ha bisogno di una specifica funzione. Questo comportamento è comunemente utilizzato dai malware per ridurre la loro rilevanza e invasività. Per chiamare la libreria solo quando necessario, vengono utilizzate funzioni fornite dal sistema operativo come "LoadLibrary" e "GetProcAddress".

**Importazione dinamica:** questo è il metodo più interessante per gli analisti di sicurezza ed è anche il più diffuso. Le librerie importate dinamicamente vengono caricate dal sistema operativo quando l'eseguibile viene avviato. Le funzioni all'interno della libreria vengono chiamate ed eseguite solo quando necessario.

Per vedere le librerie importate dal malware, utilizzo **“CFF Explorer”**, un software utilizzato principalmente per l'analisi e la modifica di file eseguibili di Windows.

Le librerie importate dal malware «Esercizio\_Pratico\_U3\_W2\_L5» sono le seguenti:

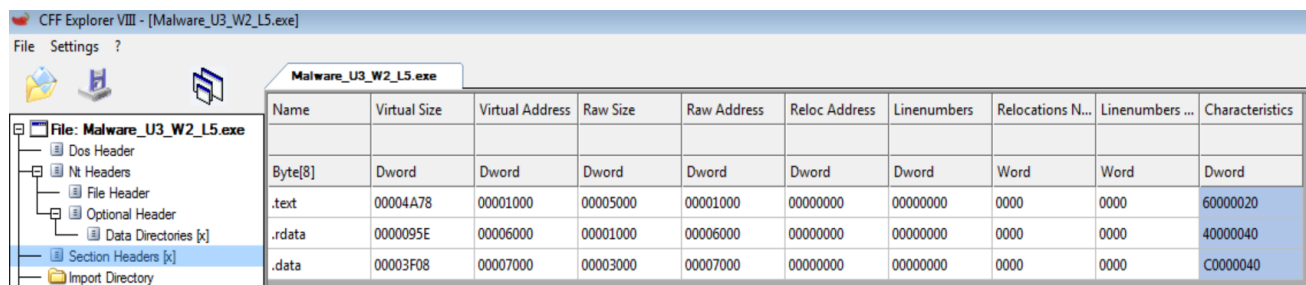


Module Name	Imports	OFTs	TimeStamp	ForwarderChain	Name RVA	FTs (IAT)
szAnsi	(nFunctions)	Dword	Dword	Dword	Dword	Dword
KERNEL32.dll	44	00006518	00000000	00000000	000065EC	00006000
WININET.dll	5	000065CC	00000000	00000000	00006664	000060B4

- **Kernel32.dll:** contiene le funzioni principali per interagire con il sistema operativo, ad esempio: manipolazione dei file, la gestione della memoria.
- **Wininet.dll:** contiene le funzioni per l'implementazione di alcuni protocolli di rete come HTTP, FTP, NTP.

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

Utilizzando sempre “CFF Explorer”, le sezioni di cui si compone il malware sono le seguenti:

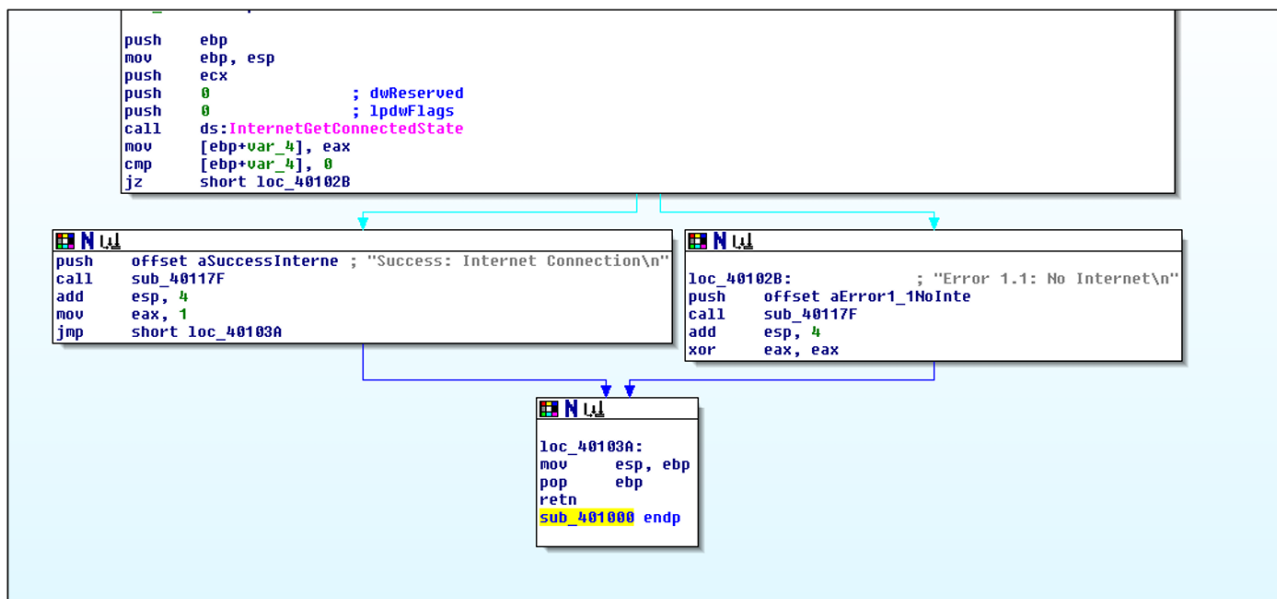


The screenshot shows the CFF Explorer interface. On the left, a tree view displays the file structure: File: Malware\_U3\_W2\_L5.exe, Dos Header, Nt Headers, File Header, Optional Header, Data Directories [x], Section Headers [x], and Import Directory. The 'Section Headers [x]' is selected. The main pane shows a table of sections for 'Malware\_U3\_W2\_L5.exe'.

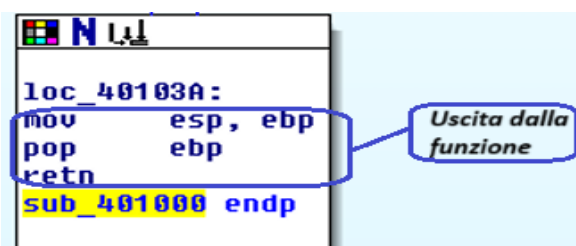
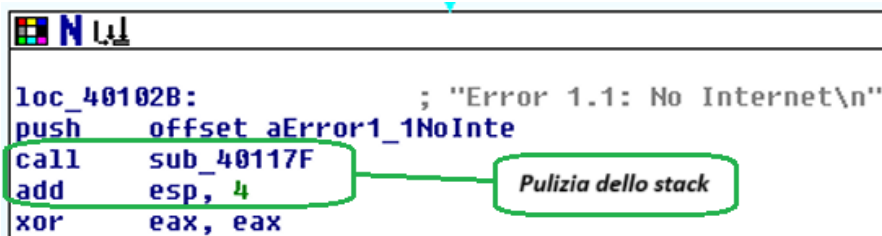
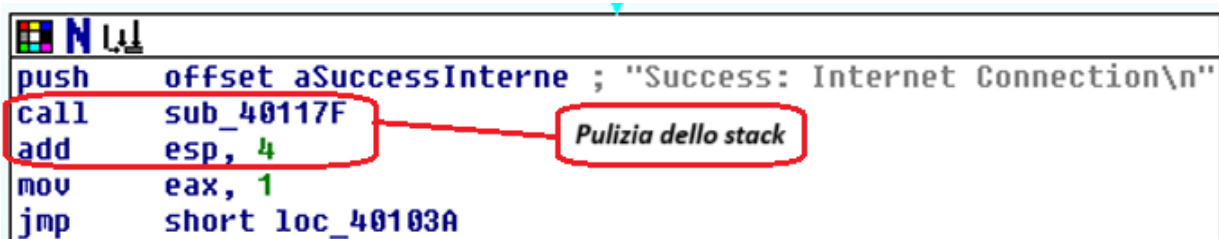
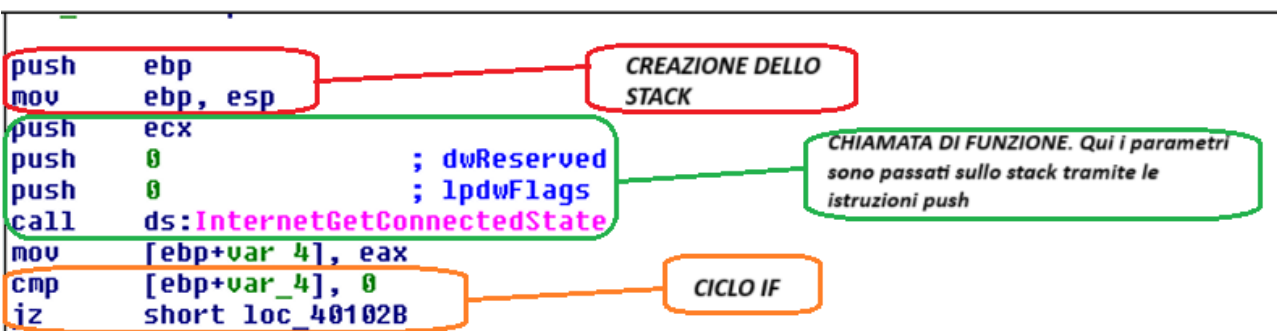
Name	Virtual Size	Virtual Address	Raw Size	Raw Address	Reloc Address	Linenumbers	Relocations N...	Linenumbers ...	Characteristics
Byte[8]	Dword	Dword	Dword	Dword	Dword	Dword	Word	Word	Dword
.text	00004A78	00001000	00005000	00001000	00000000	00000000	0000	0000	60000020
.rdata	0000095E	00006000	00001000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

- **.text**: contiene le istruzioni (le righe di codice) che la CPU eseguirà una volta che il software sarà avviato. Generalmente questa è l'unica sezione di un file eseguibile che viene eseguita dalla CPU, in quanto tutte le altre sezioni contengono dati o informazioni a supporto.
- **.rdata**: include generalmente le informazioni circa le librerie e le funzioni importate ed esportate dall'eseguibile, informazione che come abbiamo visto possiamo ricavare con CFF Explorer.
- **.data**: contiene tipicamente i dati / le variabili globali del programma eseguibile, che devono essere disponibili da qualsiasi parte del programma. Una variabile si dice globale quando non è definita all'interno di un contesto di una funzione, ma bensì è globalmente dichiarata ed è di conseguenza accessibile da qualsiasi funzione all'interno dell'eseguibile

Facendo riferimento al seguente codice in Assembly, rispondere ai quesiti:



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



#### 4. *Ipotizzare il comportamento della funzionalità implementata:*

La funzionalità implementata dal codice è la seguente: richiama la funzione `internetgetconnectedstate` e ne verifica il valore di ritorno tramite un'istruzione `"if"`. Se il valore restituito dalla funzione è diverso da zero, significa che c'è una connessione attiva.

#### 5. *BONUS fare tabella con significato delle singole righe di codice assembly*

INDIRIZZO	ISTRUZIONE	DESCRIZIONE
0x00401000	push ebp	Salva il valore corrente del registro di base dell'allocazione nello stack.
0x00401001	mov ebp, esp	Imposta il registro di base dell'allocazione uguale al puntatore.
0x00401003	push ecx	Salva il valore corrente del registro ECX nello stack.
0x00401004	push 0	Mette il valore 0 nello stack.
0x00401006	push 0	Mette il valore 0 nello stack.
0x00401008	call ds:InternetGetConnectedState	Chiama la funzione <code>InternetGetConnectedState</code> .
0x0040100E	mov [ebp+var_4], eax	Memorizza il valore restituito dalla funzione in <code>[ebp+var 4]</code> .
0x00401011	cmp [ebp+var_4], 0	Confronta il valore memorizzato in <code>[ebp+var_4]</code> con 0.
0x00401015	jz short loc_40102B	Salta all'indirizzo <code>loc_40102B</code> se il confronto precedente ha dato come risultato zero.
0x00401017	push offset asuccessInterne	Mette l'indirizzo della stringa <code>"Success: Internet Connection\n"</code> nello stack.
0x0040101C	call sub_40105F	Chiama la subroutine <code>sub_40105F</code> .
0x00401021	add esp, 4	Ripristina lo stack dopo la chiamata della funzione.

0x00401024	mov eax, 1	Imposta il registro EAX a 1.
0x00401029	jmp short loc_40103A	Salta all'indirizzo loc_40103A.
0x0040102B	push offset aError1_1NoInte	Mette l'indirizzo della stringa "Error 1.1: No Internet\n" nello stack.
	call sub_40117F	Chiama la subroutine sub_40117F.
	add esp, 4	Ripristina lo stack dopo la chiamata della funzione.
	xor eax, eax	Esegue un'operazione di XOR tra eax e eax, impostando eax a zero.
0x0040103A	mov esp, ebp	Ripristina il puntatore dello stack al valore salvato in ebp.
	pop ebp	Ripristina il registro di base dell'allocazione dallo stack.
	retn	Restituisce il controllo al chiamante.
	sub_401000 endp	Fine della subroutine sub_401000.

(le righe senza un indirizzo sono parte delle istruzioni che seguono immediatamente l'indirizzo indicato nella riga precedente).