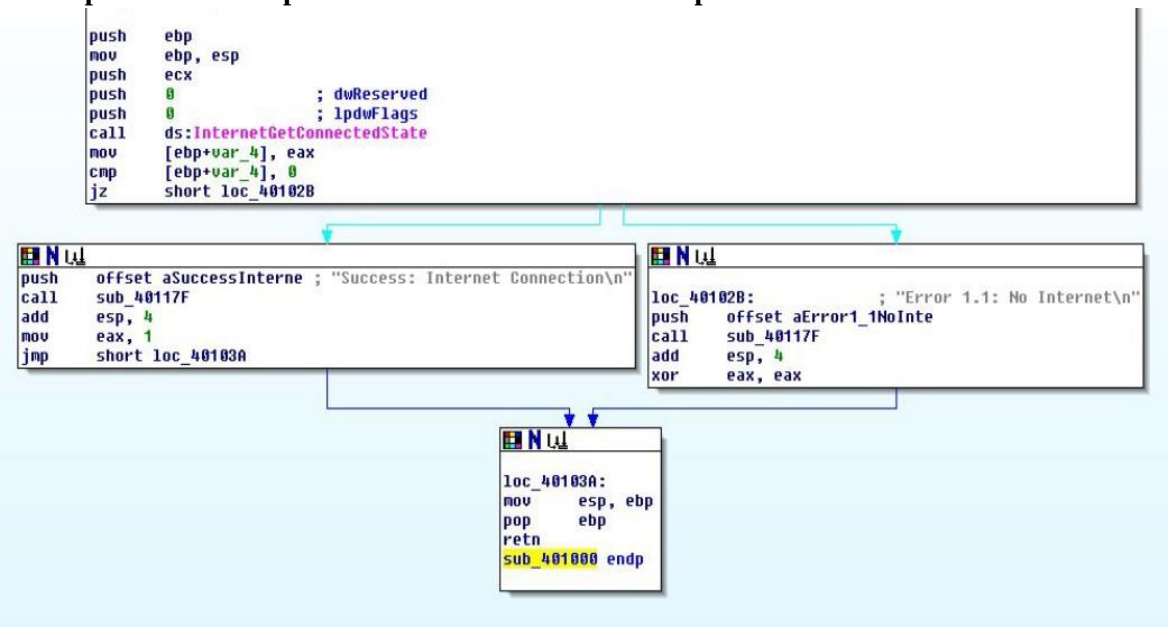
Gabriele Di giampietro **Esercitazione Settimanale 13/01/2023**

Obbiettivo:

- **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. Trovare quali librerie vengono importate e quali sono le sezioni di cui si compone il file eseguibile del malware.**

- **Con riferimento alla parte di codice in Assembly nella figura seguente, identificare i costrutti noti e ipotizzare il comportamento della funzionalità implementata.**

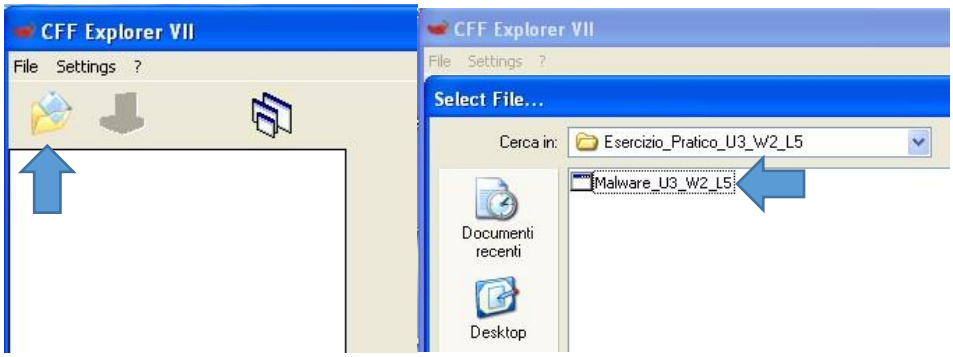


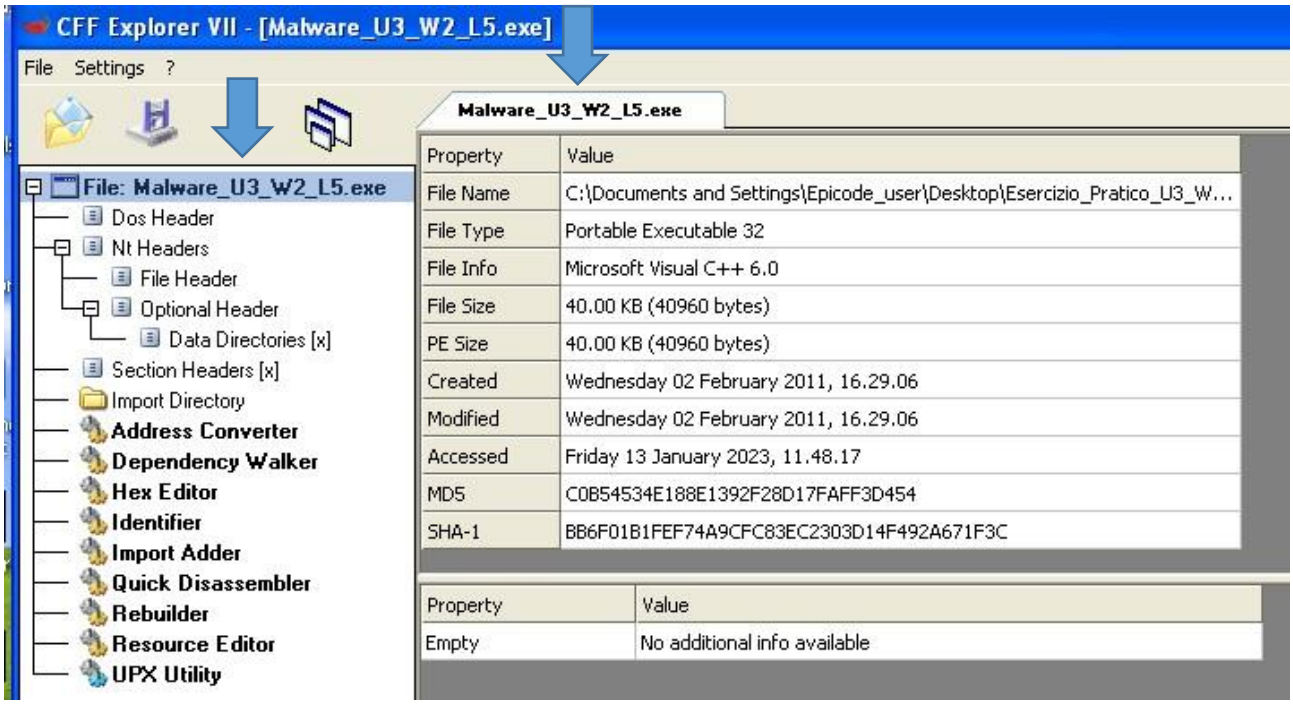
Prima Fase

Nella prima fase della nostra esercitazione andiamo ad effettuare una **Analisi statica base**. Per fare ciò andiamo ad utilizzare una VM completamente isolata sia da connessione internet che da collegamenti con cartelle condivise, porte usb attive ecc… come è necessario per svolgere una corretta Analisi di un Malware in un ambiente sicuro.

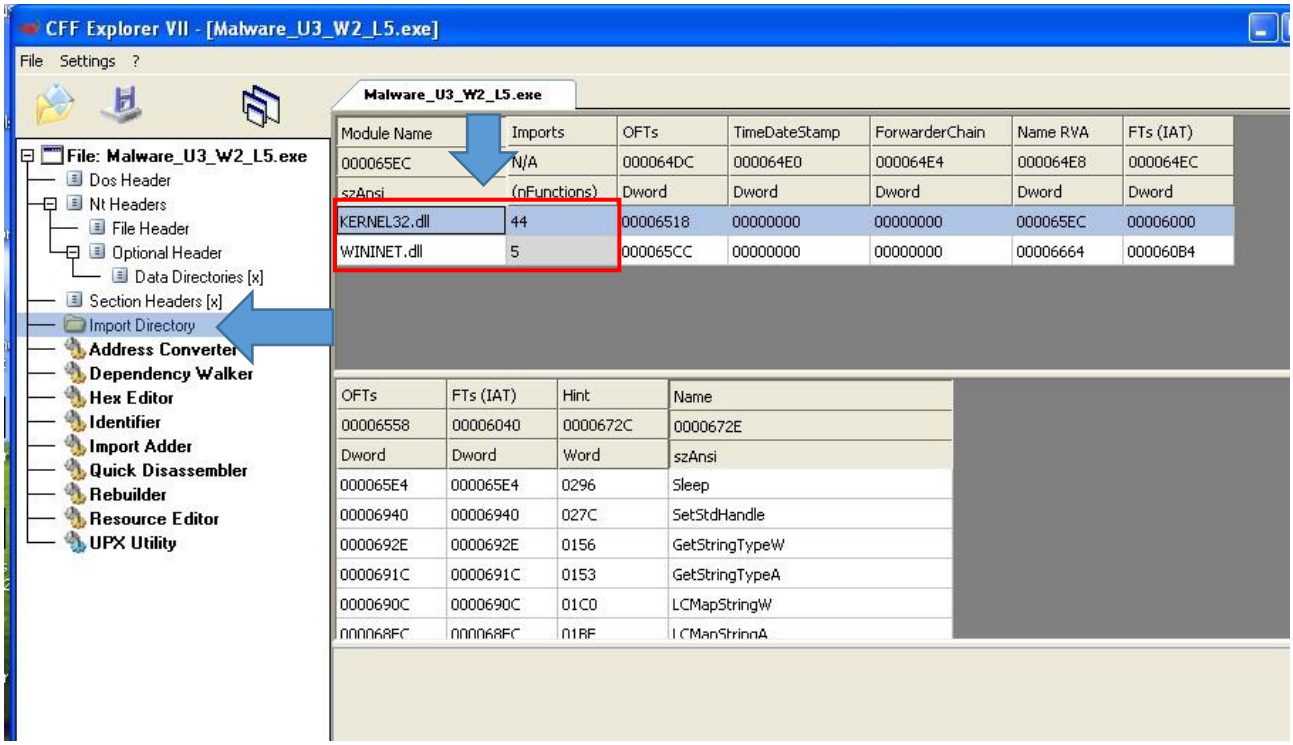
Nel nostro caso analizziamo il file eseguibile dato dall'obbiettivo dell'esercitazione, essendo un'analisi statica non andremo ad eseguire il Malware ma andremo solo ad analizzare il suo contenuto. Per fare ciò utilizziamo il tool **CFF explorer** che ci permette di estrapolare informazioni riguardo il Malware analizzato come librerie che importa e le sue sezioni.

Come prima cosa andiamo ad aprire il nostro tool e carichiamo il Malware.



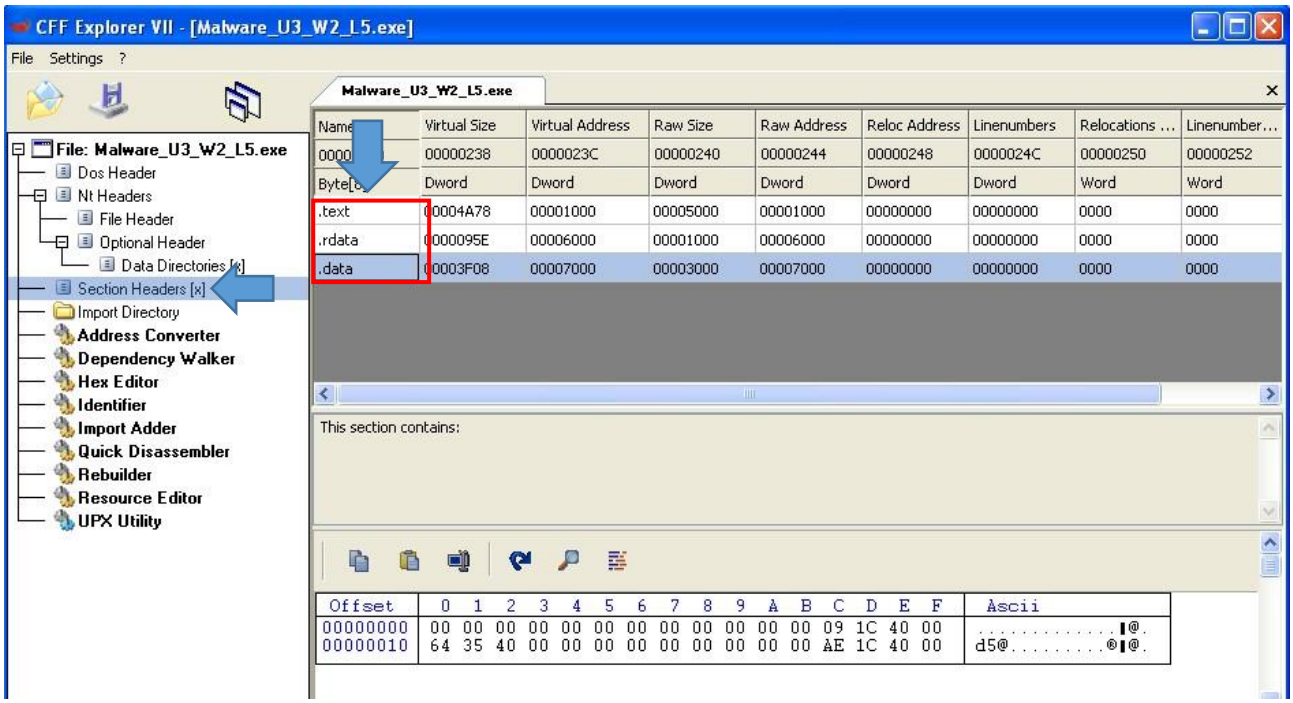


Dopo di che selezioniamo dalle opzioni laterali la sezione **Import Directory**, la quale ci mostra quale librerie sono importante dal Malware sul sistema attaccato. Troviamo infatti, evidenziato nel riquadro rosso, la libreria **KERNEL32.dll** che rispettivamente importa 44 funzioni (questa libreria è piuttosto comune e contiene le funzioni principali per interagire con il sistema operativo, i filesystem) e **WININET.dll** che importa 5 funzioni (questa libreria invece contiene le funzioni per l'implementazione di alcuni protocolli di rete come HTTP, FTP, NTP).



Selezionando sulle librerie possiamo visualizzare le funzioni che importano.

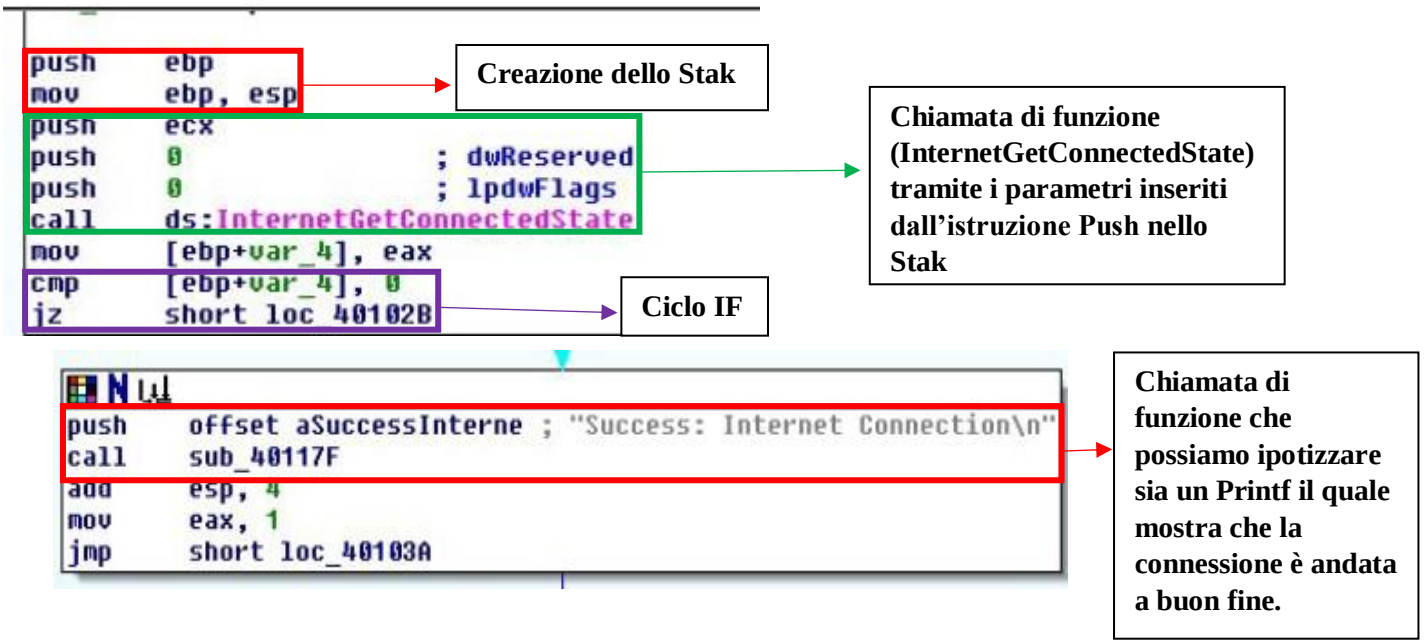
Dopo aver trovato quali librerie importa il nostro Malware andiamo a individuare di quante sezioni è composto. Selezioniamo dalle opzioni laterali **Section Headers** e troviamo infatti che è composto da tre sezioni che possiamo vedere nel riquadro rosso nell'immagine seguente.

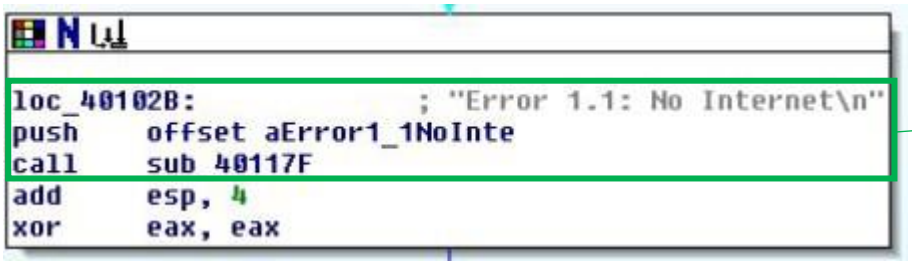


Le sezioni trovate sono: **.text** che contiene istruzioni, o meglio righe di codice, che la CPU eseguirà una volta che il software sarà avviato; **.rdata** la quale include generalmente le informazioni circa le librerie e le funzioni importate ed esportate dal file eseguibile; **.data** che contiene tipicamente i dati o le variabili globali del file eseguibile, che devono essere disponibili da qualsiasi parte del programma.

Seconda Fase

Nella seconda parte dell'esercitazione andiamo ad analizzare un codice di un Malware in Assembly dato in figura nell'obbiettivo dell'esercitazione e dalla nostra analisi abbiamo individuato **6 costrutti noti** che indichiamo in evidenza nelle immagini seguenti.





Chiamata di funzione che anche in questo caso è un Printf ma che mostra che la connessione non è andata a buon fine.

Tramite queste due ultime istruzione si svuota lo Stak

Possiamo dedurre che la funzionalità implementata è la chiamata di funzione **InternetGetConnectedState**, ovvero un controllo di connessione sulla macchina attaccata, di fatti attraverso il **ciclo if** ne controlla il valore di ritorno. Se il valore di ritorno della funzione è diverso da 0 allora significa che la connessione sul sistema attaccato è andato a buon fine, se invece il valore di ritorno sarà uguale a 0 allora la connessione non è andata a buon fine è l'istruzione **jump (jz)** salterà nell'allocazione di memoria indicata.