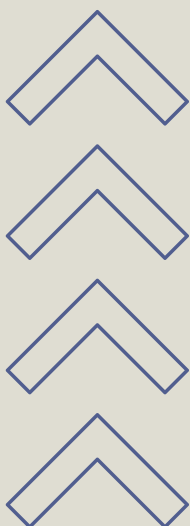




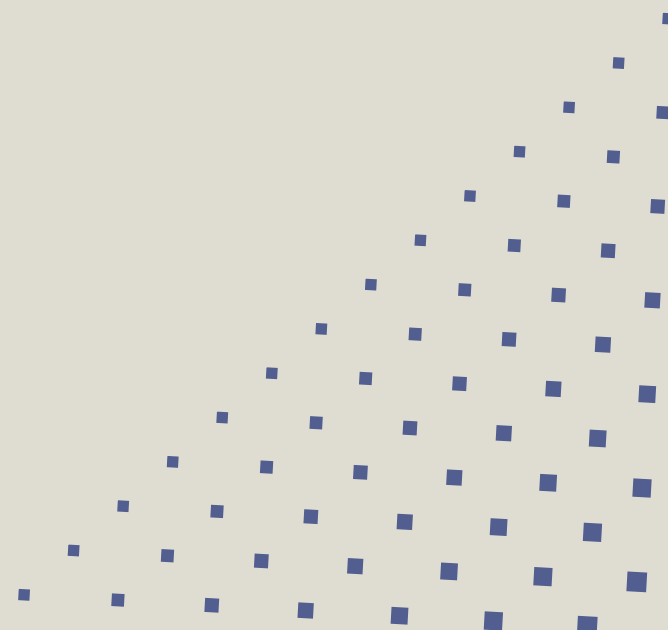
GIUGNO, 2024

PROGETTO

MALWARE ANALYSIS




Redatto da:
Anapaula Palacin





INDICE

Introduzione e risorse	3
Traccia	4
Analisi librerie	5
Analisi sezioni	6
Identificazione costrutti	7
Funzionalità	8
Spiegazione del codice	9
Conclusioni e BP	11





INTRODUZIONE

L'analisi del malware ha il compito di comprendere il funzionamento, le capacità e l'impatto dei software dannosi. Esistono principalmente due tipi di analisi:

Analisi statica: È fare l'analisi del codice senza eseguire il malware. Viene effettuata attraverso la decompilazione del codice permettendo di studiare la struttura e il suo comportamento.

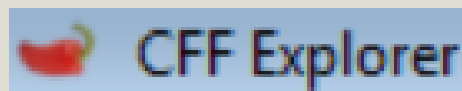
Analisi dinamica: Implica l'esecuzione del malware in un ambiente controllato per osservare il suo comportamento in tempo reale questo permette di rilevare le azioni che il malware compie durante la sua esecuzione.



RISORSE UTILIZZATE

CFF EXPLORER

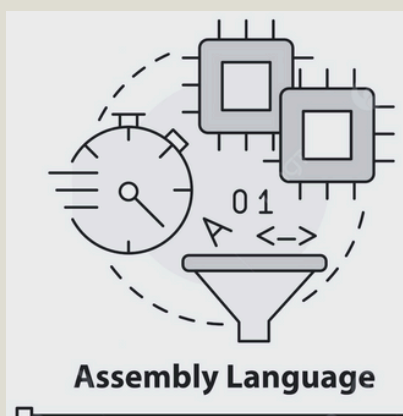
CFF Explorer è un tool gratuito per analizzare e modificare i file eseguibili di Windows. Consente di visualizzare e modificare intestazioni, sezioni, importazioni, esportazioni e risorse. È utile per l'analisi del malware, l'inversione e la decompressione dei file.



ASSEMBLY

È un linguaggio di programmazione di basso livello che viene utilizzato per comunicare direttamente con l'hardware del computer.

L'uso dell'Assembly fa parte dell'analisi statica avanzata ed è fondamentale per una comprensione approfondita e dettagliata del funzionamento interno del malware a livello d'istruzione macchina



»»»» TRACCIA

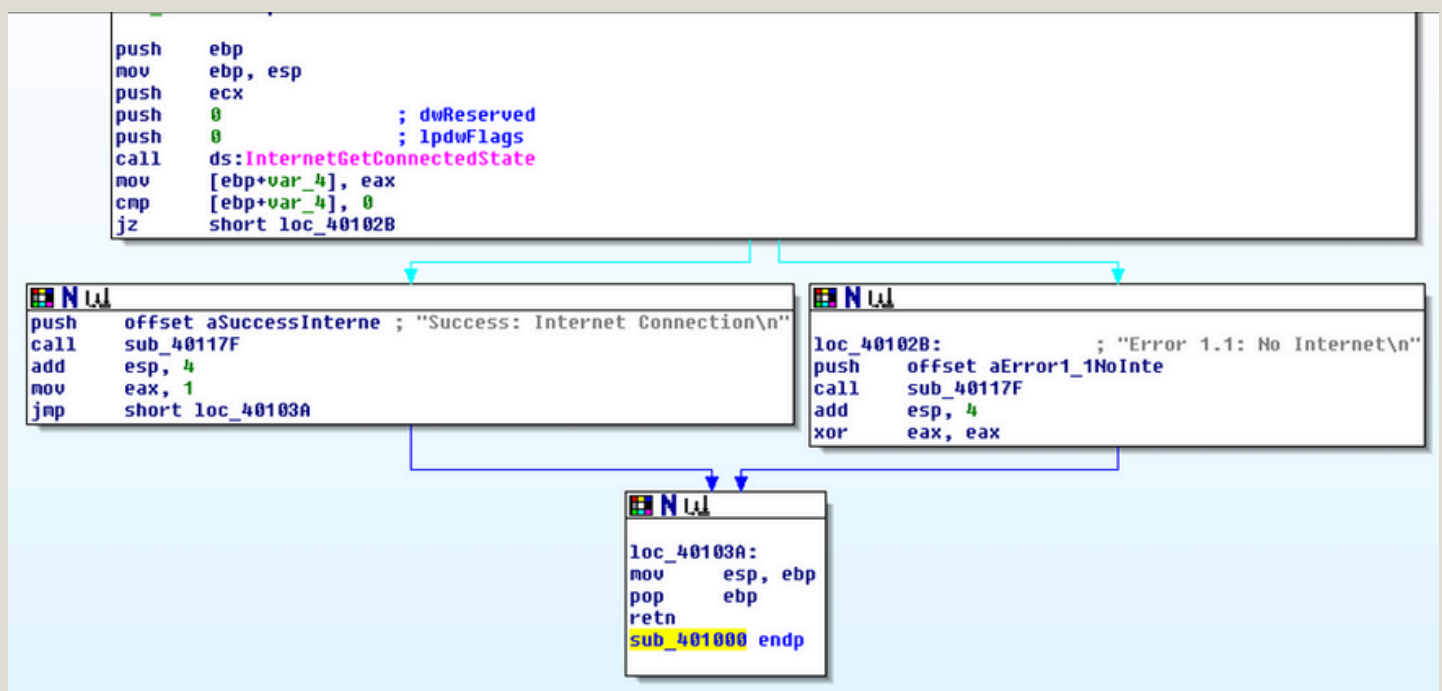
Con riferimento al file **Malware_U3_W2_L5** presente all'interno della cartella «**Esercizio_Pratico_U3_W2_L5**» sul desktop della macchina virtuale dedicata per l'analisi dei malware, rispondere ai seguenti quesiti:

- 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, altri costrutti)
- Ipotesizzare il **comportamento della funzionalità** implementata
- BONUS fare tabella con **significato** delle singole righe di codice **Assembly**

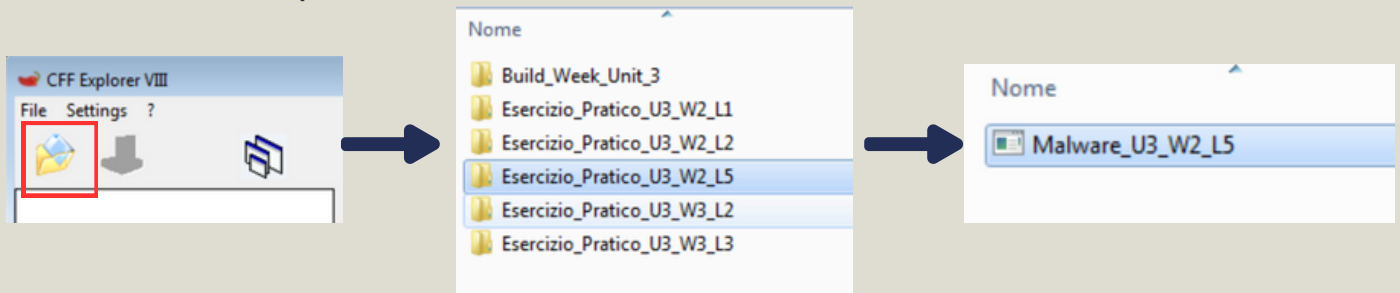
FIGURA 1



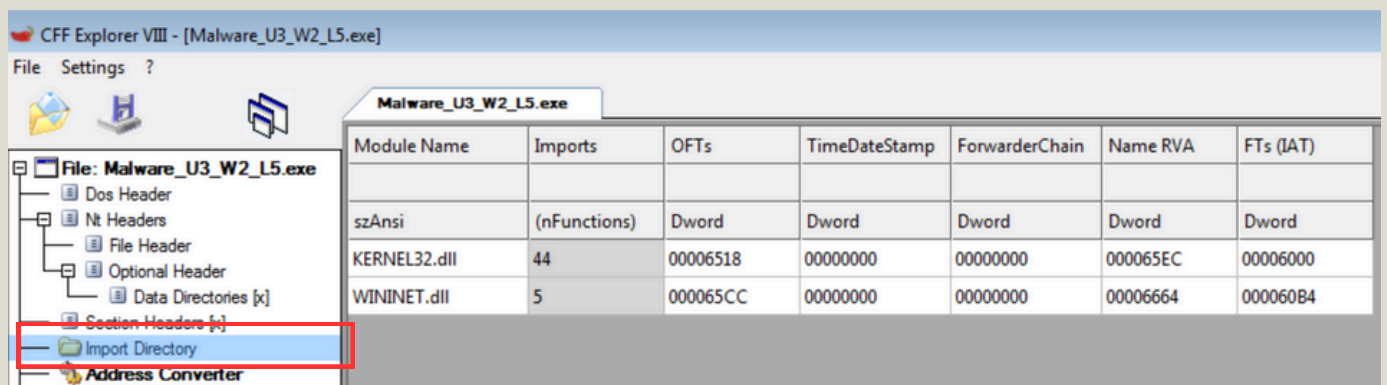


ANALISI LIBRERIE

Per iniziare l'analisi delle librerie importate del malware possiamo fare l'uso del tool CFF Explorer.



Una volta aperto il malware, dobbiamo andare al pannello principale e scegliere "Import Directory"



Le librerie contengono funzioni e risorse che i programmi possono utilizzare per svolgere varie operazioni. Nel contesto di malware si utilizzano per compiere azioni dannose o nascoste. Nell'analisi fatta possiamo vedere queste 2 librerie:

KERNEL32.dll

Contiene le funzioni base per interagire col sistema operativo windows come la gestione della memoria, la manipolazione dei file e la gestione dei processi.

Gestione dei file: Funzioni come CreateFile, ReadFile, WriteFile e DeleteFile

Gestione dei processi: Funzioni come CreateProcess, ExitProcess

WININET.dll

Fornisce le funzioni per implementare i protocolli Internet e gestire connessioni di rete attraverso protocolli come HTTP e FTP.

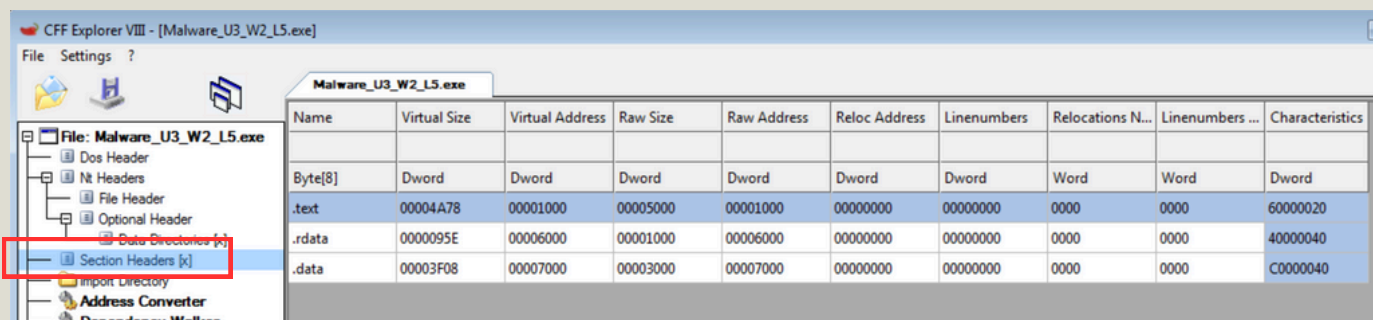
Gestione delle connessioni internet: Funzioni come InternetOpen, InternetConnect.

Trasferimento di dati: Funzioni come InternetReadFile, InternetWriteFile, HttpSendRequest.



ANALISI SEZIONI

Ora dobbiamo ritornare al pannello principale e scegliere “Section Headers”



Le sezioni sono uno degli aspetti cruciali da esaminare perché contengono dati specifici e necessari per avere una chiara visione del comportamento del malware nell'esecuzione del programma. Le sezioni trovate in questo caso sono:

.text

Contiene il codice eseguibile. Generalmente é la prima sezione ad essere eseguita quando il programma viene avviato. Ha una protezione RX: Read and Execute.

.rdata (Read-Only Data)

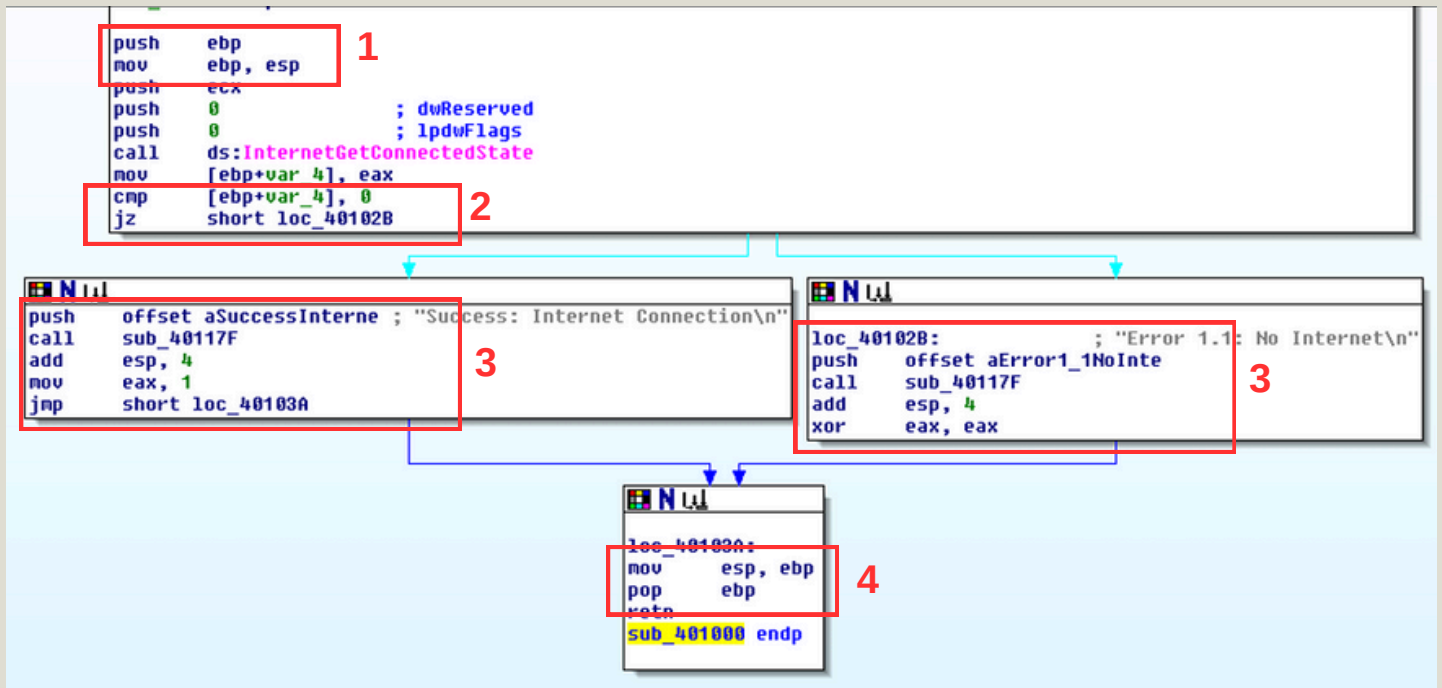
Contiene dati di solo lettura come stringhe di testo o informazioni costanti.

.data

Contiene dati che possono essere letti e modificati durante l'esecuzione del programma. Si utilizza anche per memorizzare variabili globali e dati inizializzati.



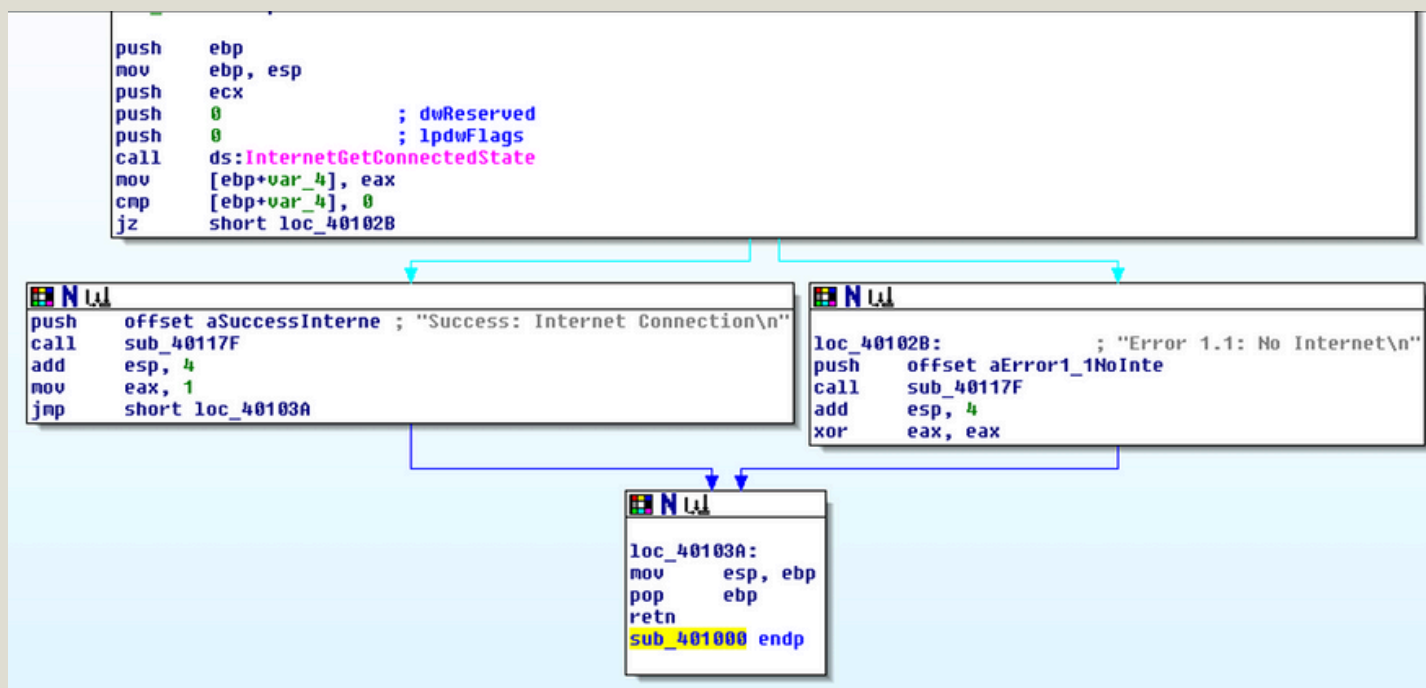
IDENTIFICAZIONE COSTRUTTI



Possiamo identificare diversi costrutti:

- 1 Costrutto per la Creazione dello Stack:** Tramite le istruzioni push e mov
- 2 Costrutto condizionale IF:** Per la presenza delle istruzioni “cmp” che confronta i registri e “jz” (jump if zero) fa un salto condizionale
- 3 Costrutto per Messaggi di Stato:** Gestisce la visualizzazione dei messaggi di successo o errore (Success / Error)
- 4 Costrutto per la Pulizia dello Stack:** esce della funzione e ripristina lo stack e i registri con le istruzioni “mov esp, ebp”

»»»» FUNZIONALITÀ



Il codice del malware verifica se il sistema è connesso a Internet utilizzando la funzione `InternetGetConnectedState`. In base al risultato della chiamata:

- Se la connessione è presente, stampa "Success: Internet Connection" e imposta `eax` a 1.
- Se la connessione non è presente, stampa "Error 1.1: No Internet" e imposta `eax` a 0.

Una volta determinato lo stato della connessione Internet la funzione esegue le istruzioni di uscita.



SPIEGAZIONE CODICE

- 1 Crea e salva il valore del puntatore nello stack
- 2 Imposta il puntatore ebp al valore del puntatore esp

```
1 * .text:00401000      push    ebp
2 * .text:00401001      mov     ebp, esp
```

- 3 Salva il valore del registro "ecx" nello stack

```
3 * .text:00401003      push    ecx
   * .text:00401004      push    0          ; dwReserved
   * .text:00401006      push    0          ; lpdwFlags
   * .text:00401008      call   ds:InternetGetConnectedState
```

- 4 Chiama la funzione "InternetGetConnectedState" e fa passare i parametri sullo stack tramite l'istruzione push per verificare lo stato della connessione a Internet.

```
5 * .text:0040100E      mov     [ebp+var_4], eax
6 * .text:00401011      cmp     [ebp+var_4], 0
7 * .text:00401015      jz      short loc_40102B
```

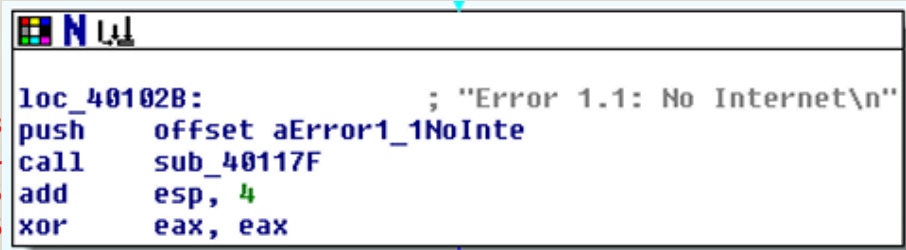
- 5 Salva la risposta di "InternetGetConnectedState" nella variabile locale "var_4"
- 6 Confronta il valore di "var_4" con 0
- 7 Se "var_4" è uguale a 0, salta all'indirizzo loc_40102B. (no internet)

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

- 8 "push" spinge l'indirizzo della stringa "aSuccessInterne" nello stack
- 9 Chiama la funzione "sub_40105F" per stampare il messaggio
- 10 aggiusta il puntatore dello stack aumentando esp di 4
- 11 Imposta la risposta di "InternetGetConnectedState" a 1
- 12 Fa un salto alla etichettata "loc_40103A"



SPIEGAZIONE CODICE



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

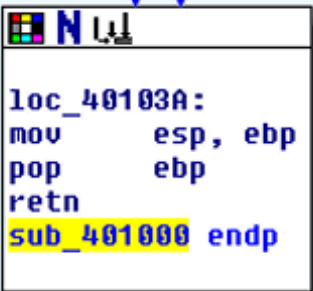
Etichetta nel caso di errore nella connessione

13 “push” spinge l’indirizzo della stringa “aError1_1NoInte” nello stack

14 Chiama la funzione “sub_40105F” per stampare il messaggio

15 aggiusta il puntatore dello stack aumentando esp di 4

16 Imposta il registro eax a 0 per indicare l'errore.



```
loc_40103A:
17 mov     esp, ebp
18 pop     ebp
19 retn
Sub_401000 endp
```

Etichetta per il termine della funzione

17 Ripristina il valore del registro esp dal registro ebp

18 Ripristina il valore precedente del registro ebp

19 Ritorna dalla funzione



BUONE PRATICHE E CONCLUSIONI

Buone pratiche:

1. Ambiente controllato:

Eseguire sempre l'analisi del malware in un ambiente controllato e isolato, come una macchina virtuale, per evitare la diffusione accidentale del malware.

2. Documentazione accurata:

Documentate ogni fase dell'analisi, comprese le sezioni di codice esaminate, le funzioni analizzate e i comportamenti osservati.

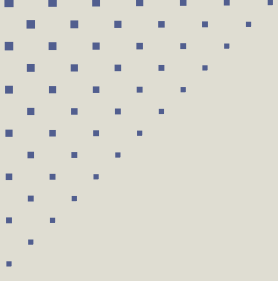
3. Analisi statica e dinamica:

Combinare l'analisi statica (esaminare il codice senza eseguirlo) con l'analisi dinamica (osservare il comportamento del malware in esecuzione) per ottenere un quadro completo.

Conclusioni:

L'esercizio ci ha permesso di identificare e capire come il malware controlla lo stato della connessione a Internet e risponde con messaggi di successo o di errore.

Strumenti e tecniche come CFF Explorer sono essenziali per disassemblare e analizzare il codice del malware, consentendo agli analisti di ispezionare le sezioni di codice e le intestazioni delle sezioni.



FINE

