MRS

MALWARE ANALYSIS | ASSEMBLY

Presentato da Alessandro Marasca
EPICODE - CS0124
S10L5

Indice

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

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

1. Identificare i **costrutti noti** (creazione dello stack, eventuali cicli, altri costrutti)
2. Ipotizzare il **comportamento** della funzionalità implementata
3. **BONUS** fare tabella con significato delle singole righe di codice assembly

LIBRERIE

Verifico attraverso **CFF Explorer** quali sono le librerie importate:

KERNEL32.DLL = contiene le funzioni principali per interagire col sistema operativo o per gestire la memoria.

WININET.DLL = permette l'utilizzo di protocolli di rete (HTTP, FTP, NTP) .

Presumo che il malware si connetta ad internet per scaricare altro oppure divulgare informazioni private ad un dispositivo o server remoto.

Malware_U3_W2_L5.exe

Module Name	Imports	OFTs	TimeDateStamp	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

SEZIONI

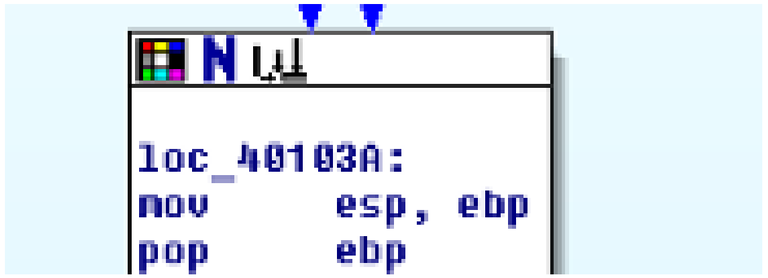
Identifico le **sezioni** da cui è composto il Malware; suggeriscono il funzionamento:
.text = contiene le istruzioni che saranno inviate alla **CPU** quando il malware sarà attivo.
.rdata = include informazioni sulle librerie importate ed esportate.
.data = contiene le variabili globali (perche sono disponibili in qualsiasi funzione dell'eseguibile) dell'eseguibile.

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

COSTRUTTI NOTI

```
1   push    ebp
2   mov     ebp, esp
3   push    ecx
4   push    0                   ; dwReserved
5   push    0                   ; lpdwFlags
6   call    ds:InternetGetConnectedState
7   mov     [ebp+var_4], eax
8   cmp     [ebp+var_4], 0
9   jz      short loc_40102B
```



1-2

Creazione stack

3-6

Chiamata funzione internetgetconnectedstate

8-9

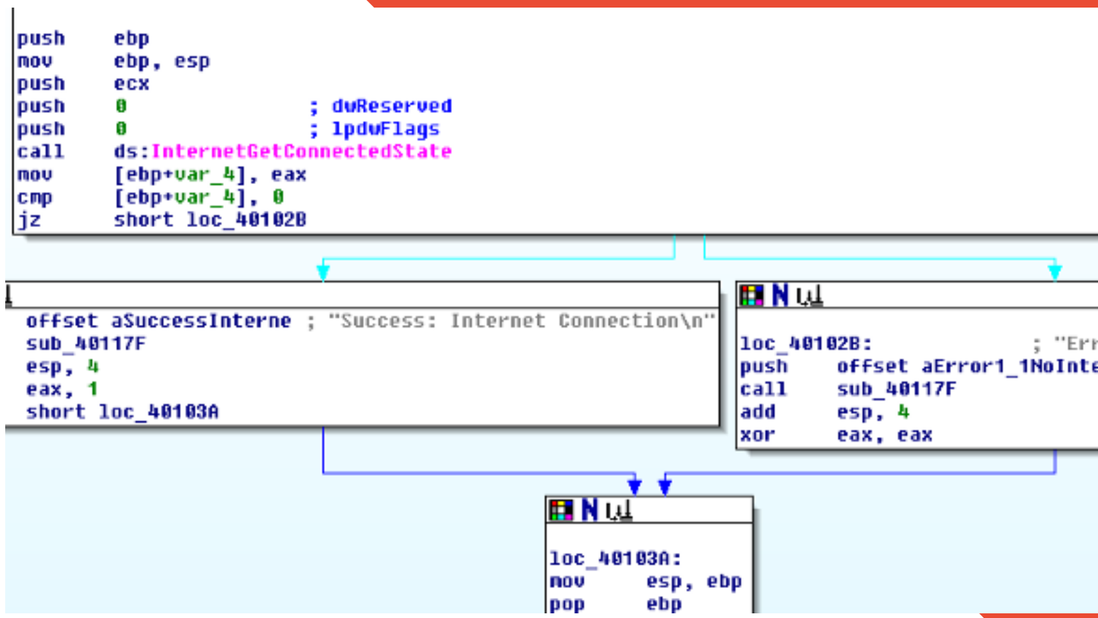
IF, comparazione che tramite Jz(jump zero) effettua un jump nella cella di memoria indicata.

10 - 12

Chiusura stack

IPOTESI COMPORTAMENTO

Il codice (una funzione scritta in linguaggio **assembly x86**) sembra gestire lo stato della connessione Internet. Setta il frame del registro base (**ebp**) e dello stack pointer (**esp**). Prepara gli argomenti per la chiamata alla funzione **InternetGetConnectedState** mettendo dei valori sulla pila.
Dopo aver ottenuto il risultato della chiamata, controlla se la connessione è attiva per poi confrontare il risultato con zero.
Connessione attiva: stampa un messaggio di successo tramite la funzione **sub_40105F**,
Connessione non attiva: passa alla gestione degli errori, chiamando la funzione **sub_40117F**.
In ultimo l'epilogo ripristina lo stack pointer e il registro base, terminando la funzione.



Grazie!

Alessandro Marasca
Epicode - CS0124
S10L5