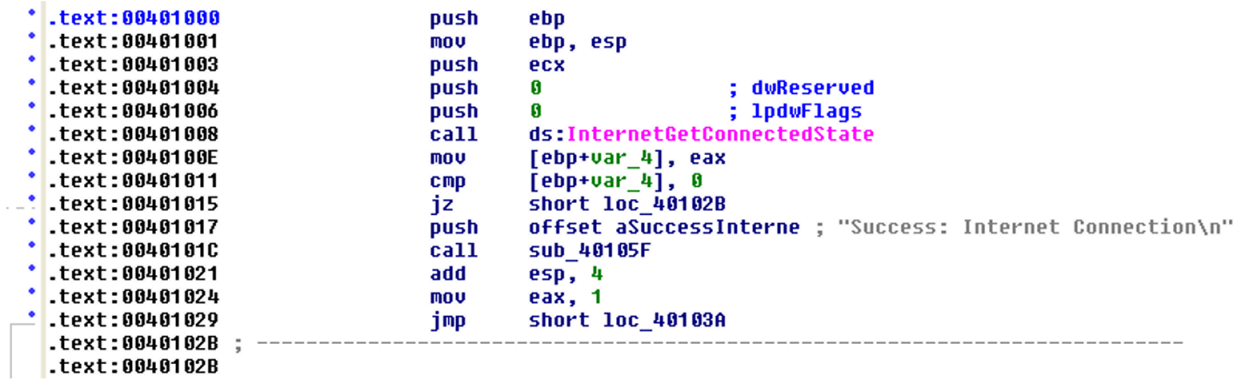
Traccia:

La figura seguente mostra un estratto del codice di un malware.
Identificare i costrutti noti visti durante la lezione teorica.



Guardando il codice fornito, possiamo identificare alcuni costrutti noti e operazioni comuni:

Push e Pop nello stack: Le istruzioni push e pop vengono utilizzate per manipolare lo stack, inserendo e rimuovendo valori.

Movimento di valori tra registri e memoria: Le istruzioni mov vengono utilizzate per spostare dati tra registri e posizioni di memoria.

Chiamata di funzione: L'istruzione call viene utilizzata per chiamare una funzione.

Controllo del flusso condizionale: L'istruzione cmp viene utilizzata per confrontare due valori, mentre l'istruzione jz (jump if zero) viene utilizzata per saltare a una posizione specificata se il confronto precedente ha dato esito zero.

Salti incondizionati: L'istruzione jmp viene utilizzata per saltare incondizionatamente a una posizione specificata nel codice.

Non ci sono cicli while o for espliciti nel codice fornito. Potrebbero esserci all'interno delle funzioni chiamate, ma non sono visibili nel frammento fornito.

PUNTO 2

Il ciclo di questo programma non è molto complesso. Esso:

- Effettua alcune operazioni preliminari nello stack e chiama *InternetGetConnectedState* per verificare la connessione a Internet.
- Se c'è una connessione, stampa un messaggio di successo.
- Infine, restituisce 1 (presumibilmente per indicare un'esecuzione con successo).

BONUS:

push ebp: Salva il valore di ebp nello stack. ebp (Base Pointer) viene spesso utilizzato per puntare ai frame delle funzioni nello stack durante l'esecuzione di un programma.

mov edp, esp: Muove il valore dello stack pointer (esp) nel registro edp. esp (Stack Pointer) indica la cima dello stack e edp viene usato come base pointer.

push 0: Mette il valore 0 nello stack. Questo valore potrebbe essere utilizzato come parametro per una funzione successiva.

push 0: Mette nuovamente il valore 0 nello stack. Potrebbe essere un altro parametro per una funzione successiva.

call ds:InternetGetConnectedState: Chiama la funzione InternetGetConnectedState presumibilmente definita nel segmento di dati (ds). Questa funzione verifica se il sistema è connesso a Internet e restituisce un valore che indica lo stato della connessione.

mov [edp +var_4], eax: Memorizza il valore restituito dalla funzione InternetGetConnectedState nella variabile locale [edp + var_4].

cmp [ebp+var_4], 0: Compara il valore memorizzato in [ebp + var_4] con 0.

jz short loc_40102B: Salta a loc_40102B se il risultato della comparazione precedente è zero, il che significa che non c'è una connessione a Internet.

push offset aSuccessInterne: Mette l'indirizzo della stringa "Success: Connection\n" nello stack. Questa stringa sembra essere un messaggio di successo.

call sub_40105F: Chiama una subroutine sub_40105F, probabilmente per stampare il messaggio di successo.

add esp, 4: Libera lo spazio per il parametro messo nello stack in precedenza, spostando lo stack pointer.

mov eax, 1: Muove il valore 1 nel registro eax. Questo valore potrebbe essere utilizzato per indicare un'esecuzione con successo.

jmp short loc_40103A: Salta a loc_40103A.