**Costrutti C -Assembly x86**

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

A screen shot of a computer program

Description automatically generated

1. Identificare i costrutti noti (es. while, for, if, switch, ecc.)
2. Ipotizzare la funzionalità –esecuzione ad alto livello
3. BONUS: studiare e spiegare ogni singola riga di codice

1) Si identifica un costrutto if dal text 00401011 fino alla 00401029. Dopo aver inizialmente assegnato i valori alle variabili, l’istruzione «cmp» (text 00401011 in figura) unita all’istruzione jnz controllano l’uguaglianza tra le due variabili. Jnz salta alla locazione di memoria specificata se gli operandi sono diversi tra di loro.

2) Il codice sembra fare una verifica dello stato di connessione a Internet del sistema e, in base a quello, esegue un certo percorso di codice se c'è una connessione. Questo tipo di controllo può essere utilizzato da un malware per determinare se è possibile connettersi a un server di comando e controllo o per decidere se intraprendere determinate azioni, come il download o l'upload di dati.

3) spiegazione delle linee di codice:

* **push ebp**, **mov ebp, esp**: Queste istruzioni sono utilizzate per impostare il frame pointer per la funzione corrente, che è una convenzione comune nell'Assembly x86 per la gestione delle chiamate a funzioni.
* **push ecx**, **push 0**: Si tratta di istruzioni che inseriscono valori nello stack. Il valore **0** potrebbe essere utilizzato come parametro per la funzione **InternetGetConnectedState** come puntatore a una variabile che riceve flag aggiuntivi di stato di connessione.
* **call ds:InternetGetConnectedState**: Questa istruzione chiama la funzione **InternetGetConnectedState**. Il prefisso **ds:** indica che l'indirizzo della funzione è ottenuto dal segmento di dati.
* **mov [ebp+var\_4], eax**: Il risultato della funzione (probabilmente un valore booleano che indica lo stato di connessione) viene memorizzato in una variabile locale (indicata come **var\_4**).
* **cmp [ebp+var\_4], 0**: Questa istruzione confronta il valore appena memorizzato con 0 per verificare se il sistema è connesso a Internet.
* **jnz short loc\_40102B**: Questo salto condizionale (**jnz** sta per "jump if not zero") porta il flusso di esecuzione all'indirizzo etichettato se il risultato del confronto non è zero, il che indica che il sistema è connesso a Internet.
* **push offset aSuccessInterne; "Success: Internet Connection\n"**: Qui viene preparato un messaggio che potrebbe essere stampato o registrato da qualche parte, indicando che la connessione Internet è stata rilevata con successo.
* **call sub\_40105F**: Questa è una chiamata a un'altra funzione o subroutine all'interno del codice, identificata come sub\_40105F. La funzionalità specifica di questa subroutine non è visibile nell'immagine, ma potrebbe essere responsabile di eseguire azioni basate sul risultato della verifica della connessione internet.
* **add esp, 4**: Dopo il ritorno dalla chiamata a funzione, questa istruzione corregge lo stack pointer esp per rimuovere il parametro precedentemente inserito nello stack (questo è tipico dopo le chiamate a funzione).
* **mov eax, 1**: Viene movimentato il valore 1 nel registro eax, che potrebbe essere usato come valore di ritorno della funzione corrente o per qualche altro scopo logico all'interno del programma.
* **jmp short loc\_40103A**: Infine, c'è un salto incondizionato a un'altra etichetta nel codice, identificata come loc\_40103A. Questo potrebbe essere il punto in cui il flusso del programma continua dopo aver gestito la connessione internet o la funzione chiamata.