Cybersecurity Specialist

Cyber Security & Ethical Hacking

COSTRUTTI C - ASSEMBLY X86

Alejandro Cristino
S10-L4

TRACCIA

La figura seguente mostra un estratto del codice di un malware.

Identificare i costrutti noti visti durante la lezione teorica.

```
.text:00401000                  push    ebp
.text:00401001                  mov     ebp, esp
.text:00401003                  push    ecx
.text:00401004                  push    0               ; dwReserved
.text:00401006                  push    0               ; lpdwFlags
.text:00401008                  call    ds:InternetGetConnectedState
.text:0040100E                  mov     [ebp+var_4], eax
.text:00401011                  cmp     [ebp+var_4], 0
.text:00401015                  jz      short loc_40102B
.text:00401017                  push    offset aSuccessInterne ; "Success: Internet Connection\n"
.text:0040101C                  call    sub_40105F
.text:00401021                  add     esp, 4
.text:00401024                  mov     eax, 1
.text:00401029                  jmp     short loc_40103A
.text:0040102B ; ---------------------------------------------------------------------------
.text:0040102B
```

Provate ad ipotizzare che funzionalità è implementata nel codice assembly.
Hint : La funzione internetgetconnectedstate se una macchina ha accesso ad Internet. prende in input 3 parametri e permette di controllare

Consegna:
1. Identificare i costrutti noti (e s. while, for, if, switch, ecc.)

2. Ipotizzare la funzionalità – esecuzione ad alto livello

3. BONUS: studiare e spiegare ogni singola riga di codice

1. IDENTIFICARE I COSTRUTTI NOTI

```
mov [ebp+var_4], eax
 cmp [ebp+var_4], 0
 jz   short loc_40102B
```

Questi tre comandi possono essere interpretati come un costrutto "if" in linguaggio C. Se il valore in [ebp+var_4] è pari a zero, il programma eseguirà un salto all'etichetta loc_40102B. In caso contrario, se il valore di var_4 non è zero, il programma continua con l'esecuzione del codice successivo, il quale include la stampa del messaggio di “success internet”. Pertanto, se il confronto precedente non ha restituito zero, il programma non effettuerà il salto al blocco di codice associato all'istruzione jz e continuerà con l'esecuzione del codice che segue immediatamente il confronto.

2. IPOTIZZARE LA FUNZIONALITÀ

Questo codice assembly sembra far parte di un programma che verifica lo stato della connessione Internet e visualizza un messaggio di successo se la connessione è attiva. Inizia impostando il puntatore di base (ebp) e il puntatore dello stack (esp) per gestire le variabili locali e i parametri della funzione.

Chiama la funzione InternetGetConnectedState per verificare lo stato della connessione Internet. Il valore restituito dalla funzione viene salvato nella variabile locale var_4. Viene controllato se il valore di var_4 è zero (che potrebbe indicare l'assenza di connessione) e, in tal caso, il flusso di esecuzione passa a un'etichetta specifica (loc_40102B). Se il valore di var_4 non è zero, il programma procede stampando un messaggio di successo relativo alla connessione Internet. Il programma termina restituendo il valore 1.

BONUS: STUDIARE E SPIEGARE OGNI SINGOLA RIGA DI CODICE

Ogni istruzione nel codice assembly è spiegata di seguito:

- `push ebp`: Salva il valore del registro ebp nello stack, utilizzato come frame pointer.
- `mov ebp, esp`: Imposta il base pointer al valore dello stack pointer.
- `push ecx`: Inserisce il valore del registro ecx nello stack, comunemente utilizzato come contatore.
- `push 0 ;dwReserved`: Inserisce il valore 0 nello stack, potenzialmente come parametro dwReserved.
- `push 0 ;lpdwFlags`: Inserisce il valore 0 nello stack, potenzialmente come parametro lpdwFlags.
- `call ds:InternetGetConnectedState`: Chiama la funzione per verificare lo stato della connessione Internet.
- `mov [ebp+var_4], eax`: eax che copia il contenuto del registro nella variabile var_4.
- `cmp [ebp+var_4], 0`: Confronta il valore di var_4 con 0 per verificare la connessione.
- `jz short loc_40102B`: Salta all'etichetta loc_40102B se il confronto dà zero (nessuna connessione).
- `push offset aSuccessInterne`: Inserisce l'indirizzo dell'etichetta aSuccessInterne nello stack per il messaggio di successo.
- `call sub_40105F`: Chiama una subroutine per stampare il messaggio di successo.
- `add esp, 4`: Ripristina lo stack pointer dopo l'uso.
- `mov eax, 1`: Imposta il valore di ritorno della funzione a 1.
- `jmp short loc_40103A`: Salta all'etichetta loc_40103A, fine della gestione della connessione.