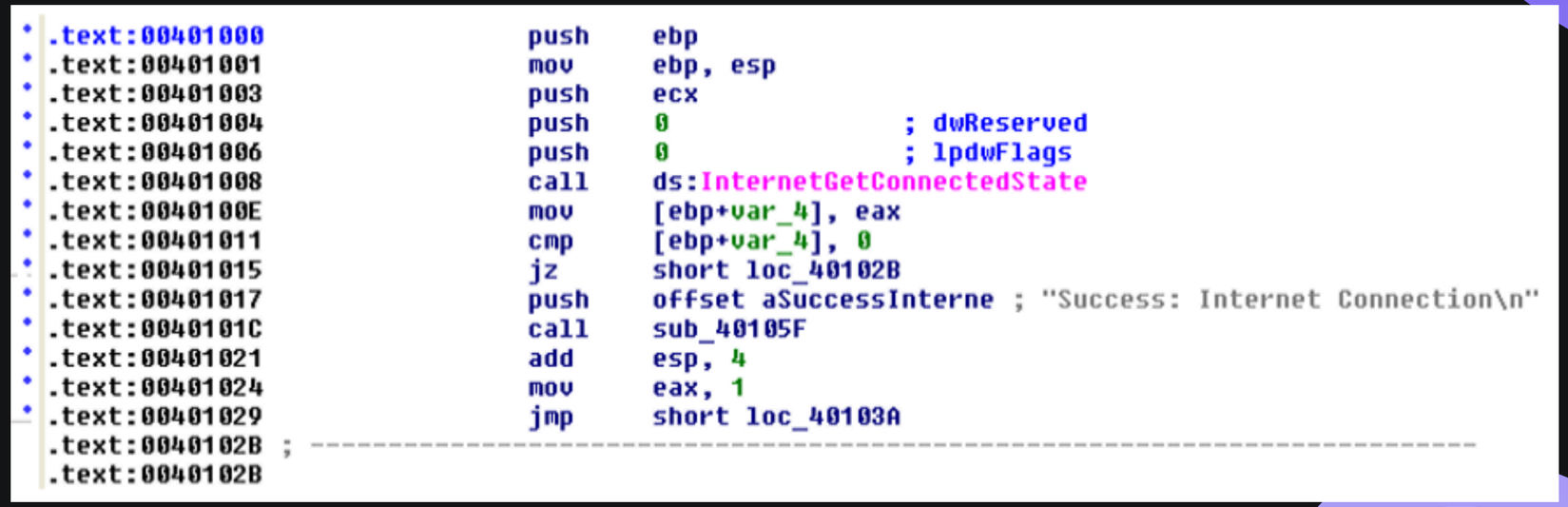
S10/L4

Costrutti C - Assembly x86

GIULIA FIACCHI

Traccia

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



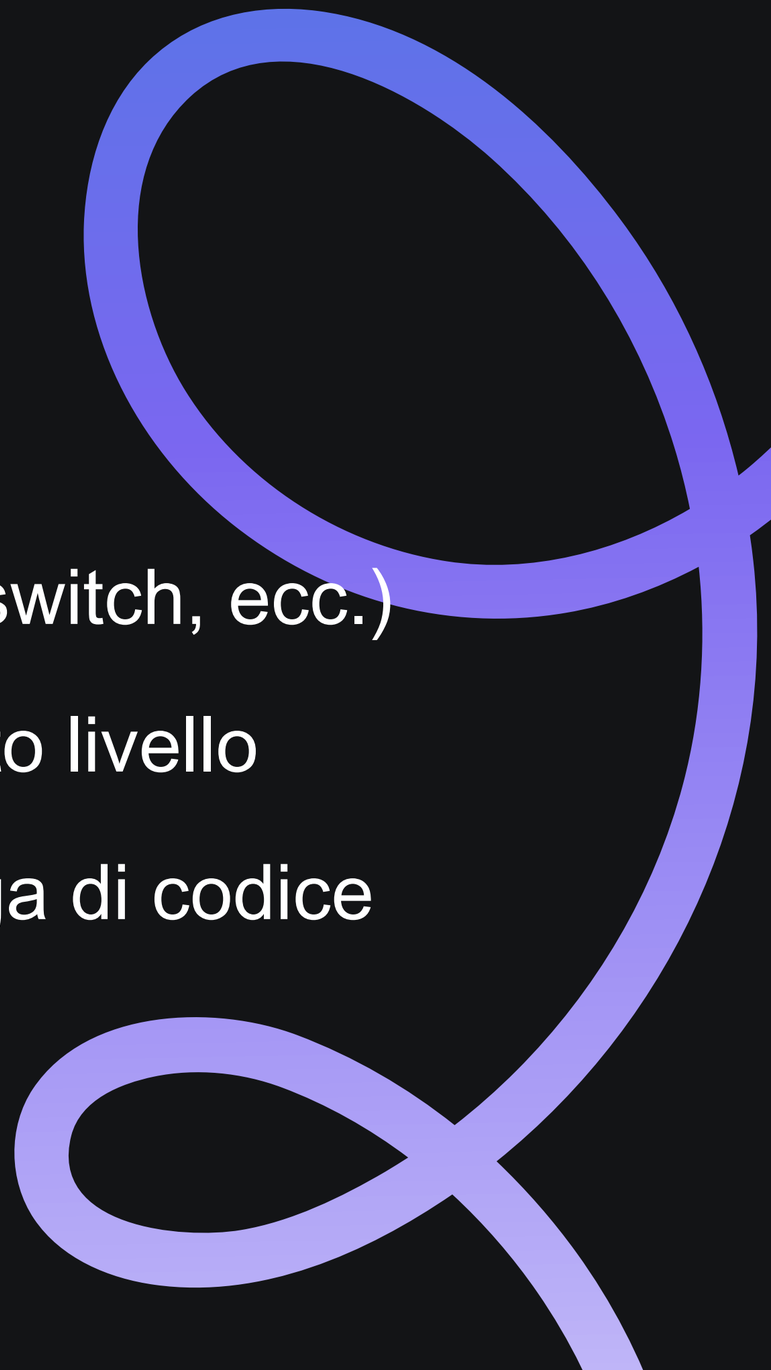
```
.text:00401000                  push    ebp
.text:00401001                  mov     ebp, esp
.text:00401003                  push    ecx
.text:00401004                  push    0               ; dwReserved
.text:00401006                  push    0               ; lpdwFlags
.text:00401008                  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** prende in input 3 parametri e permette di controllare se una macchina ha accesso ad Internet.

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



PARTE 1

Il codice assembly presentato utilizza costrutti noti come **if** (con cmp e jz) e chiama funzioni tramite call che richiamano delle funzioni, che possono essere viste come sotto-programmi o procedure.
Le istruzioni di salto jmp sono usate per trasferire il controllo ad altre parti del codice, che potrebbero essere utilizzate per implementare strutture di controllo come loop o salti condizionali.

PARTE 2

Il codice verifica se la macchina ha accesso a Internet. Se l'accesso è disponibile, stampa un messaggio di successo e imposta un valore di ritorno. Se l'accesso non è disponibile, imposta un altro valore di ritorno.

BONUS

1. push ebp: Salva il valore del base pointer (ebp) sullo stack.
2. mov ebp, esp: Imposta il base pointer (ebp) all'inizio del frame dello stack corrente.
3. push ecx: Salva il valore del registro ecx sullo stack.
4. push 0: Passa il valore 0 come argomento (dwReserved) alla funzione InternetGetConnectedState.
5. push 0: Passa il valore 0 come argomento (lpdwFlags) alla funzione InternetGetConnectedState.
6. call ds:InternetGetConnectedState: Chiama la funzione InternetGetConnectedState per verificare lo stato della connessione Internet
7. mov [ebp+var_4], eax: Salva il risultato della funzione InternetGetConnectedState nel var_4 (una variabile locale).
8. cmp [ebp+var_4], 0: Confronta il valore di var_4 con 0.
9. jz short loc_40102B: Se var_4 è zero (cioè non c'è connessione Internet), salta all'etichetta loc_40102B.
10. push offset aSuccessInterne: Spinge l'offset della stringa "Success: Internet Connection\n" sullo stack.

BONUS

11 call sub_40105F: Chiama una funzione sub_40105F (che probabilmente stampa la stringa "Success: Internet Connection\n").

12 add esp, 4: Ripristina lo stack pointer (esp), rimuovendo l'argomento passato alla funzione sub_40105F.

13 mov eax, 1: Imposta il registro eax a 1 (potrebbe essere usato come codice di ritorno).

14 jmp short loc_401030: Salta all'etichetta loc_401030.

15 loc_40102B: Etichetta per gestire il caso in cui non ci sia connessione Internet.

15 mov eax, *: Imposta il registro eax a un valore non specificato (probabilmente 0, come codice di errore).