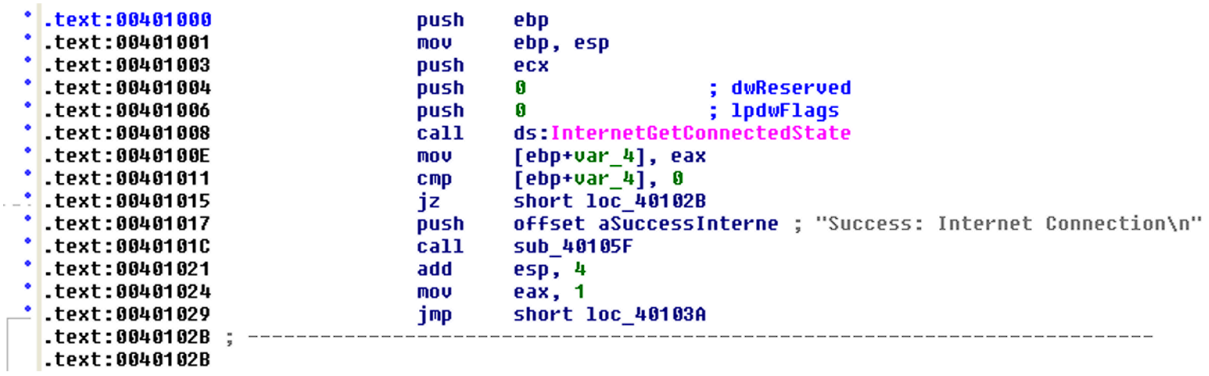
Costrutti C Assembly

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
```

Quesiti:

1. Identificare i costrutti noti (es. while, for, if, switch, creazione/distruzione stack, ecc.)

2. Ipotizzare la funzionalità –esecuzione ad alto livello

3. Spiega il codice riga per riga.

Soluzione:

1. I costrutti noti nella figura sono:
    1. Push = mettere nello stack
    2. Mov = muovere da uno stack all'altro
    3. Call =rimuovere dallo stack
    4. Cmp = conditional jump
    5. Jz = jump
    6. Add = aggiungere
    7. Jmp = jump table

2. La funzionalità del codice è di assicurarsi che ci sia connessione ad internet

3. Questo segmento di codice assembly è scritto nell'architettura x86 e sembra verificare se esiste una connessione Internet attiva in un sistema Windows. Ecco un'analisi passo-passo del codice:

1. Spingi il puntatore base (EBP) nello stack.
2. Sposta il puntatore dello stack (ESP) al puntatore base (EBP) per impostare lo stack frame. Questo consente di accedere alle variabili locali e ai parametri.
3. Spingi il registro ecx nello stack per preservare il suo valore. Questa è una pratica comune per garantire che il valore del registro non venga modificato da chiamate di funzioni annidate.
4. Spingi un valore 0 per il parametro 'dwReserved'.
5. Spingi un valore 0 per il parametro '1pdwFlags'.
6. Chiama la funzione 'InternetGetConnectedState' dal segmento 'ds'. Questa funzione verifica se esiste una connessione Internet attiva e restituisce un valore nel registro EAX.
7. Sposta il valore dal registro EAX alla variabile locale in [EBP+var_4].
8. Confronta il valore in [EBP+var_4] con zero (0).
9. Se il valore è zero (nessuna connessione Internet), salti a label 401017.
10. Spingi l'indirizzo di una stringa 'aSuccess Interne' nello stack, che contiene il messaggio "Success: Internet Connection\n".
11. Chiama una funzione non risolta, presumibilmente per visualizzare il messaggio.
12. Aggiungi 8 al valore in [EBP+var_4].
13. Salta a label 40103A.
14. Se il valore non è zero (esiste una connessione Internet), salta le istruzioni push e call.
15. Sposta 1 nel registro EAX.
16. Salta a label 40103A.

In sintesi, questo codice verifica se esiste una connessione Internet attiva nel sistema. Se non c'è connessione, salta per visualizzare un messaggio che afferma "Success: Internet Connection\n". Questo comportamento può essere dovuto a un controllo prima della connessione o a un messaggio di debug. Il codice continua quindi l'esecuzione dopo il salto. Se esiste una connessione attiva, il codice sposta semplicemente 1 nel registro EAX ed esegue il codice successivo.