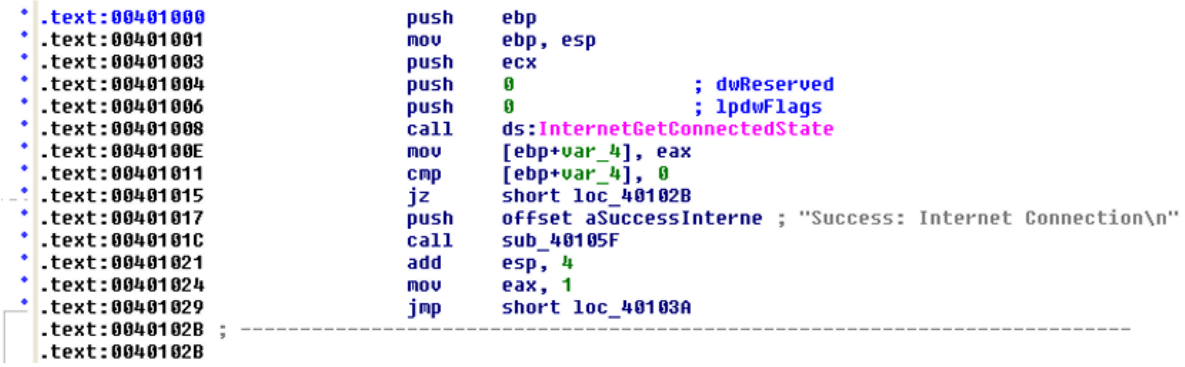
Lezione S10/L4
Rosario Giaimo

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 prende in input 3 parametri e permette di controllare se una macchina ha accesso ad Internet.
Consegna:
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. Identificare i costrutti noti (es. while, for, if, switch, ecc.)

Creazione dello Stack

```
push    ebp
mov     ebp, esp
```

I parametri sono passati sullo stack tramite le istruzioni push

```
push    ecx
push    0               ; dwReserved
push    0               ; lpdwFlags
call    ds:InternetGetConnectedState
```

Ciclo IF

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

2. Ipotizzare la funzionalità –esecuzione ad alto livello

Possiamo ipotizzare che la funzione **`internetgetconnectedstate`** controlli se un computer ha accesso a Internet.
La funzione prende in input tre parametri e restituisce un valore 0 se la connessione è attiva.

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

Questa riga salva il valore del registro **ebp** nello stack. Il registro **ebp** viene utilizzato come puntatore alla base del frame della pila corrente.

Questa riga sposta il valore del registro **esp** nel registro **ebp.** Il registro **esp** è il puntatore allo stack. Impostando **ebp** su **esp**, si stabilisce il frame della pila corrente.

Questa riga salva il valore del registro **ecx** nello stack.

Questa riga spinge il valore `0` nello stack.

Questa riga spinge un valore nello stack. Il valore effettivo non è esplicitamente codificato, ma potrebbe essere un flag che indica il tipo di connessione da controllare.

Questa riga chiama la funzione `InternetGetConnectedState`. La funzione è definita in un segmento di dati denominato `ds`.

Questa riga salva il valore del registro `eax` nella memoria, all'offset `var_4` dal registro `ebp`. L'offset `var_4` indica che il valore viene salvato a 4 byte di distanza dalla base del frame della pila.

Questa riga confronta il valore memorizzato a `[ebp+var_4]` con `0`. Se i valori sono uguali, il flag ZF viene impostato su 1.

Questa riga esegue un salto condizionale all'etichetta `loc_40102B` se il flag ZF è impostato su 1. Ciò significa che se il risultato della funzione `InternetGetConnectedState` è `0`, il codice salta alla `loc_40102B`.

Questa riga spinge l'indirizzo della stringa **"Success: Internet Connection\n"** nello stack. La stringa verrà utilizzata per stampare un messaggio di successo.

Questa riga chiama la funzione **sub_40105F**. La funzione che potrebbe stampare la stringa **"No Internet Connection\n"** sulla console.

Questa riga serve ad **aumentare il puntatore dello stack (esp) di 4 byte**.

Questa riga sposta il valore 1 nel registro chiamato EAX

Questa riga esegue un salto incondizionato a una specifica posizione di memoria