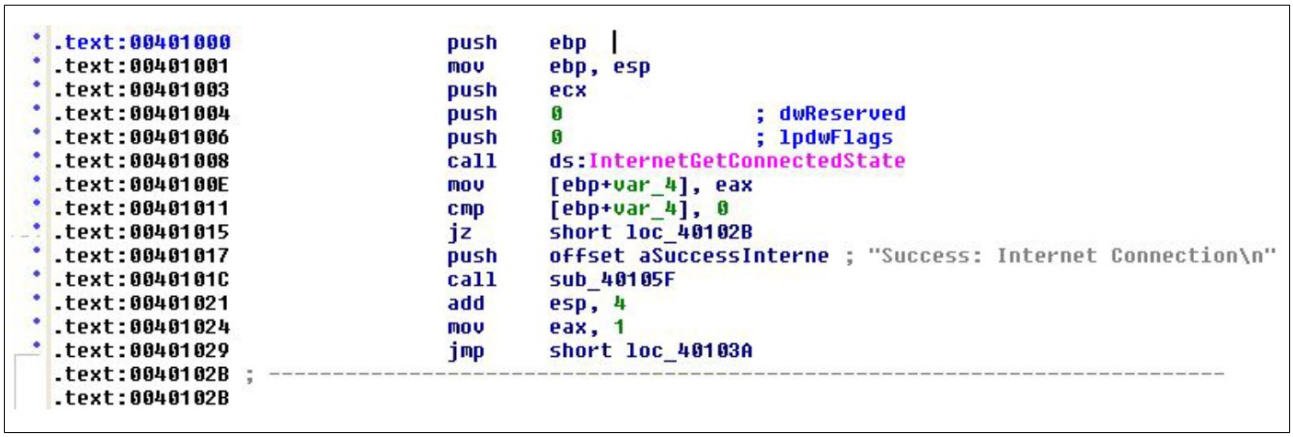
Costrutti C – Assembly x86

Obiettivo: Identificare i Costrutti di un estratto di codice Malware dato.

1) Codice

Questo è il codice che viene dato dalla traccia.



2) Identificazione dei Costrutti

La prima richiesta della traccia è di individuare i **Costrutti** presenti nel codice. Ne ho individuati in tutto Tre di cui uno sembra essere un **Ciclo IF**.

Primo Costrutto: Creazione dello Stack.

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

Secondo Costrutto: Chiamata Funzione InternetGetConnectedState

```
push    ecx
push    0
push    0
call    ds:InternetGetConnectedState
```

Terzo Costrutto: Ciclo IF dove se ZF è 1 (ovvero se il valore di [ebp+var_4] è 0) salta alla locazione di memoria indicata (loc_40102B)

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

2) Linguaggio ad alto livello

Successivamente la traccia chiede di eseguire o tentare di scrivere il codice in **Assembly** in uno ad alto livello. Analizzando il codice penso si tratti di un controllo di connessione ad Internet dato che **InternetGetConnectedState** è parte della libreria ***wininet*** (ovvero contiene le funzioni per l'implementazione di alcuni protocolli di rete).
Suppongo che una parte di codice in Linguaggio ad Alto Livello sia:

```
int main() {
    int var_4

        if (var_4 == 0){
            goto loc_40102B;        //date le istruzioni "mov e cmp"
        }
        else {
            aSuccessInterne();      //data l'istruzione "jz"
        }
    return 0;
    }
```

3) Spiegazione Linee di Codice

Come Bonus la traccia chiede di spiegare ogni istruzione elencata nel Codice proposto.

push ebp |

Inserisce nello Stack temporaneo il valore di ***ebp***.

mov ebp, esp

Assegna il Valore dello Stack Pointer a quello del Base Pointer.

push ecx
push 0
push 0

Inserisce i valori di ecx e 0 nello stack; potrebbero essere parametri della funzione che viene chiamata (call) successivamente.

call ds:InternetGetConnectedState

Viene chiamata la funzione InternetGetConnectedState. Il ds sta a significare che l'accesso è eseguito tramite data segment.

mov [ebp+var_4], eax

Assegna il valore di eax (ovvero il risultato della funzione precedente) alla variabile locale var_4.

cmp [ebp+var_4], 0

Confronta il valore 0 con quello della variabile sopracitata.

jz short loc_40102B

Se la Variabile ha un valore di 0, esegue un Salto alla locazione indicata (se ZF è 1).

push offset aSuccessInterne

Inserisce l’offset di aSuccessInterne nello stack.

call sub_40105F

Chiama la funzione indicata.

add esp, 4

Incrementa il puntatore dello stack di 4 byte.

mov eax, 1

Assegna 1 al valore del registro eax.

jmp short loc_40103A

Effettua un Salto alla locazione indicata.