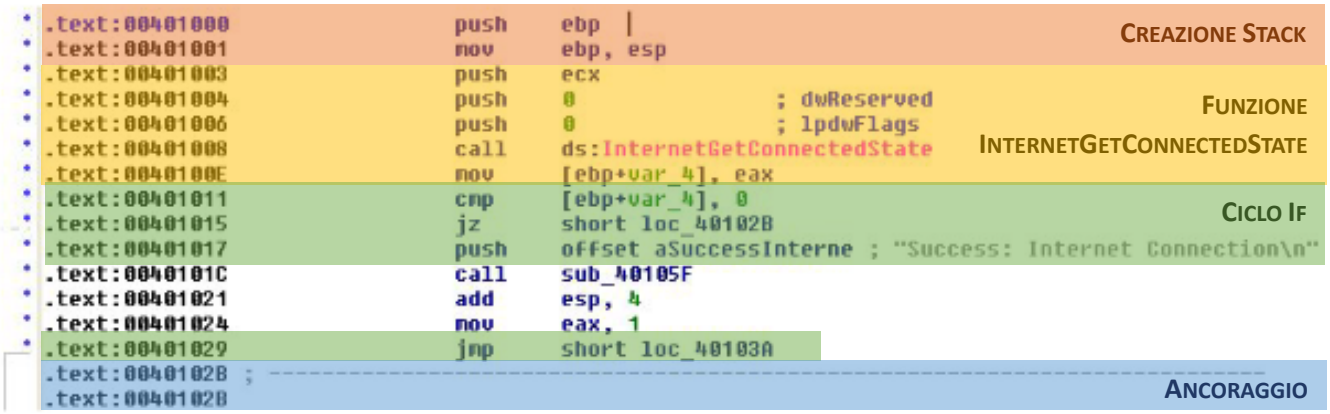
**COSTRUTTI C – ASSEMBLY X86**

**Traccia:**

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




- **push** viene usato per spingere i valori sullo stack.
- **mov** (copia) viene usato per copiare i valori tra i registri o tra un registro e una posizione di memoria.
- **call** (chiama) viene usato per chiamare una funzione o un sotto-programma.
- **jz** (jump if zero) esegue un salto condizionale se il risultato dell'ultima operazione è zero.
- **jnp** (jump if not parity) esegue un salto condizionale se il flag di parità non è impostato.
- **short** salta ad una locazione vicina nel programma.
- **cmp** (compare) confronta due operandi e imposta i flag del registro EFLAGS in base al risultato.
- **add** (addizione) viene usato per sommare due operandi e memorizzare il risultato in un registro o in una posizione di memoria.
- **ds: InternetGetConnectedState** "ds" (data segment) con etichetta “InternetGetConnectedState”.

**Prova a ipotizzare che funzionalità è implementata nel codice assembly:**
Non avendo a disposizione la totalità del codice non possiamo dire con certezza cosa fa, ma, in base alla parte presente, tramite la funzione "InternetGetConnectedState" si può afferma che si sta cercando di verificare se la macchina ha accesso a Internet.
Se è presente una connessione si può pensare di effettuare di versi tipi di azioni, dalla raccolta di informazioni sul sistema operativo, della rete, ai dati personali e sensibili, creare una connessione con un server di controllo ecc.
In caso non ci sia connessione

**CREAZIONE STACK**

**Push ebp**: spinge il valore attuale del registro EBP nello stack. Salva il valore di EBP precedente, in modo da poterlo ripristinare successivamente se necessario.

**Mov ebp, esp**: sposta il valore corrente del puntatore dello stack (ESP) nel registro EBP. EBP viene impostato allo stesso valore di ESP, creando un nuovo frame di stack per la funzione corrente. È l'inizio del frame di stack.

**CHIAMATA FUNZIONE INTERNETGETCONNECTEDSTATE**

**Push ecx**: spinge il valore del registro ECX nello stack.

**Push 0** ; dwReserve: spinge il valore 0 nello stack. ;***dwReserved*** commento in riferimento a questo parametro che è riservato e deve essere 0.

**Push 0** ; lpdwFlags: spinge il valore 0 nello stack. ;***lpdwFlags*** un commento in riferimento al puntatore a una variabile che riceve la descrizione della connessione. Questo parametro può restituire un flag valido anche quando la funzione restituisce **FALSE** .

(***dwReserved*** e ***lpdwFlags*** da https://learn.microsoft.com/en-us/windows/win32/api/wininet/nf-wininet-internetgetconnectedstate)

**Call ds:InternetGetConnectedState**: chiama la funzione "InternetGetConnectedState" indicata dal segmento di dati "ds". Questa funzione serve a verificare lo stato della connessione a Internet.

La suddetta funzione prende in input 3 parametri, sono i 3 push iniziali: ecx, 0, 0

**Mov [ebp+var_4], eax**: sposta il valore del registro EAX nella posizione di memoria [ebp+var_4]. Il registro EAX contiene il valore restituito dalla chiamata alla funzione "InternetGetConnectedState", e viene memorizzato nella variabile locale o nell'area di memoria indicata da [ebp+var_4].

## CICLO IF

**Cmp [epb+var_4], 0**: compara/confronta il valore presente all'indirizzo di memoria [ebp+var_4] con 0.

**Jz short loc_40102B**: esegue un salto condizionale alla locazione *loc_40102B* se il risultato del confronto precedente è uguale a 0 (*cioè, se [ebp+var_4] è uguale a 0 allora ZF=1*).

**Push offset a SuccessInterne**: spinge l'offset di una stringa denominata "SuccessInterne" nello stack.

**Call sub_40105F**: chiama la funzione *sub_40105F* che però non abbiamo nello screen.

**Add esp, 4**: aggiunge 4 all'ESP, spostando il puntatore dello stack verso l'alto e liberando lo spazio precedentemente occupato dai parametri.

**Mov eax, 1**: copia il valore 1 nel registro EAX.

**Jmp short loc_40103A**: esegue un salto incondizionato (short jump) alla posizione di memoria indicata da *loc_40103°* che però non abbiamo nello screen. Questo salto viene eseguito senza alcuna condizione, il che significa che il flusso di esecuzione si sposta direttamente alla posizione indicata.

Un **ANCHOR** è un oggetto appartenente alla definizione del Document Object Model di un documento HTML e permette di definire il testo di un collegamento ipertestuale. Il nome deriva proprio dal fatto che questo testo fa da "ancora" al collegamento e ne specifica solitamente il contenuto.