LINGUAGGIO ASSEMBLY – PARTE 2

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.

Esecuzione:

```
.text:00401000                push    ebp
.text:00401001                mov     ebp, esp
.text:00401003                push    ecx
```

In questa prima parte di codice il Malware effettua la creazione di uno stack in memoria.

```
.text:00401004                push    0               ; dwReserved
.text:00401006                push    0               ; lpdwFlags
.text:00401008                call    ds:InternetGetConnectedState
```

Le tre istruzioni forzano il valore “0” nello stack della CPU, aggiornando così il puntatore dello stack e di fatto eseguendo un codice malevolo.

Il comando **InternetGetConnectedState** viene chiamato con la funzione call al fine di verificare se la macchina su cui è presente il malware ha una qualsiasi connessione verso internet. I parametri prima citati (**dwReserved** e **lpdwFlags**) sono quanto seguono:

- **lpwdFlags**: Un puntatore a un valore DWORD che indica il tipo di connessione da controllare. Se impostato a 0, controlla una qualsiasi connessione disponibile.
- **dwReserved**: deve essere impostato a 0. Serve per fornire una sorta di flessibilità per le implementazioni/funzioni successive senza dover modificare l’interfaccia della funzione.

Il risultato che questa funzione porta sarà di tipo Boolean, quindi 0(false) se la connessione a internet non è presente, oppure 1 (true) se risulta esserci connessione.

```
.text:0040100E                  mov     [ebp+var_4], eax
.text:00401011                  cmp     [ebp+var_4], 0
.text:00401015                  jz      short loc_40102B
```

Di seguito:

- **mov [ebp+var_4], eax:** imposta il valore del puntatore **ebp** con offset -4 il valore contenuto in **eax**, che ipotizzo essere 0.
- **Cmp [ebp+var_4], 0:** effettua un confronto con il valore del puntatore **ebp** con offset -4 ed il valore **0.**
- **Jz short loc_40102B:** se la condizione di prima risulta effettivamente 0 (jz effettua il jump solo se la condizione è vera). In tal caso salta all'etichetta impostata che è la seguente. (ciò sta anche ad indicare che non vi è connessione)

```
.text:0040102B ; ---------------------------------------------------------------------------
```

Nel caso invece tale condizione non si verifica (quindi risulti falsa), e quindi risulta esserci connessione di fatto, il programma non effettua il jump e passa alle istruzioni con indirizzo consecutivo/seguente. Ciò che succede è il seguente.

```
.text:00401017                  push    offset aSuccessInterne ; "Success: Internet Connection\n"
.text:0040101C                  call    sub_40105F
.text:00401021                  add     esp, 4
.text:00401024                  mov     eax, 1
```

Essa restituisce un risultato "connessione internet presente" e successivamente esegue una chiamata ad una subroutine presente all'indirizzo indicato, ovvero **40105F**.

Successivamente, aggiunge al puntatore **esp** il valore 4 e imposta puntatore **eax** il valore 1, solo dopo aver eseguito il codice citato nella call precedente.

Sulla base dell'analisi eseguita in precedenza, il programma malevolo effettua un costrutto **IF (**composto dal comando cmp e condizione jz) ed una chiamata di tipo **CDECL** in quanto le istruzioni successive ripuliscono lo stack ed i registri.

Ipotizzo di tratto di un Malware che, verificata la presenza di una connessione internet, crei una backdoor con possibilità di connessione da parte di un hacker/utente malevolo.