CONSEGNA S20L4

L'obbiettivo di oggi, dato un estratto di codice di un malware, è quello di:

1. Identificare i costrutti noti (es. while, for, if, switch, creazione/distruzione stack, ecc.)
2. Ipotizzare la funzionalità –esecuzione ad alto livello
3. BONUS: studiare e spiegare ogni singola riga di codice

In Fig.1 è riportato il codice da analizzare

```
.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
```

Fig.1

Il frammento di codice sopra riportato è scritto in assembly 86, un linguaggio a basso livello usato per scrivere istruzioni direttamente eseguibili da un processore x86.

Prologo della funzione

push ebp ; Salva il valore corrente di EBP nello stack.
mov ebp, esp ; Imposta EBP per puntare alla cima attuale dello stack.

Gestione dello Stack

push ecx ; Salva il valore corrente di ECX nello stack.
push 0 ; Mette un valore 0 nello stack.
push 0 ; Mette un altro valore 0 nello stack, probabilmente questo valore corrisponde al parametro lpdwFlags (long pointer DWORD flag)

Chiamata alla funzione

call ds:InternetGetConnectedState ; Chiama la funzione del sistema operativo per verificare la connessione a Internet.

Controllo di flusso, costrutto IF

mov [ebp+var_4], eax ; Memorizza il risultato della funzione (in EAX) nella variabile locale var_4.
cmp [ebp+var_4], 0 ; Confronta il risultato salvato con 0 per vedere se si è connessi.
jz short loc_40102B ; Salta a loc_40102B se il risultato è zero (nessuna connessione).

Costrutto ELSE

push offset aSuccessInterne ; Mette l'indirizzo della stringa "Success: Internet Connection\n" nello stack.
call sub_40135F ; Chiama una sottofunzione che probabilmente stampa la stringa.

Pulizia dello Stack

add esp, 4 ; Pulisce lo stack dall'indirizzo della stringa (aggiusta lo stack pointer).
mov eax, 1 ; Imposta il valore di EAX a 1.

Termine della funzione

jmp short loc_40103A ; Salta a loc_40103A., questo salto incondizionato che potrebbe essere usato per terminare la funzione o saltare a un altro blocco di codice

lpdwFlag è un parametro che si vede spesso nelle chiamate di funzioni di Windows API che lavorano con puntatori a variabili di tipo DWORD. Il prefisso "lp" sta per "long pointer", che è un retaggio storico di quando Windows usava modelli di memoria differenti, e "dw" sta per "DWORD", che è un tipo di dato in Windows che rappresenta un intero senza segno a 32 bit.

Nel contesto della funzione **InternetGetConnectedState**, che è indicata nel frammento di codice assembly che hai fornito in precedenza, lpdwFlags sarebbe un puntatore a un DWORD che la funzione può usare per restituire informazioni aggiuntive sullo stato della connessione internet. Ad esempio, può fornire bandiere (flags) che indicano se la connessione è attraverso un modem, una LAN, un proxy, e così via.

dwReserved i indica probabilmente che l'istruzione push 0 prima di esso sta inserendo nello stack un valore che ha a che fare con un parametro o un valore riservato. Nel caso specifico di una chiamata di funzione di sistema Windows, il parametro "dwReserved" è spesso un parametro nel quale si può inserire un valore riservato per uso futuro o che deve essere impostato a zero se non utilizzato.