Epicode
CS-0124
Pratica S10/L4

Francesco Ficetti

1. Traccia

La figura seguente mostra un estratto del codice di un malware:

1. Identificare i costrutti noti.
2. Ipotizzare la funzionalità.
3. Spiegazione del codice.

```
.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
```

2. Svolgimento

2.1 Identificazione dei costrutti

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

Queste righe di codice identificano un costrutto IF:

- con l'istruzione *cmp*, si compara il valore presente nell'indirizzo di memoria *ebp+var_4*, con il valore 0. Se sono uguali lo ZF verrà impostato ad 1, altrimenti verrà impostato a 0.
- con l'istruzione *jz*, si effettua un salto alla locazione di memoria 40120B, se lo ZF è impostato a 0.

2.2 Ipotesi sul funzionamento

Questo estratto di codice sembra essere una funzione di un malware, che si occupa di controllare se la macchina su cui è eseguito è connessa o no ad Internet.

2.3 Spiegazione del codice

Le prime due righe del programma servono per creare lo stack dedicato alla funzione, diversamente dalla normale creazione di uno stack, in questo caso, non viene dedicato nessuno spazio alle variabili locali, questo permette alla funzione di avere un numero variabile di argomenti.
Alle righe 3:6 viene eseguito il *push* sullo stack, di 3 valori, che saranno passati come parametri alla funzione richiamata con l'istruzione *call*.
La funzione *InternetGetConnectedState*, restituisce il valore *False* (0), e la macchina non è connessa ad Internet, *True* (1), se invece lo è. Questo valore viene salvato nel registro *EAX*. Il valore di EAX viene copiato, con l'istruzione *mov*, nell'indirizzo di memoria *ebp+var_4*, a riga 7.
Alle righe 8:9 viene controllato se la connessione è attiva o no, in caso negativo viene effettuata una jump ad una riga al di fuori di questa parte di codice.
A riga 10 si esegue il *push*, dell'indirizzo della stringa "SuccesInterne", sullo stack.

A riga 11 viene chiamata una subroutine, tramite l'istruzione *call*, che si ipotizza stampi il messaggio salvato in precedenza sullo stack.
A riga 12 viene rimosso lo stack, eseguendo un'istruzione *add* sul registro *ESP*.
A riga 13, tramite l'istruzione *mov*, viene assegnato il valore 1 al registro EAX.
A riga 14, tramite l'istruzione *jmp*, si esegue un salto incondizionato alla locazione 40103A.