Epicode
CS-0124
Pratica S10/L5

Francesco Ficetti

1. Traccia

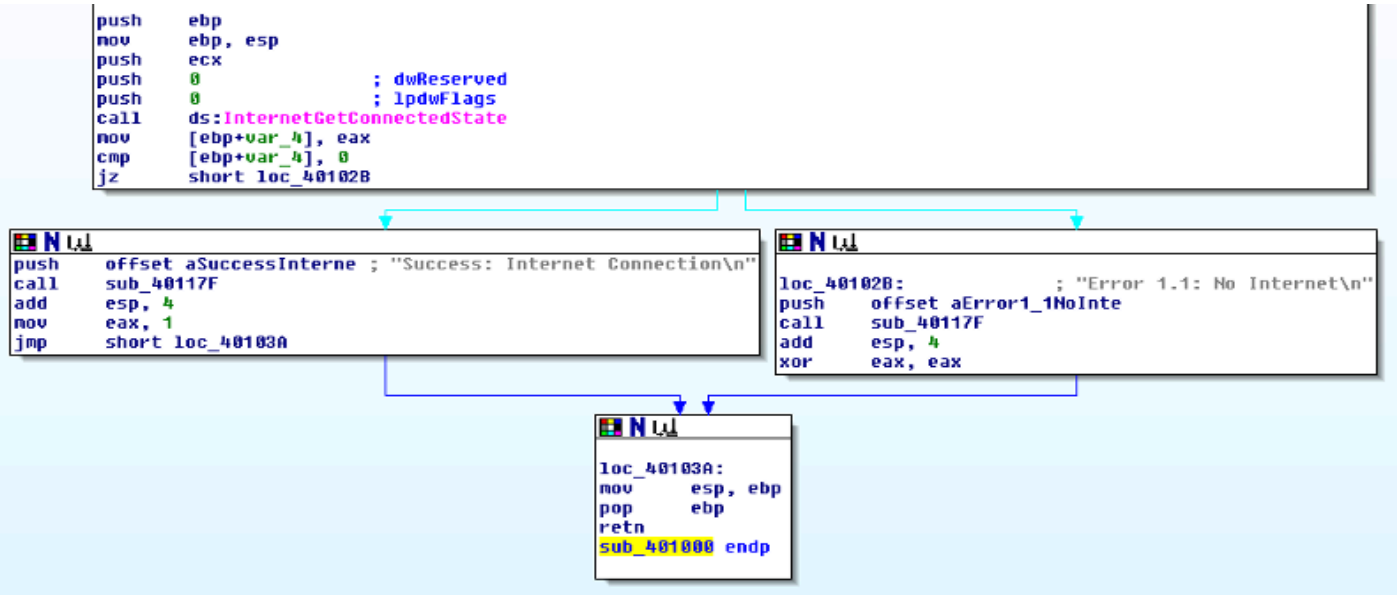
Con riferimento al malware allegato a questo esercizio, rispondere ai seguenti quesiti:

1. Quali librerie vengono importate dal file eseguibile?
2. Quali sono le sezioni di cui si compone il file eseguibile del malware?

Di entrambi dare una breve spiegazione.

Con riferimento alla figura seguente, risponde ai seguenti quesiti:

3. Identificare i costrutti noti (creazione dello stack, eventuali cicli, altri costrutti).
4. Ipotizzare il comportamento della funzionalità implementata.
5. BONUS fare tabella con significato delle singole righe di codice assembly.



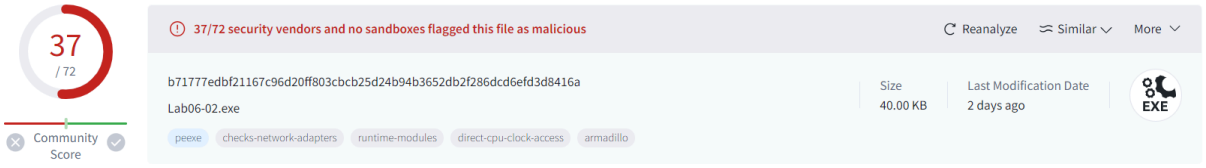
2. Svolgimento

2.1 Analisi della firma

Per analizzare la firma del file, si può utilizzare l'hash del file, che si trova su CFF Explorer.

Property	Value
File Name	C:\Users\user\Desktop\MALWARE\Esercizio_Pratico_U3_W2_L5\Malw...
File Type	Portable Executable 32
File Info	Microsoft Visual C++ 6.0
File Size	40.00 KB (40960 bytes)
PE Size	40.00 KB (40960 bytes)
Created	Wednesday 02 February 2011, 16.29.06
Modified	Wednesday 17 January 2024, 17.48.15
Accessed	Wednesday 02 February 2011, 16.29.06
MD5	C0B54534E188E1392F28D17FAFF3D454
SHA-1	BB6F01B1FEF74A9CFC83EC2303D14F492A671F3C

La stringa alfanumerica evidenziata nell'immagine, può essere analizzata da VirusTotal.



Questo file è identificato come malevolo da 37 software antivirus su 72. Nello specifico, viene identificato come trojan.

2.2 Analisi delle librerie

Una DLL è una libreria che contiene codice e dati utilizzabili da più di un programma contemporaneamente. Questo favorisce il riutilizzo del codice e l'utilizzo efficiente della memoria. Utilizzando una DLL, un programma può essere modularizzato in componenti distinti. Un modulo viene caricato solo quando tale funzionalità viene richiesta. Gli aggiornamenti possono inoltre essere applicati a ogni modulo senza influire sulle altre parti del programma.

Module Name	Imports	OFTs	TimeDateStamp	ForwarderChain	Name RVA	FTs (IAT)
szAnsi	(nFunctions)	Dword	Dword	Dword	Dword	Dword
KERNEL32.dll	44	00006518	00000000	00000000	000065EC	00006000
WININET.dll	5	000065CC	00000000	00000000	00006664	000060B4

Il malware analizzato importa due librerie:

- *Kernel32.dll*, che fornisce una vasta gamma di funzioni di basso livello, necessarie per gestire le risorse di sistema e fornire servizi di base ai programmi in esecuzione su Windows. Di questa libreria vengono importate 44 funzioni.
- *Wininet.dll*: fornisce funzionalità per la comunicazione via Internet e la gestione delle risorse di rete. Di questa libreria vengono importate 5 funzioni.

È importante sapere che non tutte le funzioni importate devono obbligatoriamente essere utilizzate all'interno del programma. Spesso, chi crea i malware, importa molte più librerie di quante ne utilizza, questo per rendere più difficoltosa l'analisi.

2.3 Analisi delle sezioni

Le sezioni all'interno di un file eseguibile sono blocchi di dati che contengono informazioni specifiche per il caricamento e l'esecuzione del programma.

Name	Virtual Size	Virtual Address	Raw Size	Raw Address	Reloc Address	Linenumbers	Relocations N...	Linenumbers ...	Characteristics
Byte[8]	Dword	Dword	Dword	Dword	Dword	Dword	Word	Word	Dword
.text	00004A78	00001000	00005000	00001000	00000000	00000000	0000	0000	60000020
.rdata	0000095E	00006000	00001000	00006000	00000000	00000000	0000	0000	40000040
.data	00003F08	00007000	00003000	00007000	00000000	00000000	0000	0000	C0000040

Il malware analizzato è formato da tre sezioni:

- *.text*, che contiene le istruzioni che la CPU eseguirà una volta che il software sarà avviato.
- *.rdata*, che contiene le informazioni circa le librerie e le funzioni importate ed esportate dall'eseguibile.
- *.data*, che contiene le variabili globali del programma. Le variabili locali, invece, vengono salvate nello stack che si crea quando si invoca una funzione.

Gran parte di queste sezioni, potrebbe, come nel nostro caso, essere cifrata o incomprensibile. Questo per rendere più difficoltosa l'analisi.

2.4 Analisi delle stringhe

Le stringhe del malware sono state estratte tramite l'utility *strings*.

```
__GLOBAL_HEAP_SELECTED
__MSVCRT_HEAP_SELECT
runtime error
TLOSS error
SING error
DOMAIN error
R6028
- unable to initialize heap
R6027
- not enough space for lowio initialization
R6026
- not enough space for stdio initialization
R6025
- pure virtual function call
R6024
- not enough space for _onexit/atexit table
R6019
- unable to open console device
R6018
- unexpected heap error
R6017
- unexpected multithread lock error
R6016
- not enough space for thread data
abnormal program termination
R6009
- not enough space for environment
R6008
- not enough space for arguments
R6002
- floating point not loaded
Microsoft Visual C++ Runtime Library
Runtime Error!
Program:
...
<program name unknown>
GetLastActivePopup
GetActiveWindow
MessageBoxA
user32.dll
```

Le stringhe non sono altro che il contenuto delle sezioni viste nel paragrafo precedente. Le stringhe qui riportate sono le sole scritte in formato leggibile. Da esse si può presupporre che il malware cercherà di connettersi ad Internet, probabilmente per scaricare altri malware. Questo comportamento identifica un malware di tipo *downloader*.

```
Sleep
KERNEL32.dll
InternetGetConnectedState
InternetReadFile
InternetCloseHandle
InternetOpenUrlA
InternetOpenA
WININET.dll
GetCommandLineA
GetVersion
ExitProcess
TerminateProcess
GetCurrentProcess
UnhandledExceptionFilter
GetModuleFileNameA
FreeEnvironmentStringsA
FreeEnvironmentStringsW
WideCharToMultiByte
GetEnvironmentStrings
GetEnvironmentStringsW
SetHandleCount
GetStdHandle
GetFileType
GetStartupInfoA
GetModuleHandleA
GetEnvironmentVariableA
GetVersionExA
HeapDestroy
HeapCreate
VirtualFree
HeapFree
RtlUnwind
WriteFile
HeapAlloc
GetCPInfo
GetACP
GetOEMCP
VirtualAlloc
HeapReAlloc
GetProcAddress
LoadLibraryA
GetLastError
FlushFileBuffers
SetFilePointer
MultiByteToWideChar
LCMapStringA
LCMapStringW
GetStringTypeA
GetStringTypeW
SetStdHandle
CloseHandle
d5@
Error 1.1: No Internet
Success: Internet Connection
Error 2.3: Fail to get command
Error 2.2: Fail to ReadFile
Error 2.1: Fail to OpenUrl
http://www.practicalmalwareanalysis.com/cc.htm
Internet Explorer 7.5/pma
Success: Parsed command is %c
```

2.5 Identificazione dei costrutti

I costrutti principali sono tre.

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

Creazione di uno stack.

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

Costrutto IF.

```
mov     esp, ebp
pop     ebp
```

Rimozione dello stack.

2.6 Ipotesi sul comportamento

Questo estratto del malware, controlla, tramite la funzione *InternetGetConnectedState*, se la macchina su cui è eseguito è connessa o meno ad Internet.
In caso positivo, stampa la stringa “SuccesInterne”.
In caso negativo, stampa la stringa “Error1_1NoInterne”.

2.7 Spiegazione del codice

Istruzione	**Descrizione**
push ebp	Crea un nuovo stack
mov ebp, esp	Copia il contenuto del registro esp, nel registro ebp
push ecx	Effettua il push del valore del registro ecx, nello stack
push 0	Effettua il push del valore 0, nello stack
push 0	Effettua il push del valore 0, nello stack
call ds.InternetGetConnectedState	Chiama la funzione InternetGetConnectedState e gli passa come parametri i tre valori precedentemente salvati nello stack
mov [ebp+var_4], eax	Copia il valore di ritorno della funzione InternetGetConnectedState, contenuto nel registro EAX, all’interno della variabile ebp+var_4

cmp [ebp+var_4], 0	Compara il valore 0, con il valore contenuto all'interno della variabile ebp+var_4
jz short loc_40102B	Salta alla locazione indicata, se i due valori comparati precedentemente sono uguali
push offset aSuccesInterne	Effettua il push dell'indirizzo della stringa "SuccesInterne", nello stack
call sub_40117F	Chiama la subroutine indicata e gli passa come parametro il il valore precedentemente salvato sullo stack
add esp, 4	Chiude lo stack aperto in precedenza
mov eax, 1	Copia il valore 1 all'interno del registro eax
jmp short loc_40103A	Effettua un salto incondizionato alla locazione indicata
loc_40102B:	Identifica la locazione di memoria
push offset aError1_1NoInterne	Effettua il push dell'indirizzo della stringa "Error1_1NoInterne", nello stack
call sub_40117F	Chiama la subroutine indicata e gli passa come parametro il il valore precedentemente salvato sullo stack
add esp, 4	Chiude lo stack aperto in precedenza
xor eax, eax	Azzera il valore del registro eax, effettuando un'operazione di xor con se stesso
loc_40103A:	Identifica la locazione di memoria
mov esp, ebp	Copia il valore del registro ebp, nel registro esp
pop ebp	Chiude lo stack
retn	Termina la subroutine e ritorna al punto della sua chiamata
sub: 401000 endp	Indica la fine della subroutine