CONSEGNA 2 APRILE S11L1

L'obbiettivo di oggi, con riferimento agli estratti di un malware reale riportato in Fig.1 / Fig.2, è quello di rispondere alle seguenti domande:

1) Descrivere come il malware ottiene la persistenza, evidenziando il codice in assembly dove le relative istruzioni e chiamate di funzione vengono eseguite.
2) Identificare il client software utilizzato dal malware per per la connessione a internet.
3) 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
4) BONUS: Qual'è il significato ed il funzionamento del comando assembly "lea"

```
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
```

Fig.1

```
.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
```

Fig.2

1) Questo codice di un malware apre una chiave di registro per aggiungere un valore in modo da ottenere la **persistenza:** un malware, utilizzando un registro, ottiene la persistenza per per aggiungersi alle entry dei programmi che vengono avviati all'avvio in modo da essere eseguito in maniera automatica.

Come vediamo in Fig.1. il malware si serve di due funzioni che fanno parte della libreria **Advapi32.dll,** una libreria che contiene funzioni relative alla gestione avanzata di Windows., queste funzioni sono:

RegOpenKeyeEx: Questa funzione viene utilizzata per aprire una chiave di registro esistente. La funzione prende vari parametri che specificano la chiave di registro da aprire, il livello di accesso desiderato, e altre opzioni. Questi parametri consentono di specificare come si desidera accedere alla chiave di registro. In Fig.3 vediamo come i parametri della funzione sono passati sullo Stack tramite le istruzioni "push". Con questa funzione il malware accede alla chiave di registro prima di modificarne il valore.

```
)040286F   push    2                   ; samDesired
)0402871   push    eax                 ; ulOptions
)0402872   push    offset SubKey       ; "Software\\Microsoft\\Windows\\CurrentVersion\\Run"
)0402877   push    HKEY_LOCAL_MACHINE ; hKey
```

Fig.3

Importante notare la chiave di registro che viene utilizzata dal malware:
Software\\Microsoft\\Windows\\CurrentVersion\\Run

RegSetValueEx: Questa funzione è usata per impostare o modificare i dati di un valore all'interno di una chiave di registro aperta (tramite RegOpenKeyEx). Questo e cruciale per la configurazione del comportamento del sistema operativo e delle applicazioni, poiché consente alle applicazioni di salvare le impostazioni o ai componenti del sistema di aggiornare le configurazioni del sistema. In Fig.4 vediamo che questo codice sta preparando una chiamata alla funzione RegSetValueEx inserendo due dei suoi parametri (lpValueName, hKey) nello stack e poi effettuando una chiamata. La funzione utilizzerà questi parametri per impostare un valore all'interno di una chiave di registro specificata (HKEY_LOCAL_MACHINE)

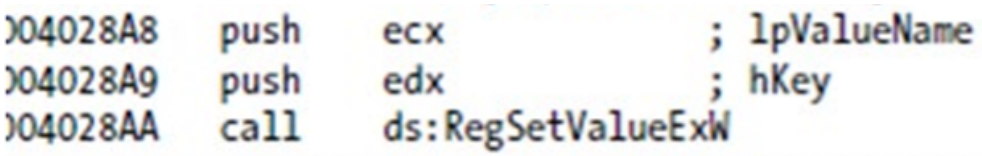

Fig.4

2) Come mostrato il Fig.5, il client usato per connettersi ad internet è Internet Explorer 8.0

Fig.5

3) Tramite la libreria **Wininet.dll**, che include le funzioni per l'implementazione di protocolli di rete com HTTP e FTP, il malware può stabilire una connessione a internet o ad un determinato URL. Nello specifico le funzioni sono:

InternetOpen: Inizializza una connessione a internet

InternetOpenUrl: Utilizzato per la connessione ad un determinato URL

Nel caso del nostro codice, il malware tenta di connettersi a "http://www.malware12.com".

In Fig.6 vediamo la chiamata di funzione, **InternetOpenUrl**, che consente al malware di connettersi verso l'URL appena citato, che passa come parametro di questa funzione sullo stack tramite l'istruzione "push".

Fig.6

4) **LEA - Load Effective Address**

Questa istruzione è simile all'istruzione mov, ma più potente perché può eseguire operazione aritmetiche durante il calcolo dell'indirizzo. Viene quindi utilizzato per calcolare l'indirizzo di un operando e inserirlo nel registro.

Come nell'esempio di Fig.7. l'istruzione **lea** calcola un indirizzo sommando l'offset 434h (**offset** è la differenza o la distanza tra due indirizzi) all'indirizzo contenuto nel registro esp, e aggiunge l'indirizzo di una variabile chiamata ValueName. Il risultato viene poi inserito nel registro ecx.

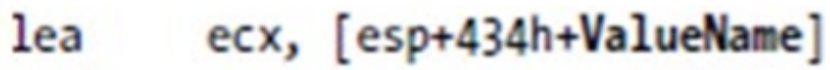

Fig.7