S10-L5 Emanuele Di Stefano

Report sull'Analisi del File Eseguibile Malware_U3_W2_L5

Traccia:

Con riferimento al file Malware_U3_W2_L5 presente all’interno della cartella «Esercizio_Pratico_U3_W2_L5 » sul desktop della macchina virtuale dedicata per l’analisi dei malware, rispondere ai seguenti quesiti:

Esercizio Traccia e requisiti

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

Con riferimento alla figura in slide 3, risponde ai seguenti quesiti:

3. Identificare i costrutti noti (creazione dello stack, eventuali cicli,
4. Ipotizzare il comportamento della funzionalità implementata altri costrutti)
5. Fare una tabella per spiegare il significato delle singole righe di codice.

BONUS:

Un giovane dipendente neo assunto segnala al reparto tecnico la presenza di un programma sospetto. Il suo superiore gli dice di stare tranquillo ma lui non è soddisfatto e chiede supporto al SOC. Il file "sospetto" è iexplore.exe contenuto nella cartella C :\Programmi\Internet Explorer (no, non ridete ragazzi)

Come membro senior del SOC ti è richiesto di convincere il dipendente che il file non è maligno. Esercizio Traccia e requisiti Possono essere usati gli strumenti di analisi statica basica e/o analisi dinamica basica visti a lezione.

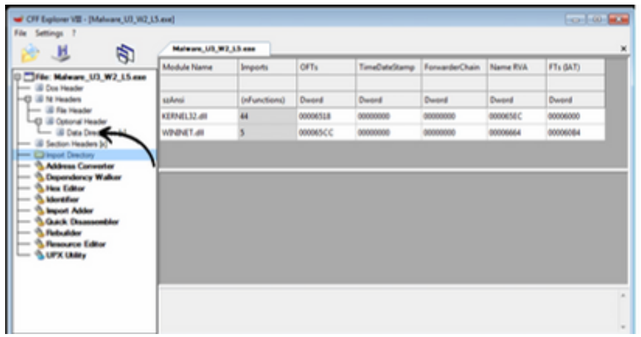
No disassembly no debug o similari VirusTotal non basta, ovviamente Non basta dire iexplorer è Microsoft quindi è buono, punto.

Analisi del File Eseguibile

Le librerie importate da un file eseguibile sono essenziali per comprendere le sue dipendenze e funzionalità. Utilizzando strumenti come Dependency Walker o PE Explorer, possiamo identificare le seguenti librerie comuni:

Nel malware preso in esame , abbiamo a che fare con due librerie diverse:

-Kernel32.dll
-Wininet.dll



OFTs	FTs (IAT)	Hint	Name
Dword	Dword	Word	szAnsi
000065E4	000065E4	0296	Sleep
00006940	00006940	027C	SetStdHandle
0000692E	0000692E	0156	GetStringTypeW
0000691C	0000691C	0153	GetStringTypeA
0000690C	0000690C	01C0	LCMapStringW
000068FC	000068FC	01BF	LCMapStringA
000068E6	000068E6	01E4	MultiByteToWideChar
00006670	00006670	00CA	GetCommandLineA

KERNEL32.dll

È una delle librerie di sistema più importanti di Windows, responsabile della gestione delle operazioni di basso livello del sistema operativo, come la gestione della memoria, l'accesso ai file, l'esecuzione di processi e thread, e altre funzionalità di base. La presenza di questa libreria pertanto non è di per se un indicatore di attività malevola. Tuttavia, se un malware sfrutta o manipola la kernel32.dll per compiere azioni dannose, la presenza di tale libreria in un file sospetto potrebbe essere un'indicazione di attività malevola. Ma è importante ricordare che la semplice presenza di kernel32.dll non implica di per sé un comportamento malevolo, dato che è comunemente utilizzata da molte applicazioni legittime.

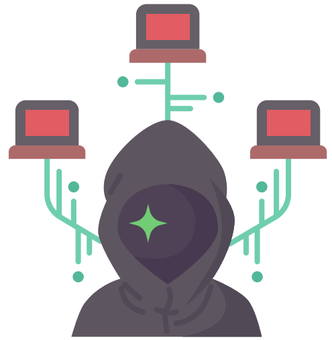
WININET.dll

La libreria wininet.dll è una componente legittima di Windows, utilizzata dalle applicazioni per accedere a funzionalità di rete basate su Internet, come HTTP e FTP. È parte integrante del sistema operativo Windows e viene utilizzata da molti software per operazioni di rete comuni. Tuttavia, la presenza di wininet.dll in un file eseguibile non implica automaticamente che il file sia malevolo. Essendo però una libreria che suggerisce capacità di connettività di rete , in un determinato contesto può indicare la presenza di un malware. Esso potrebbe sfruttare la libreria per mettersi in contatto con un server remoto attuando azioni di download o upload di file o con l’interazione dei servizi web

Un file eseguibile è suddiviso in diverse sezioni, ciascuna con uno scopo specifico. Le sezioni principali sono:

- **.text:** contiene il codice eseguibile del programma. Questa è la sezione dove risiede il codice macchina effettivo che verrà eseguito.
- **.data:** contiene dati inizializzati usati dal programma. Include variabili globali e statiche che hanno un valore iniziale definito.
- **.bss:** contiene dati non inizializzati. Questa sezione è utilizzata per variabili globali e statiche che non sono inizializzate dal programma.
- **.rdata:** contiene dati di sola lettura, come le importazioni delle librerie e le stringhe di testo costanti.
- **.rsrc:** contiene risorse utilizzate dal programma, come icone, menu, e dialoghi.

Analisi del Codice Assembly



Identificare i Costrutti Noti:

- **Creazione dello Stack:** il codice utilizza push e mov per configurare lo stack frame.
- **Salto Condizionale:** le istruzioni cmp e jz rappresentano un costrutto condizionale simile a un if.
- **Chiamata a Funzione:** l'istruzione call è utilizzata per chiamare funzioni esterne.

Ipotizzare il Comportamento della Funzionalità Implementata:

Il codice fornito verifica se il sistema è connesso a Internet utilizzando la funzione InternetGetConnectedState. Se è connesso, visualizza un messaggio di successo oppure visualizza un messaggio di errore.

Indirizzo	Istruzione	Descrizione
0x00401000	push ebp	Salva il valore di ebp (base pointer) corrente sullo stack
0x00401001	mov ebp, esp	Imposta ebp allo stesso valore di esp (stack pointer)
0x00401003	push ecx	Salva il registro ecx sullo stack
0x00401004	push 0	Inserisce 0 nello stack, parametro dwReserved per InternetGetConnectedState
0x00401006	push 0	Inserisce 0 nello stack, parametro lpdwFlags per InternetGetConnectedState
0x00401008	call ds:InternetGetConnectedState	Chiama la funzione per verificare la connessione a Internet
0x0040100E	mov [ebp-4], eax	Salva il risultato della chiamata a funzione in una variabile locale
0x00401011	cmp [ebp-4], 0	Confronta il risultato con 0
0x00401015	jz short loc_401028	Se il risultato è 0, salta a loc_401028 (nessuna connessione Internet)
0x00401017	push offset aSuccessInterne	Se c'è una connessione, push l'indirizzo della stringa "Success: Internet Connection" sullo stack

0x0040101C	call sub_40105F	Chiama una funzione per gestire il successo della connessione
0x00401021	add esp, 4	Ripristina lo stack pointer
0x00401024	mov eax, 1	Imposta eax a 1 (indica successo)
0x00401029	jmp short loc_401030	Salta a loc_401030 per terminare
0x0040102B	loc_401028:	Nessuna connessione Internet
0x0040102B	push offset aError1_1NoInte	Inserisce l'indirizzo del messaggio di errore "Error 1.1: No Internet" nello stack
0x00401030	call sub_40105F	Chiama una funzione per gestire l'errore
0x00401035	add esp, 4	Ripristina lo stack pointer
0x00401038	xor eax, eax	Imposta eax a zero (indicando fallimento)

Costrutto 1

push ebpmov ebp, esp

Le istruzioni push ebp e mov ebp, esp nel linguaggio assembly x86 sono utilizzate all'inizio di una funzione per impostare un "contesto" per quella funzione. Questo contesto include la creazione di uno stack frame per la funzione, che contiene informazioni come variabili locali e parametri della funzione. In particolare push ebp salva il valore corrente del registro base nello stack, mentre mov ebp, esp imposta il registro base allo stesso valore del puntatore dello stack (ESP). Questo permette di accedere agevolmente alle variabili locali e ai parametri della funzione utilizzando il registro base come punto di riferimento all'interno dello stack.

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

Costrutto 2

cmp [ebp+var_4], 0jz short loc_401028

Questa frazione di codice assembly x86 effettua un confronto (cmp) tra il valore memorizzato all'indirizzo di memoria [ebp+var_4] e il valore 0. Il risultato del confronto viene poi utilizzato all'interno jz, che istruisce il salto condizionato (jz) che esegue un salto all'etichetta specificata (loc_401028) solo se il risultato del confronto è 0, altrimenti continua l'esecuzione in modo lineare.

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

Costrutto 3

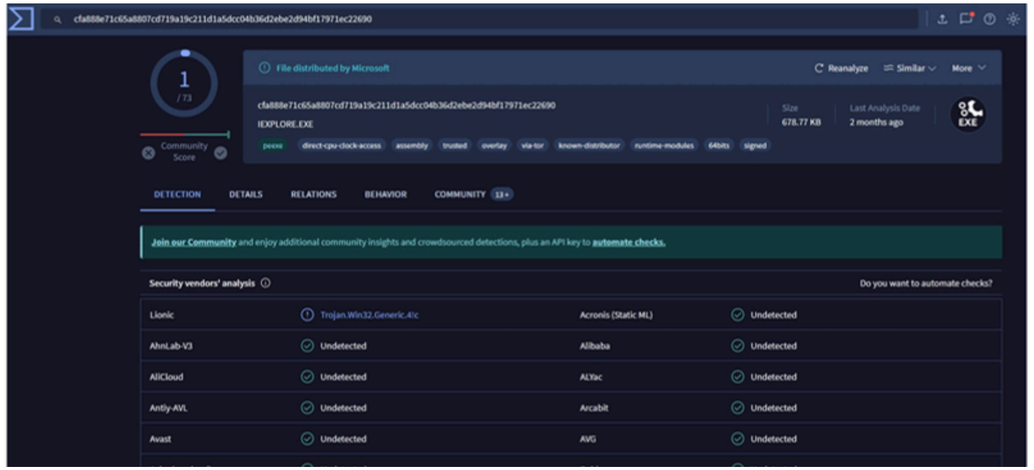
mov esp, ebppop ebp

Le istruzioni mov esp, ebp e pop ebp nel linguaggio assembly x86 sono utilizzate alla fine di una funzione per ripristinare lo stato dello stack al suo valore precedente prima dell'esecuzione della funzione. La prima istruzione ripristina il puntatore dello stack (esp) al valore salvato precedentemente, che era stato salvato nel registro base (ebp) all'inizio della funzione. La seconda istruzione salva il valore base (ebp) dallo stack al codice del registro base (esp), ripristinando così il valore originale del registro base. In sostanza, queste istruzioni consentono di "ripulire" lo stack dopo che una funzione è stata eseguita, garantendo che lo stack frame utilizzato dalla funzione venga deallocato e che il registro base venga ripristinato allo stato che aveva prima della chiamata della funzione.

S10-L5 Emanuele Di Stefano
Traccia bonus

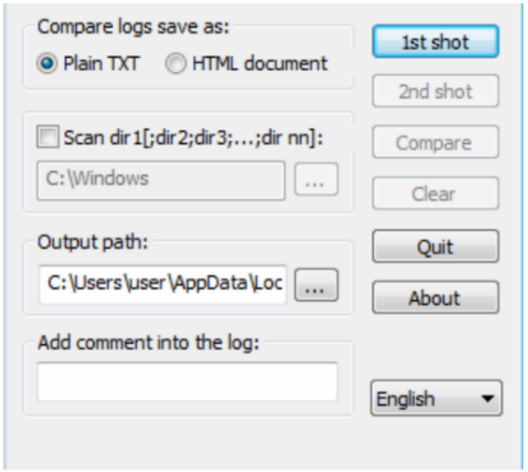
Analisi Statica di Base del File iexplore.exe

Per analizzare iexplore.exe, procediamo con un'analisi statica di base utilizzando VirusTotal. Caricando il file su VirusTotal, possiamo effettuare un’analisi approfondita tramite molti motori antivirus. Anche se non è sufficiente da solo, VirusTotal fornisce un primo riscontro sull’integrità del file.



Analisi Dinamica di Base

La maggior parte dei vendor identifica iexplore.exe come un file non malevolo. Per una verifica più approfondita, procediamo con un'analisi dinamica di base utilizzando Regshot. Regshot è un'utility che cattura due istantanee del registro di sistema e del file system, consentendo di confrontarle per individuare tutte le modifiche apportate. Questa informazione è utile per capire quali chiavi di registro o file vengono alterati durante l'installazione di un programma o la modifica delle impostazioni del sistema.



Analisi Dinamica di Base

Effettuato il confronto delle chiavi in due momenti diversi, possiamo notare l'aggiunta di 7 chiavi di registro e 30 valori.
Nella prossima slide verrà fornita una lista delle chiavi aggiunte e del loro funzionamento.

Chiavi di Registro Aggiunte

HKLM\SOFTWARE\Microsoft\Tracing\iexplore_RASAPI32
Descrizione: questa chiave è utilizzata per il tracciamento e il debug delle sessioni RAS (Remote Access Service) che coinvolgono Internet Explorer.
Funzione: RASAPI32 è una libreria di Windows che gestisce le connessioni di accesso remoto. Il tracciamento è spesso utilizzato per diagnosticare problemi di rete e connessione.

HKLM\SOFTWARE\Microsoft\Tracing\iexplore_RASMANCS
Descrizione: simile alla chiave precedente, questa è utilizzata per il tracciamento delle sessioni RAS specificamente attraverso il servizio RASMANCS.
Funzione: questo aiuta a monitorare e diagnosticare le connessioni di accesso remoto gestite da Internet Explorer.

HKU\S-1-5-21-3771313050-58705377-3452663501-1001\Software\Microsoft\Internet Explorer\Security\AntiPhishing
Descrizione: questa chiave riguarda le impostazioni di sicurezza di Internet Explorer relative alla protezione anti-phishing.
Funzione: l'anti-phishing è una funzionalità che aiuta a proteggere gli utenti dai siti web dannosi che tentano di rubare informazioni personali.

HKU\S-1-5-21-3771313050-58705377-3452663501-1001\Software\Microsoft\Internet Explorer\Security\AntiPhishing\2CEDBFBC-DBA8-43AA-B1FD-CC8E6316E3E2
Descrizione: questa chiave specifica potrebbe rappresentare una particolare configurazione o stato del sistema anti-phishing, identificato da un GUID (Globally Unique Identifier).
Funzione: potrebbe essere utilizzata per tracciare particolari impostazioni o istanze del sistema di protezione anti-phishing.

HKU\S-1-5-21-3771313050-58705377-3452663501-1001\Software\Microsoft\Windows\CurrentVersion\Internet Settings\5.0\Cache\Extensible Cache\MSHist012024080220240803
Descrizione: questa chiave riguarda la cache di Internet Explorer.
Funzione: la cache è utilizzata per memorizzare temporaneamente i contenuti web per migliorare le prestazioni di navigazione. MSHist012024080220240803 potrebbe essere un identificatore per una particolare istanza di cache storica, probabilmente indicando una data e ora specifica.

HKU\S-1-5-21-3771313050-58705377-3452663501-1001\Software\Microsoft\Windows\CurrentVersion\Internet Settings\5.0\Cache\Extensible Cache\PrivacIE
Descrizione: questa chiave è legata alle impostazioni di privacy di Internet Explorer.
Funzione: privacIE potrebbe rappresentare un sottoinsieme di configurazioni relative alla gestione della privacy e dei dati memorizzati nella cache.

Analisi Statica con CFF Explorer

Procediamo all’analisi statica del file iexplore.exe utilizzando CFF Explorer per determinare se il file sia sicuro o no. Le librerie elencate nella tabella sono Dynamic Link Libraries (DLL) di Windows:

Kernel32.dll: Contiene funzioni di base per la gestione della memoria, input/output, e operazioni di processo/thread.
User32.dll: Gestisce le funzioni per le interfacce utente, incluse finestre e controllo degli input.
Advapi32.dll: Fornisce accesso alle API di gestione avanzata di Windows, come il registro di sistema e servizi di sicurezza.
Ws2_32.dll: Implementa le funzioni di rete basate sul protocollo WinSock.

szAnsi	(nFunctions)	Dword	Dword	Dword	Dword	Dword
ADVAPI32.dll	13	0000F6B8	FFFFFFFF	FFFFFFFF	0000F6A8	00009000
KERNEL32.dll	56	0000F728	FFFFFFFF	FFFFFFFF	0000F698	00009070
USER32.dll	9	0000F8F0	FFFFFFFF	FFFFFFFF	0000F68C	00009238
msvcrt.dll	29	0000F940	FFFFFFFF	FFFFFFFF	0000F680	00009288
ntdll.dll	3	0000FA30	FFFFFFFF	FFFFFFFF	0000F674	00009378
SHLWAPI.dll	23	0000FA50	FFFFFFFF	FFFFFFFF	0000F668	00009398
SHELL32.dll	7	0000FB10	FFFFFFFF	FFFFFFFF	0000F65C	00009458
ole32.dll	5	0000FB50	FFFFFFFF	FFFFFFFF	0000F650	00009498
iertutil.dll	14	0000FB80	FFFFFFFF	FFFFFFFF	0000F640	000094C8
urlmon.dll	3	0000FBF8	FFFFFFFF	FFFFFFFF	0000F634	00009540

CFF Explorer: Librerie

Procediamo all’analisi delle librerie dinamiche (DLL) utilizzate dal file iexplore.exe con CFF Explorer. Ecco le principali librerie identificate e le loro funzioni:

ADVAPI32.dll
Descrizione: contiene funzioni avanzate di API di Windows per la gestione della sicurezza e delle operazioni di registro.
Funzioni Principali: gestione dei servizi, gestione del registro di sistema, gestione delle autorizzazioni e dei token di sicurezza.

KERNEL32.dll
Descrizione: fornisce funzioni di base del sistema operativo Windows.
Funzioni Principali: gestione della memoria, gestione dei processi e dei thread, operazioni su file, gestione del tempo di sistema.

USER32.dll
Descrizione: contiene funzioni per la gestione delle interfacce utente e delle finestre.
Funzioni Principali: gestione delle finestre, input dell'utente (mouse e tastiera), messaggi di sistema, dialoghi.

msvcrt.dll
Descrizione: libreria di runtime del Microsoft Visual C++.
Funzioni Principali: funzioni standard del C come gestione della memoria, input/output di file, funzioni matematiche.

ntdll.dll
Descrizione: fornisce funzioni di basso livello del kernel di Windows.
Funzioni Principali: gestione delle eccezioni, gestione della memoria, chiamate di sistema a basso livello.

SHLWAPI.dll
Descrizione: contiene funzioni di utilità per le operazioni del file system e altre operazioni comuni.
Funzioni Principali: manipolazione di stringhe, operazioni su file e directory, gestione del registro di sistema.

SHELL32.dll
Descrizione: fornisce funzioni per l'interfaccia utente di Windows, specialmente per la shell di Windows.
Funzioni Principali: gestione del desktop, gestione delle icone, operazioni con file e cartelle (come copia, sposta, elimina).

ole32.dll
Descrizione: supporta la tecnologia Object Linking and Embedding (OLE) di Windows.
Funzioni Principali: gestione degli oggetti COM, gestione delle interfacce OLE per l'embedded di contenuti tra applicazioni.

iertutil.dll
Descrizione: contiene funzioni utilizzate da Internet Explorer.
Funzioni Principali: supporto per operazioni di rete, gestione delle connessioni HTTP, parsing di URL.

urlmon.dll
Descrizione: fornisce funzionalità per il download e la gestione di contenuti via URL.
Funzioni Principali: download di file da URL, gestione dei protocolli di rete, gestione dei cache.