

.5'

02.08.2025

REPORT PROGETTO S10/L5

ADAM DERRO



ESERCIZIO N.1

INTRODUZIONE

Analisi statica del malware identificato come **Malware_U3_W2_L5.exe** utilizzando il programma **CFF Explorer**. L'analisi si concentra sull'identificazione delle librerie importate dal file eseguibile e sulle sezioni del file per comprendere meglio il comportamento e le potenziali minacce associate a questo malware. Attraverso l'analisi delle funzioni importate dal modulo **KERNEL32.dll**, si cerca di identificare le tecniche utilizzate dal malware per nascondere le proprie attività

1. LIBRERIE IMPORTATE

Dall'analisi della sezione Import Directory del malware, si osserva che esso importa due librerie principali:

1. KERNEL32.dll

Descrizione: Contiene funzioni fondamentali di sistema utilizzate per la gestione della memoria, il file system e altre operazioni essenziali a livello di sistema

2. WININET.dll

Descrizione: Fornisce funzioni per l'accesso alle risorse Internet

Module Name	Imports	OFTs	TimeStamp	ForwarderChain
szAnsi	(nFunctions)	Dword	Dword	Dword
KERNEL32.dll	44	00006518	00000000	00000000
WININET.dll	5	000065CC	00000000	00000000

2. SEZIONI DEL FILE ESEGUIBILE

L'analisi delle sezioni del file eseguibile rivela tre sezioni principali:

1 .text

Descrizione: Contiene il codice eseguibile del malware.

2 .rdata

Descrizione: Contiene dati di sola lettura, come stringhe e tavole di importazione/esportazione.

3 .data

Descrizione: Contiene dati inizializzati utilizzati dal malware.

Name	Virtual Size	Virtual Address	Raw Size	Raw Address	Reloc Address
Byte[8]	Dword	Dword	Dword	Dword	Dword
.text	00004A78	00001000	00005000	00001000	00000000
.rdata	0000095E	00006000	00001000	00006000	00000000
.data	00003F08	00007000	00003000	00007000	00000000

3. FUNZIONI IMPORTATE DAL KERNEL32.DLL

Dall'analisi delle funzioni importate, le seguenti sono particolarmente indicative delle capacità di nascondere le attività del malware:

1. GetProcAddress

Descrizione: Recupera l'indirizzo di una funzione esportata in modo dinamico, evitando di dichiarare le funzioni direttamente nella tabella delle importazioni. Questo potrebbe complicare il rilevamento del malware.

0000688E	0000688E	013E	GetProcAddress
----------	----------	------	----------------

2. LoadLibraryA

Descrizione: Carica una DLL a runtime, espandendo le capacità del malware senza che le librerie siano visibili nelle importazioni iniziali

000068A0	000068A0	01C2	LoadLibraryA
----------	----------	------	--------------

3. Sleep

Descrizione: Introduce ritardi nelle operazioni del malwar in questo modo potrebbe eludere l'analisi automatica

000065E4	000065E4	0296	Sleep
----------	----------	------	-------

4. TerminateProcess

Descrizione: potrebbe essere utilizzato per terminare processi di sicurezza, come antivirus, consentendo al malware di operare indisturbato.

0000669E

0000669E

029E

TerminateProcess

CONCLUSIONE

*Le funzioni importate da **KERNEL32.dll** indicano che il malware è progettato per interagire con il sistema a livello profondo, potenzialmente alterando file di sistema, espandendo le sue capacità in modo furtivo, e probabilmente puntando ad neutralizzando processi di sicurezza. L'uso di funzioni come **GetProcAddress** e **LoadLibraryA** possono anche consentire al malware di rimanere nascosto fino a quando non è pronto a eseguire le sue operazioni malevole. Queste caratteristiche suggeriscono che lo scopo del malware potrebbe includere il furto di dati sensibili, la compromissione del sistema, e il mantenimento della persistenza senza essere rilevato.*





ESERCIZIO N.2

INTRODUZIONE

In questo esercizio, analizzeremo un frammento di codice assembly. L'obiettivo è identificare i costrutti noti del codice, spiegare dettagliatamente ogni istruzione e formulare ipotesi sul comportamento complessivo della routine.

1. IDENTIFICAZIONE COSTRUTTI

Creazione dello Stack:

push ebp e mov ebp, esp: Queste istruzioni configurano un nuovo frame di stack salvando il valore del base pointer (`ebp`) e aggiornandolo con il valore corrente dello stack pointer (`esp`).

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

Chiamata di Funzione:

I parametri sono passati sullo stack tramite funzione di push

call ds:InternetGetConnectedState: Invoca la funzione che verifica lo stato della connessione Internet.

```
push    0          ; dwReserved  
push    0          ; lpdwFlags  
call    ds:InternetGetConnectedState
```

Branching Condizionale "if/else" (jump if zero):

cmp [ebp+var_4], 0:

- Descrizione: Questa istruzione confronta il valore contenuto nella variabile locale var_4 con il valore 0

jz short loc_401028:

- Descrizione: Questa istruzione è un salto condizionale. jz sta per "jump if zero" e provoca un salto all'indirizzo loc_401028 se il risultato del confronto precedente era zero

```
    cmp      [ebp+var_4], 0  
    jz       short loc_401028
```

Chiamata di funzione se la connessione è attiva:

I parametri sono passati sullo stack tramite funzione di push

call sub_A0117F: Chiama una funzione interna utilizzata per gestire la visualizzazione di messaggi, in questo caso un Messaggio di Successo.

```
    push    offset aSuccessInterne ; "Success: Internet Connection\n"  
    call    sub_40117F
```

Chiamata di funzione se la connessione non è attiva:

L'parametri sono passati sullo stack tramite funzione di push

call sub_A0117F: Chiama una funzione interna utilizzata per gestire la visualizzazione di messaggi, in questo caso un Messaggio di Errore

```
push    offset aError1_1NoInte  
call    sub_40117F
```

Operazioni di Pulizia:

Le operazioni di pulizia rimuovono dati temporanei dallo stack e ripristinano il contesto di esecuzione.

mov esp, ebp: Ripristina lo stack pointer al valore del base pointer.

pop ebp: Ripristina il base pointer originale, rimuovendo il valore dallo stack

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

COMPORTAMENTO IPOTETICO

L'analisi del codice assembly del malware rivela l'uso di costrutti noti come **if/else**, **chiamate di funzione**, e **gestione dello stack**. Questi costrutti sono utilizzati per verificare la connessione Internet e visualizzare un messaggio di successo o errore. L'organizzazione e il flusso del codice mostrano una tipica struttura di programma che esegue operazioni condizionali basate sull'esito di una funzione di sistema. Questi costrutti indicano che il codice è progettato per interagire con l'ambiente di sistema in modo condizionato, eseguendo operazioni specifiche in base allo stato della connessione Internet.





TABELLA SPIEGAZIONE DELLE SINGOLE RIGHE DI CODICE

1

push ebp	Salva il valore corrente del puntatore di base (EBP) sullo stack.
mov ebp, esp	Imposta il base pointer (ebp) al valore corrente dello stack pointer (esp)
push ecx	Salva il valore del registro ecx nello stack.
push 0	Inserisce il valore 0 nello stack come parametro dwReserved per la chiamata di funzione.
push 0	Inserisce il valore 0 nello stack per il parametro lpdwFlags, un puntatore a una descrizione di connessione.
call ds:InternetGetConnectedState	Chiama la funzione InternetGetConnectedState per verificare lo stato della connessione a Internet.
mov [ebp+var_4], eax	Salva il valore restituito dalla funzione in una variabile locale var_4.
cmp [ebp+var_4], 0	Confronta il valore di var_4 con 0.
jz short loc_401028	Se il risultato del confronto è zero (nessuna connessione), salta all'etichetta loc_401028.
loc_40101D	Blocco di codice eseguito se c'è connessione Internet (var_4 != 0):
push offset aSuccessInterne	Inserisce l'indirizzo del messaggio "Success: Internet Connection \n" nello stack.

2

call sub_40117F	Chiama una funzione (per stampare il messaggio).
add esp, 4	Ripristina lo stack pointer incrementandolo di 4 byte (rimuove il parametro dal stack).
mov eax, 1	Imposta il registro eax a 1, indicando successo.
jmp short loc_40103A	Salta alla fine della funzione.
loc_401028:	Blocco di codice eseguito se non c'è connessione Internet (var_4 == 0):
push offset aError1_1NoInte	Inserisce l'indirizzo del messaggio "Error 1.1: No Internet\n" nello stack.
call sub_40117F	Chiama una funzione (per stampare il messaggio).
add esp, 4	Ripristina lo stack pointer incrementandolo di 4 byte (rimuove il parametro dal stack).
xor eax, eax	Imposta il registro eax a 0.
loc_40103A:	Epilogo della funzione:
mov esp, ebp	Ripristina lo stack pointer al valore originale salvato nel base pointer.
pop ebp	Ripristina il valore del base pointer originale.

3

retn

Ritorna dalla funzione, pulendo lo stack.



BONUS

INTRODUZIONE

Il file iexplore.exe, situato nella directory C:\Programmi\Internet Explorer, è stato segnalato come sospetto da un nuovo dipendente del reparto tecnico. Come membri del SOC, è nostra responsabilità verificare la legittimità del file e assicurare che non costituisca una minaccia per il sistema. In questo report, verranno utilizzati strumenti di analisi statica e informazioni pubbliche per dimostrare che iexplore.exe non è un file maligno.

ANALISI STATICÀ CON CFF EXPLORER

Librerie Importate:

- KERNEL32.dll, USER32.dll, SHELL32.dll, ADVAPI32.dll, OLE32.dll

Queste sono librerie standard del sistema operativo Windows utilizzate per funzioni di base e interazioni di sistema.

Conclusioni sulle Importazioni:

L'utilizzo esclusivo di librerie standard e conosciute supporta la legittimità del file. Non sono presenti importazioni di librerie non comuni o sospette che potrebbero suggerire attività malevole.

ANALISI STATICÀ CON CFF EXPLORER

Sezioni del File

- **.text, .rdata, .data, .rsrc, .reloc**

Assenza di Offuscamento

Non vi sono segni evidenti di offuscamento nel file. Gli indirizzi, le dimensioni, e le informazioni delle sezioni sono coerenti e leggibili, indicando che non vi è tentativo di nascondere informazioni.

ANALISI TRAMITE VIRUSTOTAL

Rilevamenti Antivirus

Risultato: 0 su 74 motori antivirus hanno rilevato il file come maligno.

: L'assenza di rilevamenti da parte di numerosi motori antivirus è un forte indicatore che il file non è considerato pericoloso o malevolo da nessun fornitore di sicurezza.

Firma Digitale

Il file è firmato da Microsoft Corporation.

La firma è valida, confermando che il file è stato distribuito ufficialmente da Microsoft e non è stato modificato

ANALISI TRAMITE VIRUSTOTAL

Comunicazioni di Rete

Solo domini Microsoft, come www.microsoft.com, sono stati contattati. Le comunicazioni di rete sono limitate a server legittimi di Microsoft, coerenti con le operazioni di un browser web legittimo.

CONCLUSIONE

Dall'analisi condotta utilizzando **CFF Explorer** e **VirusTotal**, possiamo affermare con sicurezza che il file **iexplore.exe** è legittimo e non rappresenta una minaccia per il sistema. Gli elementi chiave che supportano questa conclusione includono:

- **Librerie e Sezioni Standard:** Le importazioni e le sezioni del file sono tipiche di un'applicazione di sistema autentica e non ci sono tentativi di offuscamento
- **Nessun Rilevamento Antivirus:** Nessun fornitore di sicurezza ha identificato il file come malevolo.
- **Firma Digitale Valida:** La firma digitale di Microsoft è intatta e conferma l'integrità del file.
- **Comportamento Normale di Rete:** Le comunicazioni di rete sono legittime e limitate a domini affidabili.

Pertanto, possiamo tranquillizzare il dipendente che *iexplore.exe*, essendo parte del software di Internet Explorer, **non è maligno**. Rimane un componente legittimo e sicuro del sistema operativo Windows.



GRAZIE

ADAM DERRO