PRIMA PARTE

Con il seguente report andremo ad analizzare l'eseguibile richiesto dalla traccia e per prima cosa per capire che tipo di librerie utilizza ci serviamo di un potente strumento a nostra disposizione direttamente nella macchina che ci è stata fornita.

Andremo quindi ad utilizzare CFF Exploer che è uno strumento software utilizzato per l'analisi e la modifica dei file eseguibili Windows ed inoltre offre funzionalità per esaminare la struttura dei file, come le intestazioni PE, le sezioni e le risorse, nonché per apportare modifiche come l'aggiunta o la sostituzione di risorse. Noi lo andremo ad utilizzare per scopi di reverse engineering e debugging tramite analisi statica e/o dinamica basica.

Aprendo la sessione nella sezione d'utilizzo il software ci dà come output la seguente schermata

Malware_U3_W2_L5.exe

Module Name	Imports	OFTs	TimeDateStamp	ForwarderChain	Name RVA	FTs (IAT)
szAnsi	(nFunctions)	Dword	Dword	Dword	Dword	Dword
KERNEL32.dll	44	00006518	00000000	00000000	000065EC	00006000
WININET.dll	5	000065CC	00000000	00000000	00006664	000060B4

Da questa schermata ricaviamo che le librerie utilizzate dal malware sono due e sono ***KERNEL32.dll*** e ***WININET.dll***

La prima ***Kernel32.dll*** è una libreria di sistema di Windows che fornisce funzioni fondamentali per il funzionamento di molti programmi ed è un componente essenziale per il corretto funzionamento del sistema operativo Windows.

La seconda ***WININET.dll*** è una libreria di sistema di Windows che è responsabile della gestione delle operazioni di rete e del supporto per l'interazione con il web all'interno dei programmi su Windows.

Per quanto riguarda invece il secondo quesito relativo alle sezioni utilizzate dal malware analizzato scopriamo che quelle che lo compongono sono le seguenti che vediamo in figura sempre utilizzando CFF Explorer

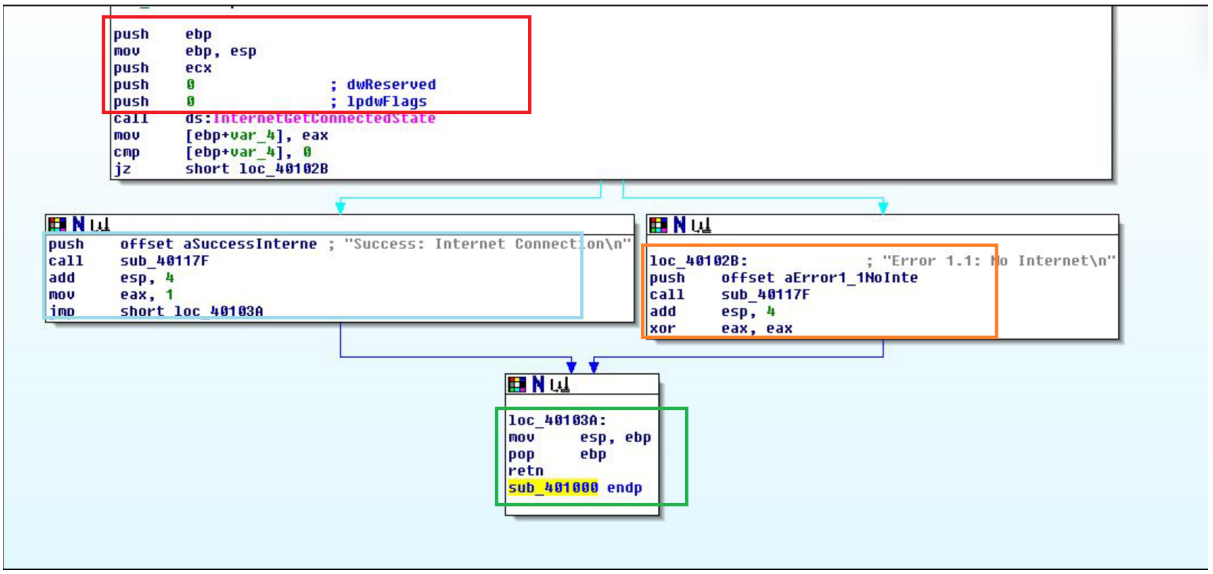
Malware_U3_W2_L5.exe

Name	Virtual Size	Virtual Address	Raw Size	Raw Address	Reloc Address	Linenumbers	Relocations N...	Linenumbers ...	Characteristics
Byte[8]	Dword	Dword	Dword	Dword	Dword	Dword	Word	Word	Dword
.text	00004A78	00001000	00005000	00001000	00000000	00000000	0000	0000	60000020
.rdata	0000095E	00006000	00001000	00006000	00000000	00000000	0000	0000	40000040
.data	00003F08	00007000	00003000	00007000	00000000	00000000	0000	0000	C0000040

Possiamo notare tra queste ***.text, .rdata e .data*** e andiamo insieme ad analizzarle qui di seguito.

- ***.text*** è la sezione di un file eseguibile o di una libreria che contiene il codice eseguibile del programma, ovvero le istruzioni che la CPU deve eseguire durante l'esecuzione del programmi
- ***.rdata*** è la sezione di un file eseguibile o di una libreria contenente dati costanti o di sola lettura utilizzati dal programma durante l'esecuzione. Questi dati non possono essere modificati e sono protetti da modifiche accidentali
- ***.data*** è la sezione che contiene dati inizializzati utilizzati durante l'esecuzione di un programma. Questi dati possono essere modificati durante l'esecuzione del programma, consentendo di memorizzare valori variabili o informazioni dinamiche. La sezione .data serve quindi a conservare dati modificabili durante l'esecuzione del programma, come variabili, array, strutture dati e altre informazioni che possono essere alterate durante l'esecuzione del programma.

SECONDA PARTE



Nella parte in ***rosso*** avviene innanzitutto la creazione dello stack ed in seguito avvia una chiamata a una funzione esterna utilizzando l'istruzione call, seguita da un confronto (cmp) e un salto condizionale (jz) basato sul risultato del confronto. In breve la funzione InternetGetConnectedState viene chiamata e il suo valore di ritorno viene confrontato con zero, con un salto verso una determinata posizione di memoria se il risultato del confronto è uguale a zero.

Nella parte in ***celeste*** il costrutto include una chiamata a una funzione esterna utilizzando l'istruzione call, seguita da operazioni per gestire i parametri nello stack e il flusso di esecuzione attraverso un salto incondizionato (jmp) e un salto condizionale (jz). Inoltre, viene utilizzata un'istruzione di movimento (mov) per assegnare un valore al registro EAX. In linea generale il codice sembra coinvolgere la gestione delle chiamate a funzioni, l'uso di parametri e la gestione del flusso di esecuzione.

Nella parte in ***arancione*** il costrutto gestisce un'operazione di gestione dell'errore. Se il flusso di esecuzione raggiunge l'etichetta "loc_401028", viene eseguita una serie di operazioni per visualizzare un messaggio di errore specifico e impostare un valore di ritorno appropriato. Le istruzioni coinvolte includono l'utilizzo di push per mettere l'offset di una stringa nello stack, call per chiamare una funzione correlata all'elaborazione dell'errore, add per ripulire la pila, e xor per impostare un registro a zero.

Nella parte in ***verde*** il costrutto presente in questo frammento di codice assembly rappresenta la conclusione di una funzione. Le istruzioni mov, pop e retn vengono utilizzate per ripristinare lo stack e restituire il controllo al punto di chiamata della funzione chiamante. In generale, questo costrutto indica l'uscita ordinata dalla funzione e il ripristino dello stato precedente all'invocazione della funzione.

Malware Analysis

L'ipotesi sul comportamento della funzionalità implementata nel codice assembly dipende dalle specifiche delle funzioni chiamate e dal contesto in cui viene utilizzato il codice ed in questo caso non ne siamo a conoscenza ma si possono fare alcune supposizioni generiche basate sulle operazioni e sulle istruzioni presenti nel codice.

Inizialmente il codice sembra coinvolgere il controllo della connessione a Internet, utilizzando la funzione InternetGetConnectedState, il codice determina se il sistema ha una connessione Internet attiva. Se il risultato del controllo è zero, il codice passa alla gestione dell'errore.
Nel caso in cui il controllo di connessione a Internet restituisca un valore diverso da zero, il codice passa alla gestione del successo. Viene mostrato un messaggio di successo "Success: Internet Connection\n".
Nel caso in cui il controllo di connessione a Internet restituisca zero, il codice passa alla gestione dell'errore. Viene mostrato un messaggio di errore "Error 1.1: No Internet\n" e potrebbe essere chiamata la funzione sub_40117F per elaborare ulteriormente l'errore.
Alla fine, indipendentemente dal successo o dall'errore, il codice esegue un'operazione di ripristino dello stack e restituisce il controllo al punto di chiamata della funzione chiamante.

In sintesi, il malware analizzato sembra controllare lo stato della connessione a Internet e fornire feedback all'utente tramite messaggi di successo o di errore.

TERZA PARTE

Per risolvere l'ultimo quesito bonus possiamo avvalerci di vari strumenti e varie modalità che andrò sotto a spiegare.

Per quanto riguarda ***l'analisi statica***:

- la prima cosa che facciamo è la verifica della firma digitale dell'exe a cui riasaliamo cliccando con il tasto destro del mouse sul file "IEXPLORE.EXE" e selezionando "Proprietà". Se la firma digitale risulta valida e associata a Microsoft Corporation, ciò suggerisce che il file proviene da una fonte attendibile.
- Dopodiché confrontiamo l'hash del file, calcolando l'hash del file "IEXPLORE.EXE" attraverso uno strumento come MD5deep o un sito web specializzato nell'hashing dei file con l'hash noto del file legittimo di Internet Explorer. Per eseguire questo passaggio possiamo utilizzare VIRUSTOTAL.
- Controlliamo che il file "IEXPLORE.EXE" si trovi nella directory corretta, ovvero "C:\Program Files\Internet Explorer".
- Verifichiamo inoltre chela data di creazione e l'ultima modifica del file siano coerenti con l'installazione di Internet Explorer o con gli aggiornamenti recenti

Per quanto riguarda ***l'analisi dinamica***:

- Utilizziamo un ambiente di sandboxing sicuro per eseguire il file "IEXPLORE.EXE" isolato dal sistema operativo principale. Potremo utilizzare strumenti come Any.run o Hybrid Analysis per questo scopo. Durante l'esecuzione, osserveremo l'attività del file all'interno del sandbox.
- Possiamo invece monitorare le attività con Process Monitor che registra e monitora le attività di un processo o del file sul sistema operativo Windows. Tracciando le operazioni di accesso al registro di sistema e al file system, fornisce informazioni sui processi e i thread in esecuzione e offre funzionalità di filtro e ricerca. Utilizzando Process Monitor, è possibile identificare le azioni eseguite dal file "IEXPLORE.EXE" e determinare se presenta comportamenti sospetti o coerenti con un componente legittimo del sistema operativo.
- Wireshark, invece, ci consentirà di catturare e analizzare il traffico di rete generato dal file "IEXPLORE.EXE". Puoi osservare le connessioni di rete che stabilisce e verifica se vi sono comunicazioni sospette o indirizzi IP non riconosciuti.

Riassumendo, queste sono le fasi dell'analisi che andremo ad eseguire per valutare la natura del file "IEXPLORE.EXE" che non è affatto un malware ma l'eseguibile di Windows Explorer, alla fine dell'analisi che faremo effettuare poi direttamente al neo assunto per renderlo consapevole e soprattutto partecipe di questa analisi con il quale inizierà a familiarizzare con i processi relativi alla malware analysis potremo avere la certezza dell'autenticità di quanto ipotizzavamo.