CS 0423 - Alex Fiorillo

CONSEGNA S11/L5

Analisi avanzate: un approccio pratico

TRACCIA

Con riferimento al codice presente nelle immagini successive, **rispondere ai seguenti quesiti:**

1) **Spiegare**, motivando, quale **salto condizionale** effettua il malware.
2) **Disegnare un diagramma di flusso** identificando i salti condizionali (sia quelli effettuati che quelli non effettuati). Indicare con una linea verde i salti effettuati e con una rossa i salti non effettuati.
3) Quali sono le diverse **funzionalità implementate** all'interno del malware?
4) Con riferimento alle istruzioni «call» presenti in *tabella 2* e *3*, **dettagliare come sono passati gli argomenti alle successive chiamate di funzione**.

Tabella 1

Locazione	Istruzione	Operandi	Note
00401040	mov	EAX, 5	
00401044	mov	EBX, 10	
00401048	cmp	EAX, 5	
0040105B	jnz	loc 0040BBA0	; tabella 2
0040105F	inc	EBX	
00401064	cmp	EBX, 11	
00401068	jz	loc 0040FFA0	; tabella 3

Tabella 2

Locazione	Istruzione	Operandi	Note
0040BBA0	mov	EAX, EDI	EDI= www.malwaredownload.com
0040BBA4	push	EAX	; URL
0040BBA8	call	DownloadToFile()	; pseudo funzione

Tabella 3

Locazione	Istruzione	Operandi	Note
0040FFA0	mov	EDX, EDI	EDI: C:\Program and Settings\Local User\Desktop\Ransomware.exe
0040FFA4	push	EDX	; .exe da eseguire
0040FFA8	call	WinExec()	; pseudo funzione

Cos’è un Malware?

il termine malware nasce dalla combinazione delle parole “**malicious”** (malevolo) e **“software”** e si riferisce a qualsiasi tipo di software progettato per danneggiare, compromettere o alterare il funzionamento di un sistema informatico, di un dispositivo o di una rete, senza il consenso o la conoscenza da parte dell’utente. Un malware può assumere diverse forme e può essere progettato per svolgere una vasta gamma di attività dannose.

I tipi più comuni di malware sono:

- **Virus**
  Si diffonde passando da computer a computer, senza azione diretta o autorizzazione da parte dei sistemi infetti. Si copiano in sezioni particolari all’interno del file system e i più sofisticati cercano di nascondersi dalle analisi dei vari sistemi di sicurezza (come antivirus o anti malware.
- **Trojan**
  Un tipo di malware che si nasconde all’interno di un file apparentemente innocuo, come un documento office oppure un PDS. Si attiva quando la vittima apre il file. Tra i troja più comunemente utilizzati troviamo le backdoor, che sono generalmente utilizzate per fornire agli attaccanti delle shell sui sistemi infetti.
- **Rootkit**
  Un malware progettato per nascondersi dagli utenti e dagli antivirus per prendere il controllo completo del sistema operativo. Un rootkit permette di mantenere privilegi elevati su una macchina senza essere notati.
- **Bootkit**
  Sono dei rootkit che aggirano le protezioni del sistema operativo in quanto entrano

in funzione prima dell'avvio completo del sistema operativo, in particolar modo prima dell'attivazione dei moduli di sicurezza di un sistema operativo.

- **Adware**
  Sono dei programmi fastidiosi che mostrano pubblicità agli utenti di un pc.
- **Spyware**
  Programmi che si usano per raccogliere informazioni sulle attività degli utenti di un sistema, ad esempio: il tipo di sistema operativo installato sulla macchina, i siti visitati, le password. Queste informazioni vengono inviate successivamente ad un server sotto il controllo dell'attaccante.
- **Dialer**
  Un programma che cerca di chiamare numeri telefonici a pagamento per guadagnare soldi
- **Keylogger**
  Programma che registra ogni tasto premuto sulla macchina della vittima. I keylogger registrano: i tasti premuti sulla tastiera, il nome delle finestre aperte dall'utente. Salvano poi queste informazioni in un file di log che spediscono ad un server controllato dall'attaccante.

Malware analysis

L'analisi del malware o «malware analysis» è l'insieme di competenze e tecniche che permettono ad un analista della sicurezza informatica di indagare accuratamente un malware per studiare e capire esattamente il suo comportamento al fine di rimuoverlo dal sistema.
Durante lo studio dell'analisi dei malware incontreremo due tecniche principali di analisi:

- **Analisi statica**
- **Analisi dinamica**

Mentre **l'analisi dinamica** prevede **l'esecuzione del malware** in ambiente controllato, **l'analisi statica fornisce tecniche e strumenti per analizzare** il comportamento di un software malevolo **senza la necessità di eseguirlo.**

Le due tecniche sono tra di loro complementari, per un'analisi efficace i risultati delle analisi statiche devono essere poi confermate dai risultati delle analisi dinamiche.

Assembly

L'analisi statica avanzata presuppone la conoscenza di un particolare tipo di linguaggio, chiamato linguaggio Assembly. Il linguaggio Assembly è univoco per una data architettura di un PC, ma cambia da architettura ad architettura.

L'analista di sicurezza durante l'analisi statica utilizzerà dei tool chiamati «Disassembler» che sono programmati per tradurre le istruzioni binarie eseguite dalla CPU in istruzioni più leggibili dall'uomo, che è proprio il linguaggio Assembly.

La conoscenza del linguaggio Assembly servirà per "leggere" le istruzioni eseguite dalla CPU in formato leggibile dall'uomo.

Come detto in precedenza Assembly è un linguaggio che dipende dall'architettura del calcolatore, tra le più note troviamo x86, 64, ARM, MIPS e PowerPC.
Quello che ci interessa sapere è che i processori possono essere a 32 o 64 bit, cioè l'ammontare massimo di informazioni che la CPU riesce a gestire per ogni singola operazione.

Per lo scopo del nostro progetto vedremo l'Assembly per i set di istruzioni x86, ovvero per i processori a 32 bit.

La base del linguaggio Assembly sono le istruzioni che sono costituite da due parti:

- **un codice mnemonico,** ovvero una parola che identifica l'istruzione da eseguire
- **uno o più operandi,** che identificano le variabili o la memoria oggetto dell'istruzione.

Si possono utilizzare tre tipi diversi di operandi:

- **un valore,** come ad esempio un numero. Attenzione, generalmente i valori immediati non sono scritti in formato decimale ma bensì in formato esadecimale nella forma **0xYY** dove **YY** rappresenta la versione esadecimale del numero decimale
- **uno dei registri** messi a disposizione della CPU
- **un indirizzo di memoria** che contiene un valore di interesse

Un **registro** è un tipo di memoria a rapido accesso (contenute nelle dimensioni ma che hanno una velocità di accesso maggiore rispetto ad altre memorie, come quelle secondarie) che consente di salvare temporaneamente una variabile che deve essere utilizzata dalla CPU.

1) Spiegare, motivando, quale salto condizionale effettua il malware

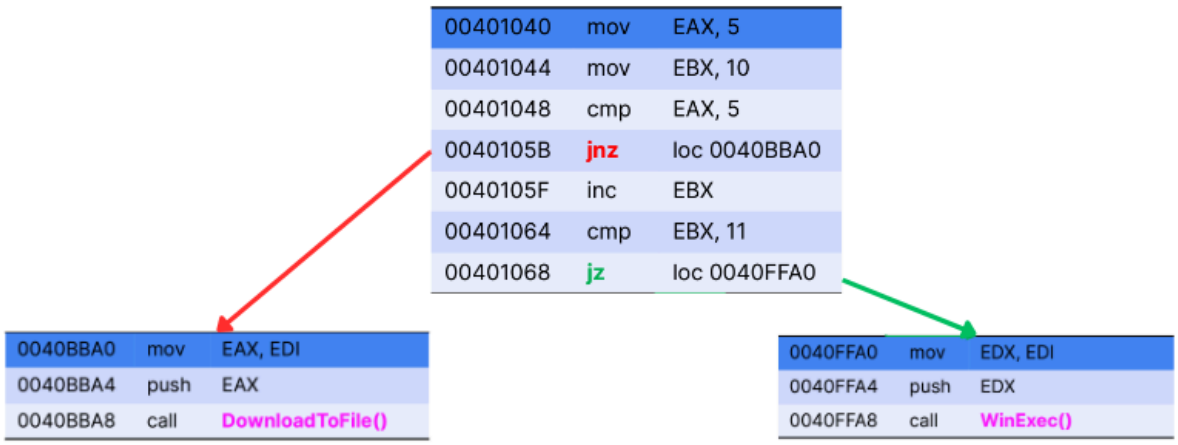
Con riferimento alla *tabella 1* (pag 2) notiamo che il malware effettua il salto condizionale alla locazione di memoria 00401068.

L'istruzione "jz" effettua un salto al punto designato se i valori degli operandi nella precedente istruzione "cmp" sono identici, come in questa situazione, dove EBX assume il valore di 11.

2) Visualizzazione grafica salti condizionali

La linea verde identifica il salto condizionale eseguito (ossia la chiamata alla funzione "WinExec()"), mentre il salto condizionale non eseguito è stato evidenziato in rosso (ossia la mancata chiamata alla funzione "DownloadToFile()".

Diagramma:



3) Quali sono le diverse funzionalità implementate all'interno del malware?

Le funzionalità implementate all'interno del malware sono due ma ne esegue soltanto una.

- Scarica un malware da internet, comportandosi in questo caso come un downloader
- Esegue un malware già presente sulla macchina locale utilizzando la funzione "WinExec()", che è proprio la funzionalità che andrebbe ad eseguire.

4) Con riferimento alle istruzioni «call» presenti in tabella 2 e 3, dettagliare come sono passati gli argomenti alle successive chiamate di funzione.

In entrambi i casi, sia per la funzione "DownloadToFile()" che per la funzione "WinExec()", i parametri vengono passati sullo stack utilizzando l'istruzione push. In particolare:

- Per la funzione "**DownloadToFile()**" viene passato l'URL (www.malwaredownload.com) da cui scaricare ulteriori file compromessi
- Per la funzione "**WinExec()**" viene passato il percorso assoluto dell'eseguibile da avviare