S11/L4

Analisi comportamentale delle categorie dei malware più note

Traccia

La figura nella slide successiva mostra un estratto del codice di un malware.

Identificate:

1. Il tipo di Malware in base alle chiamate di funzione utilizzate.

2. Evidenziate le chiamate di funzione principali aggiungendo una descrizione per ognuna di essa.

3. Il metodo utilizzato dal Malware per ottenere la persistenza sul sistema operativo.

4. BONUS: Effettuare anche un’analisi basso livello delle singole istruzioni

Traccia

.text: 00401010	push eax	
.text: 00401014	push ebx	
.text: 00401018	push ecx	
.text: 0040101C	push WH_Mouse	; hook to Mouse
.text: 0040101F	call SetWindowsHook()	
.text: 00401040	XOR ECX,ECX	
.text: 00401044	mov ecx, [EDI]	EDI = «path to startup_folder_system»
.text: 00401048	mov edx, [ESI]	ESI = path_to_Malware
.text: 0040104C	push ecx	; destination folder
.text: 0040104F	push edx	; file to be copied
.text: 00401054	call CopyFile();	

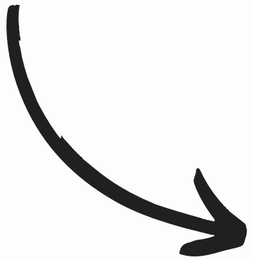
Analisi comportamentale

Nella lezione di oggi abbiamo studiato l'**analisi comportamentale** del malware, che ci aiuta ad identificare **funzionalità specifiche comuni** che hanno determinate tipologie di Malware, almeno quelle più note.

Studiare i **comportamenti** e le caratteristiche comuni del malware ci aiuta a identificare meglio la **tipologia**, permettendoci di ottimizzare la ricerca e l'analisi in modo più **mirato** ed efficiente.

1 - Tipo di malware

Dallo snippet di codice fornito nella traccia, senza il codice e l'analisi completi non si può catalogare con certezza assoluta. Potrebbe essere parte di un **keylogger (date le funzioni chiamate)**, di un **trojan di sorveglianza** o di uno **spyware**. Posso ipotizzare questo basandomi sulle chiamate di funzione che stabiliscono **persistenza** e **monitoraggio** di determinati eventi del sistema.



Le chiamate a **SetWindowsHook** e **CopyFile** indicano un comportamento tipico delle categorie di cui sopra, con il fine di **intercettare** eventi di sistema, in questo caso eventi relativi al mouse dato l'**hook impostato**.

2 - Chiamate di funzione principali

CallSetWindowsHook()

Utilizzata per **installare un hook**, in questo caso specificamente per **intercettare eventi del mouse (WH_MOUSE)**. Questo permette al malware di **monitorare** le azioni dell'utente legate al mouse, potenzialmente per scopi di sorveglianza o per attivare altre funzionalità malevole.

```
.text: 0040101F                call SetWindowsHook()
```

CopyFile()

Usata per **copiare** il file del malware nella cartella di avvio del sistema (**startup_folder_system**). Questo garantisce che il malware venga eseguito automaticamente ogni volta che il sistema viene avviato, assicurandovi persistenza.

```
.text: 00401054                call CopyFile();
```

3 - Metodo per ottenere persistenza

Il malware, come detto poco nella descrizione della funzione, ottiene la persistenza **copiando sé stesso nella cartella di avvio del sistema**. Questo viene realizzato tramite la funzione **CopyFile**(), che duplica il file del malware in una directory destinata all'esecuzione automatica all'avvio del sistema operativo.

Per ottenere persistenza, potrebbe copiare se stesso in vari **percorsi strategici** del sistema operativo. Tra i più comuni ci sono le cartelle di avvio (come **Startup**), sia quella **globale** che specifica per l'utente.

4 - Analisi delle singole istruzioni

L'analisi delle istruzioni assembly rivela come il malware **manipola direttamente** i **registri** e sfrutta le **API** di Windows per **controllare il sistema**.

Istruzione	Analisi
push eax, push ebx, push ecx	Spinge i valori dei registri nello stack per preservarli
push WH_Mouse	Imposta il tipo di hook su eventi del mouse
call SetWindowsHook()	Installa un hook per intercettare eventi del mouse
XOR ECX, ECX	Azzera il registro ECX per prepararlo a operazioni future
mov ecx, [EDI]	Copia il valore puntato da EDI nel registro ECX (percorso destinazione).

Istruzione	Analisi
mov edx, [ESI]	Copia il valore puntato da ESI nel registro EDX (percorso sorgente).
push ecx	Salva il valore di ECX nello stack (prepara parametro per CopyFileA).
push edx	Salva il valore di EDX nello stack (prepara parametro per CopyFileA).
call CopyFile()	Copia un file dalla sorgente alla destinazione per persistenza.

Grazie

Flavio Scognamiglio