

# S11

TEAM CYBERSECURITY



# Traccia Giorno 4

La figura 1 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

Figura 1:

.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();	

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

Figura 1:

```
.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»
                    ESI= path_to_Malware
                    ; destination folder
                    ; file to be copied
.text: 00401048      mov edx, [ESI]
.text: 0040104C      push ecx
.text: 0040104F      push edx
                    call CopyFile()
```



Il codice presente nella tabella sotto ci fa pensare ad un Malware di tipo Keylogger, infatti vediamo l'utilizzo della funzione «`SetWindowsHook`», per l'installazione di un «`hook`» per controllare un device.

Quello che notiamo, tuttavia è che a differenza del codice della lezione teorica, l'ultimo parametro passato sullo stack è «`WH_MOUSE`». Questo ci fa pensare che il Malware non registra la digitazione dei tasti della tastiera dell'utente, ma bensì la digitazione dei tasti del mouse!

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

Possiamo notare in evidenza le seguenti chiamate di funzione: SetWindowsHook e CopyFile

### SetWindowsHook

Installa una procedura di hook definita dall'applicazione in una catena di hook.

Installa una procedura di hook per monitorare il sistema per determinati tipi di eventi. Questi eventi sono associati a un thread specifico o a tutti i thread nello stesso desktop del thread chiamante.

### CopyFile

Quando CopyFile viene usato per copiare un file crittografato, tenta di crittografare il file di destinazione con le chiavi usate nella crittografia del file di origine. Se non è possibile eseguire questa operazione, questa funzione tenta di crittografare il file di destinazione con chiavi predefinite. Se non è possibile eseguire nessuno di questi metodi, CopyFile non riesce con un codice di errore ERROR\_ENCRYPTION\_FAILED .



Figura 1:

.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();	

## SetWindowsHook

Queste istruzioni salvano i registri eax, ebx, e ecx nello stack per preservare il loro stato attuale.

Spinta del Tipo di Hook:

### **push WH\_Mouse**

Questa istruzione spinge il valore WH\_MOUSE nello stack. WH\_MOUSE è una costante che specifica che l'hook è per gli eventi del mouse.

Chiamata a SetWindowsHook():

```
call SetWindowsHook()
```

Questa istruzione chiama la funzione SetWindowsHook() utilizzando i parametri precedentemente spinti nello stack. In questo caso, SetWindowsHook(WH\_MOUSE, lpfn) (dove lpfn è l'indirizzo della funzione di hook) imposta un hook per intercettare gli eventi del mouse.

### **Scopo dell'Hook**

L'obiettivo di impostare questo hook è monitorare gli eventi del mouse. Tuttavia, nel contesto di questo codice assembly, l'intenzione è malevola. Intercettare gli eventi del mouse può essere utilizzato per:

Spiare l'utente: Registrare i clic del mouse per ottenere informazioni sensibili.

Manipolare il comportamento del sistema: Modificare il comportamento delle applicazioni basato sugli eventi del mouse.

Persistenza: Combinato con altre azioni, come copiare il malware nella cartella di avvio, assicurarsi che il malware venga eseguito e possa continuare a monitorare l'utente.

### **Conclusione**

In sintesi, SetWindowsHook() viene utilizzato per intercettare gli eventi del mouse impostando un hook specifico. Nel contesto del codice mostrato, questa funzione viene utilizzata come parte di un processo malevolo per monitorare e potenzialmente manipolare gli eventi del mouse, probabilmente come parte di un malware che cerca di ottenere persistenza e raccogliere dati sensibili dall'utente.



## SetWindowsHook

In questo contesto specifico, la funzione SetWindowsHook() è utilizzata per impostare un hook che intercetta gli eventi del mouse. Un hook è una funzione definita dall'utente che può monitorare e, in alcuni casi, modificare o prevenire l'esecuzione di determinati eventi del sistema operativo.

Sintassi Generale

**HHOOK SetWindowsHook(int idHook, HOOKPROC lpfn);**

Parametri

idHook: Identificatore del tipo di hook da installare. Nel caso mostrato nell'immagine, viene utilizzato WH\_MOUSE, che indica un hook per intercettare gli eventi del mouse.

lpfn: Puntatore alla funzione di hook, che verrà chiamata ogni volta che si verifica l'evento specificato.

Funzionamento nel Codice Mostrato

Nel frammento di codice assembly mostrato, l'hook viene impostato come segue:

Preparazione dello Stack:

```
push eax  
push ebx  
push ecx
```



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

La funzione CopyFile riceve come parametri il registro ecx, che contiene una cartella di avvio del sistema, e edx che contiene il file da copiare, cioè il path del Malware.

Lo scopo di questa funzione è quindi quello di copiare il Malware in una cartella di avvio del sistema in modo da ottenere la **persistenza** e cioè l'avvio del Malware ad ogni accensione del dispositivo senza interventi antropici.

Possiamo immaginare che nel codice di implementazione della funzione ci sia un loop per leggere i dati dal file sorgente e copiarli nel file destinazione.



## BONUS: Effettuare anche un'analisi basso livello delle singole istruzioni

Istruzione	Descrizione
00401010 push eax	Inserisce il valore del registro EAX nello stack.
00401014 push ebx	Inserisce il valore del registro EBX nello stack.
00401018 push ecx	Inserisce il valore del registro ECX nello stack.
0040101C push WH_Mouse	Questo viene usato per impostare un hook per monitorare gli eventi del mouse.
0040101F call SetWindowsHook	Chiama la funzione SetWindowsHook per installare una procedura di hook che monitorizza gli eventi del mouse.
00401040 XOR ECX, ECX	Azzera il registro ECX eseguendo un'operazione di XOR bit a bit di ECX con sé stesso (ECX = 0).
00401044 mov ecx, [EDI]	Sposta il valore all'indirizzo di memoria puntato da EDI nel registro ECX. Qui, EDI contiene il percorso della cartella di avvio del sistema.
00401048 mov edx, [ESI]	Sposta il valore all'indirizzo di memoria puntato da ESI nel registro EDX. Qui, ESI contiene il percorso del file malware.
0040104C push ecx	Inserisce il valore del registro ECX (cartella di destinazione) nello stack.
00401054 call CopyFile	Chiama la funzione CopyFile per copiare il file malware dal percorso puntato da ESI alla cartella di avvio puntata da EDI.

## BONUS: Effettuare anche un'analisi basso livello delle singole istruzioni

### Analisi del Codice

Imposta un hook per il mouse: Le prime istruzioni salvano i registri eax, ebx, e ecx nello stack, quindi chiamano SetWindowsHook() con il parametro WH\_Mouse. Questo impone un hook per intercettare gli eventi del mouse, permettendo al malware di eseguire codice ogni volta che il mouse viene utilizzato.

Copia un file nella cartella di avvio del sistema: Successivamente, il codice muove i percorsi della cartella di avvio e del malware nei registri ecx e edx, e poi li spinge nello stack prima di chiamare CopyFile(). Questo copia il file malware nella cartella di avvio del sistema, assicurando che venga eseguito ad ogni avvio del sistema.

Queste operazioni sono tipiche di un malware che tenta di persistere nel sistema intercettando input dell'utente e assicurandosi di essere eseguito all'avvio del sistema.



**Thank  
you!**

# *Our* **TEAM**



**Mario Marsicano**



**ZhongShi Liu**

## **TEAM LEADER**



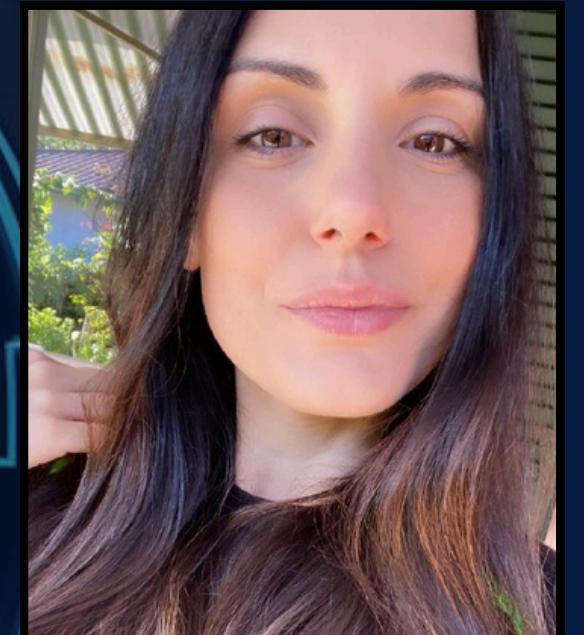
**Mara Dello Russo**



**Andrè Vinicius**



**Anapaula Palacin**



**Roberta Mercadante**