MALWARE ANALYSIS: FUNZIONALITA MALWARE

S11 - L4

.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();	

Oggi analizzeremo un piccolo pezzo di codice Assembly, riga per riga. Le prime 3 righe mettono semplicemente i registri **EAX**, **EBX** ed **ECX** nello stack, preparandoli per un uso successivo.

.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();	

Ora viene eseguita la funzione principale del malware, **WH_Mouse**, che è una procedura specifica della funzione **SetWindowsHook**, richiamata dal codice con l'istruzione call. Questa funzione consente di catturare e registrare le interazioni del mouse, ovvero è una sorta di keylogger che probabilmente in seguito salverà queste informazioni in qualche file o registro.

```
.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();
```

Potremmo dire che questa è una sezione di codice completamente separata dalla precedente. Per prima cosa cancella il registro **ECX** con l'istruzione **XOR** (*XORring* un registro a se stesso cancella tutti i bit al suo interno). Dopo asegna il percorso della cartella di avvio di Windows **[EDI]** a **ECX**, e poi il percorso in cui si trova il malware **[ESI]** al registro **EDX**.

.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();	

Infine, mette i 2 registri appena citati nello stack, preparandoli per essere utilizzati come argomenti nella successiva e ultima funzione: CopyFile, dove il file Malware viene copiato nella directory di avvio di Windows, ottenendo così la persistenza ad ogni avvio del sistema.