

Malware Analysis

Build_Week_Unit_3

Analisi statica

- 1) Alla funzione Main() sono passati 3 **parametri** : argv, argc, envp.
- 2) Alla funzione Main() sono passate 4 **variabili**: hModule, Data, var_8, var_4.

```
.text:004011D0 ; .text:004011D0
.text:004011D0 ; Attributes: bp-based frame
.text:004011D0
.text:004011D0 ; int __cdecl main(int argc,const char **argv,const char *envp)
.text:004011D0 _main      proc near          ; CODE XREF: start+AF↓p
.text:004011D0
.text:004011D0 hModule      = dword ptr -11Ch
.text:004011D0 Data         = byte ptr -118h
.text:004011D0 var_8        = dword ptr -8
.text:004011D0 var_4        = dword ptr -4
.text:004011D0 argc         = dword ptr 8
.text:004011D0 argv         = dword ptr 0Ch
.text:004011D0 envp         = dword ptr 10h
.text:004011D0
```

```





















00000000 ; -----
00000000
00000000 _cinfo      struc ; (sizeof=0x14, standard type)
00000000 MaxCharSize dd ?
00000004 DefaultChar db 2 dup(?)
00000006 LeadByte db 12 dup(?)
00000012 _padding db 2 dup(?)
00000014 _cinfo      ends
00000014
00000000 ; -----
00000000
00000000 _SECURITY_ATTRIBUTES struc ; (sizeof=0xC, standard type)
00000000 nLength      dd ?
00000004 lpSecurityDescriptor dd ? ; offset
00000008 bInheritHandle dd ?
0000000C _SECURITY_ATTRIBUTES ends
0000000C
00000000 - -----

```

- 3) All'interno del file eseguibile sono presenti 4 sezioni : **_cinfo** ; **_Security_attributes** ; **_OsVersionfoa** .

_cinfo è una struttura dati utilizzata per contenere informazioni su una pagina di codice.

_Security_Attributes è una struttura dati utilizzata per contenere le informazioni di sicurezza per un oggetto, come un file o processo. Uno degli elementi di questa struttura è **binheritHandle**, che determina se l'handle può essere ereditato da un processo figlio.

Address	Ordinal	Name	Library
 00407000		RegSetValueExA	ADVAPI32
 00407004		RegCreateKeyExA	ADVAPI32
 0040700C		SizeofResource	KERNEL32
 00407010		LockResource	KERNEL32
 00407014		LoadResource	KERNEL32
 00407018		VirtualAlloc	KERNEL32
 0040701C		GetModuleFileNameA	KERNEL32
 00407020		GetModuleHandleA	KERNEL32
 00407024		FreeResource	KERNEL32
 00407028		FindResourceA	KERNEL32
 0040702C		CloseHandle	KERNEL32
 00407030		GetCommandLineA	KERNEL32
 00407034		GetVersion	KERNEL32
 00407038		ExitProcess	KERNEL32
 0040703C		HeapFree	KERNEL32
 00407040		GetLastError	KERNEL32
 00407044		WriteFile	KERNEL32
 00407048		TerminateProcess	KERNEL32
 0040704C		GetCurrentProcess	KERNEL32
 00407050		UnhandledExceptionFilter	KERNEL32

00407050	UnhandledExceptionFilter	KERNEL32
00407054	FreeEnvironmentStringsA	KERNEL32
00407058	FreeEnvironmentStringsW	KERNEL32
0040705C	WideCharToMultiByte	KERNEL32
00407060	GetEnvironmentStrings	KERNEL32
00407064	GetEnvironmentStringsW	KERNEL32
00407068	SetHandleCount	KERNEL32
0040706C	GetStdHandle	KERNEL32
00407070	GetFileType	KERNEL32
00407074	GetStartupInfoA	KERNEL32
00407078	GetEnvironmentVariableA	KERNEL32
0040707C	GetVersionExA	KERNEL32
00407080	HeapDestroy	KERNEL32
00407084	HeapCreate	KERNEL32
00407088	VirtualFree	KERNEL32
0040708C	RtlUnwind	KERNEL32
00407090	HeapAlloc	KERNEL32
00407094	HeapReAlloc	KERNEL32
00407098	SetStdHandle	KERNEL32
0040709C	FlushFileBuffers	KERNEL32

- 4) Il malware importa librerie **ADVAPI32** e **Kerell32**. Queste funzioni possono dare un'indicazione delle funzionalità che il malware potrebbe implementare:
- RegSetValueExA**: Questa funzione è utilizzata per impostare i dati per una chiave di registro specificata. Il malware potrebbe utilizzare questa funzione per modificare le impostazioni del sistema o per garantire la persistenza nel sistema.

SizeofResource, LoadResource: Queste funzioni sono utilizzate per gestire le risorse in un file eseguibile. Il malware potrebbe utilizzare queste funzioni per accedere a risorse incorporate, come codice aggiuntivo o dati di configurazione.

VirtualAlloc: Questa funzione è utilizzata per allocare la memoria. Il malware potrebbe utilizzare questa funzione per allocare memoria per il codice o i dati che deve eseguire o memorizzare.

GetModuleFileNameA: Questa funzione recupera il percorso completo del file eseguibile del modulo corrente. Il malware potrebbe utilizzare questa funzione per scoprire la sua posizione nel sistema.

FreeLibrary: Questa funzione libera la memoria caricata con la funzione

LoadLibrary. Il malware potrebbe utilizzare questa funzione per liberare le librerie caricate dopo aver utilizzato le funzioni di cui ha bisogno.

UnhandledExceptionFilter: Questa funzione permette al malware di gestire le eccezioni non gestite. Potrebbe essere utilizzata per prevenire il crash del malware in caso di errori.

FreeEnvironmentStringsA,
FreeEnvironmentStringsW: Queste funzioni liberano un blocco di memoria che contiene le stringhe di ambiente. Il malware potrebbe utilizzare queste funzioni per pulire dopo aver letto o modificato le variabili di ambiente.

WideCharToMultiByte: Questa funzione converte le stringhe **Unicode** in stringhe di caratteri **multibyte**. Il malware potrebbe utilizzare questa funzione per manipolare le stringhe in un formato specifico.

GetEnvironmentStrings: Questa funzione recupera le stringhe di ambiente per il processo corrente. Il malware potrebbe utilizzare questa funzione per leggere le variabili di ambiente, che potrebbero

includere informazioni utili come i percorsi del sistema.

Lo scopo della funzione chiamata alla locazione di memoria **00401021 call ds : RegCreateKeyExA** è utilizzata per creare una chiave specificata nel Registro di Windows. Se la chiave esiste già, la funzione la apre.

text:00401021

call ds:RegCreateKeyExA

Di solito i parametri vengono generalmente passati a una funzione utilizzando le istruzioni **push**, per spingere un parametro nello **stack** prima della chiamata alla funzione. L'ordine in cui i parametri vengono spinti nello **stack** è generalmente inverso rispetto all'ordine in cui appaiono nella lista dei parametri della funzione.

Per la funzione **RegCreateKeyExA**, i parametri includono un handle a una chiave del Registro di sistema aperta, il nome della sottochiave del Registro di sistema da aprire, un puntatore a una struttura **SECURITY_ATTRIBUTES** che determina se l'handle restituito può essere ereditato dai processi figlio. Questi parametri vengono spinti nello **stack** prima della chiamata alla funzione.

.text:00401000	push	ebp	
.text:00401001	mov	ebp, esp	
.text:00401003	push	ecx	
.text:00401004	push	0	; lpdwDisposition
.text:00401006	lea	eax, [ebp+hObject]	
.text:00401009	push	eax	; phkResult
.text:0040100A	push	0	; lpSecurityAttributes
.text:0040100C	push	0F003Fh	; samDesired
.text:00401011	push	0	; dwOptions
.text:00401013	push	0	; lpClass
.text:00401015	push	0	; Reserved
.text:00401017	push	offset SubKey	; "SOFTWARE\\Microsoft\\Windows NT\\CurrentVe"...
.text:0040101C	push	80000002h	; hKey
.text:00401021	call	ds:RegCreateKeyExA	
.text:00401027	test	eax, eax	
.text:00401029	jz	short loc_401032	

Il parametro alla locazione di memoria 00401017 push offset SubKey ; "Software\\Microsoft\\Windows NT\\Current Version" Rappresenta un percorso nel Registro di sistema Windows. Questo percorso punta alla chiave Current Version che si trova all'interno della chiave Windows NT, che a sua volta all'interno della chiave Microsoft sotto la chiave principale Software.

La chiave **Current Version** contiene informazioni sulla versione corrente del **sistema operativo Windows**. Quindi, se un programma sta cercando di accedere a questa chiave del Registro di sistema, potrebbe essere interessato a ottenere **informazioni sulla versione di Windows** attualmente in esecuzione **sul computer**.

```
ext:00401017      push     offset SubKey      ; "SOFTWARE\\Microsoft\\Windows NT\\CurrentVe"...
```

test eax, eax: Questa istruzione esegue un'operazione **AND** tra i valori in **eax** e **eax**, ma non salva il risultato. L'obiettivo principale di questa istruzione è modificare i flag dello stato, in particolare il flag zero (**ZF**).. Se **eax** è zero, allora il risultato dell'operazione AND sarà zero, e quindi il flag zero sarà impostato a 1. Altrimenti, se **eax** non è zero, allora il risultato

dell'operazione AND non sarà zero, e il flag zero sarà impostato a 0.

jz short loc_401032: Questa è un'istruzione di salto condizionale. **jz** sta per “**jump if zero**”. Quindi, se il flag zero è impostato a 1 (cioè, se il risultato dell'ultima operazione che ha influenzato il flag zero era zero), allora il controllo salta alla posizione di memoria loc_401032. Se il flag zero non è impostato a 1, allora questa istruzione non fa nulla e il controllo passa alla prossima istruzione.

```
ext:00401027      |      test    eax, eax
ext:00401029      |      jz      short loc_401032
```

Queste istruzioni si potrebbero tradurre in costrutto C nel seguente modo :

```
if (eax == 0) {
    // Vai a loc_401032
}
```

Il valore del parametro “**ValueName**” alla locazione di memoria 00401047 è la stringa “**GinaDLL**”

00401039	. 52	PUSH EDI	Buffer
0040103A	. 6A 01	PUSH 1	ValueType = REG_SZ
0040103C	. 6A 00	PUSH 0	Reserved = 0
0040103E	. 68 4C804000	PUSH Malware_.0040804C	ValueName = "GinaDLL"
00401043	. 8B45 FC	MOV EAX, DWORD PTR SS:[EBP-4]	hKey
00401046	. 50	PUSH EAX	RegSetValueExA
00401047	. FF15 00704000	CALL DWORD PTR DS:[<&ADVAPI32.RegSetValueExA>]	

Analisi Dinamica

La chiave di registro che viene creata è **HKLM\Software\Microsoft\WindowsNT\Current Version\Winlogon.**

Il valore che vien associato a questa chiave di registro è **C:\Documents and**

Settings\Administrator\Desktop\Build_Week_Unit_3\msgina32.dll.

4:33:03.63668...	Malware_Build_Week_U3...	1428	RegCreateKey	HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon
4:33:03.63671...	Malware_Build_Week_U3...	1428	RegSetValue	HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\GinaDLL
4:33:03.64064...	Malware_Build_Week_U3...	1428	RegCloseKey	HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon

Desired Access: All Access

Type: REG_SZ, Length: 520, Data: C:\Documents and Settings\Administrator\Desktop\Build_Week_Unit_3\msgina32.dll

La chiamata di sistema **WriteFile** ha cambiato il contenuto della seguente cartella

Malware_Build_Week_U3...	1428	WriteFile	C:\Documents and Settings\Administrator\Desktop\Build_Week_Unit_3\msgina32.dll
Malware_Build_Week_U3...	1428	WriteFile	C:\Documents and Settings\Administrator\Desktop\Build_Week_Unit_3\msgina32.dll

All'interno della cartella dove è situato l'eseguibile di un malware è stata creata una libreria MSGINA32.DLL. Questa è una libreria di collegamento dinamico (DLL) fornita da Microsoft che implementa l'interfaccia grafica di accesso di Windows (GINA). Gina è l'interfaccia

che gestisce l'accesso degli utenti a un sistema Windows.

La DLL MSGina.dll fornisce funzionalità standard per l'accesso, come la visualizzazione della finestra di dialogo di accesso, l'autenticazione dell'utente e la creazione dei token di accesso.

Nel nostro contesto, msgina32.dll, è un file creato dal malware per intercettare le credenziali dell'utente.

Da queste informazioni dell'analisi statica che dall'analisi dinamica