BUILD WEEK 3

TEAM 2

Introduzione

MALWARE ANALYSIS

Cos'è il Malware?

Il termine "**malware**" è una contrazione di "**malicious software**" e si riferisce a qualsiasi software progettato per danneggiare, sfruttare o prendere il controllo di un sistema informatico senza il consenso del proprietario. I tipi comuni di malware includono **virus**, **worm**, **trojan**, **ransomware**, **spyware** e **adware**.

L'analisi del malware è il processo di studiare il comportamento e il codice di un eseguibile malevolo per capire come funziona, cosa fa, e come difendersi efficacemente.

Fasi della Malware Analysis

L'analisi del malware può essere suddivisa in diverse fasi principali:

1. Raccolta del Campione: Identificare e raccogliere i campioni di malware da analizzare.
2. Analisi Statica: Esaminare il **malware** senza eseguirlo, analizzando il codice sorgente, le stringhe e le dipendenze.
3. Analisi Dinamica: Eseguire il **malware** in un ambiente controllato per osservare il suo comportamento in tempo reale.
4. Analisi del Comportamento: Studiare come il **malware** interagisce con il sistema operativo, la rete e altre applicazioni.
5. Reverse Engineering: Utilizzare tecniche avanzate per decompilare il codice del malware e capire esattamente come funziona.

Tecniche e Strumenti

- **Sandboxing:** Esecuzione del malware in un ambiente isolato per osservare il suo comportamento senza rischiare la sicurezza del sistema principale.
- **Debugging:** Utilizzo di debugger per esaminare il flusso di esecuzione del malware e identificare le sue funzioni principali.
- **Disassembling** e **Decompiling**: Conversione del codice binario del malware in un linguaggio assembly o in un linguaggio di alto livello per analizzare la logica del programma.
- **Static Analysis Tools**: Strumenti come IDA Pro, Ghidra e radare2 per analizzare il codice del malware senza eseguirlo.
- **Dynamic Analysis Tools**: Strumenti come Cuckoo Sandbox, Wireshark e Process Monitor per osservare il comportamento del malware durante l'esecuzione.

Sfide e Considerazioni

- **Evasione delle Analisi:** Molti malware moderni utilizzano tecniche avanzate per evadere l'analisi, come la crittografia del codice, il rilevamento degli ambienti virtuali e il mutamento del codice.
- **Ambienti Controllati:** È essenziale disporre di ambienti di analisi ben isolati per evitare che il malware si diffonda o danneggi i sistemi reali.
- **Aggiornamento Costante:** Gli analisti di malware devono rimanere aggiornati con le ultime tecniche e strumenti, poiché il panorama delle minacce è in continua evoluzione.

ANALISI STATICA E DINAMICA

L'analisi statica è il processo di esaminare un file malware senza eseguirlo. Questo tipo di analisi permette di raccogliere informazioni sull'eseguibile attraverso l'ispezione del codice sorgente, dei binari e delle risorse incluse. Gli obbiettivi dell'analisi statica sono:

- **Identificare** la struttura del file.
- **Estrare stringhe** di testo leggibili che possono indicare funzionalità o obiettivi del malware.
- **Analizzare** le dipendenze e le librerie utilizzate dal malware.
- **Riconoscere** le tecniche di offuscamento e crittografia usate per nascondere il codice maligno.

I tool maggiormente utilizzati per questo tipo di analisi sono : **IDA Pro, Ghidra, radare2**

Esistono 2 tipi di analisi statica:

1) Analisi Statica Basica

L'analisi statica basica è il punto di partenza per esaminare un file malware senza eseguirlo. Questa fase include una serie di passaggi e tecniche semplici che possono fornire informazioni utili sul malware.

2) ***Analisi Statica Avanzata***

L'analisi statica avanzata va oltre i passaggi di base e richiede una comprensione più approfondita del codice e delle tecniche di offuscamento utilizzate dai malware. Questa fase coinvolge strumenti e metodi più sofisticati per ottenere una visione dettagliata del funzionamento interno del malware.

ANALISI DINAMICA

L'analisi dinamica implica l'esecuzione del malware in un ambiente controllato e monitorato per osservare il suo comportamento in tempo reale. Questo tipo di analisi consente di vedere come il malware interagisce con il sistema operativo, la rete e le altre applicazioni.

Gli obiettivi dell'analisi dinamica sono:

- ***Osservare*** il comportamento del malware, incluse le modifiche al file system, al registro di sistema e alle connessioni di rete.
- ***Identificare*** i processi creati dal malware e le risorse utilizzate.
- ***Raccogliere informazioni*** sulle tecniche di persistenza e sui metodi di comunicazione con server di comando e controllo (C&C).

I Tool maggiormente utilizzati per questo tipo di analisi sono **Process Monitor, Process Explorer e OLLYdbg.**

Esistono 2 tipi di analisi dinamica:

*1) L'**analisi dinamica basica*** implica l'esecuzione del malware in un ambiente controllato per osservare il suo comportamento in tempo reale. Questa fase include una serie di passaggi e tecniche semplici ma fondamentali per raccogliere informazioni sul malware.

*2) **L'analisi dinamica avanzata*** approfondisce ulteriormente l'osservazione del comportamento del malware, utilizzando strumenti e tecniche sofisticate per ottenere una comprensione dettagliata del funzionamento interno del malware.

GIORNO 1

Il Malware da analizzare è nella cartella **Build_Week_Unit_3** presente sul desktop della macchina virtuale dedicata.

Giorno 1: Con riferimento al file eseguibile Malware_Build_Week_U3, rispondere ai seguenti quesiti utilizzando le tecniche apprese nelle lezioni teoriche:

- Quanti parametri sono passati alla funzione Main()?
- Quante variabili sono dichiarate all'interno della funzione
-Quali sezioni sono presenti all'interno del file eseguibile? Descrivete brevemente almeno 2 di quelle identificate
- Quali librerie importa il Malware ? Per ognuna delle librerie importate, fate delle sola analisi statica delle funzionalità che il Malware potrebbe implementare. Utilizzate le funzioni che sono richiamate all'interno delle librerie per supportare le vostre ipotesi.

Utilizzeremo **CFFexplorer** e **IDApro** tool per l'analisi statica del malware

-Quanti *parametri* sono passati alla funzione Main()?
-Quante variabili sono dichiarate all'interno della funzione Main()?

IDA identifica Funzioni / chiamate di funzione, Analisi dello stack, Variabili locali e parametri, inoltre uno degli aspetti principali di IDA è la sua capacità di riconoscere le funzioni, di assegnare loro un'etichetta (detta anche label) e di evidenziare le variabili locali ed i parametri della funzione.

- **Parametro** è un argomento che viene passato alla funzione.
- Le **variabili** vengono dichiarate per riservare spazio in memoria e specificare il tipo di dati che possono contenere. Le variabili memorizzano valori che possono variare nel corso dell'esecuzione del programma.

IDA Pro ci aiuta utilizzando una convenzione:

- Le **variabili** hanno un offset **negativo**.
- I **parametri** hanno un offset **positivo**.

```
; Attributes: bp-based frame

; int __cdecl main(int argc, const char **argv,
_main proc near

hModule= dword ptr -11Ch
Data= byte ptr -118h
var_117= byte ptr -117h
var_8= dword ptr -8
var_4= dword ptr -4
argc= dword ptr  8
argv= dword ptr  0Ch
envp= dword ptr  10h

push    ebp
mov     ebp, esp
sub     esp, 11Ch
push    ebx
push    esi
push    edi
```

Variabili

```
; int __cdecl main(int argc, const char **
_main proc near

hModule= dword ptr -11Ch
Data= byte ptr -118h
var_117= byte ptr -117h
var_8= dword ptr -8
var_4= dword ptr -4
argc= dword ptr  8
argv= dword ptr  0Ch
envp= dword ptr  10h

push    ebp
mov     ebp, esp
sub     esp, 11Ch
push    ebx
push    esi
```

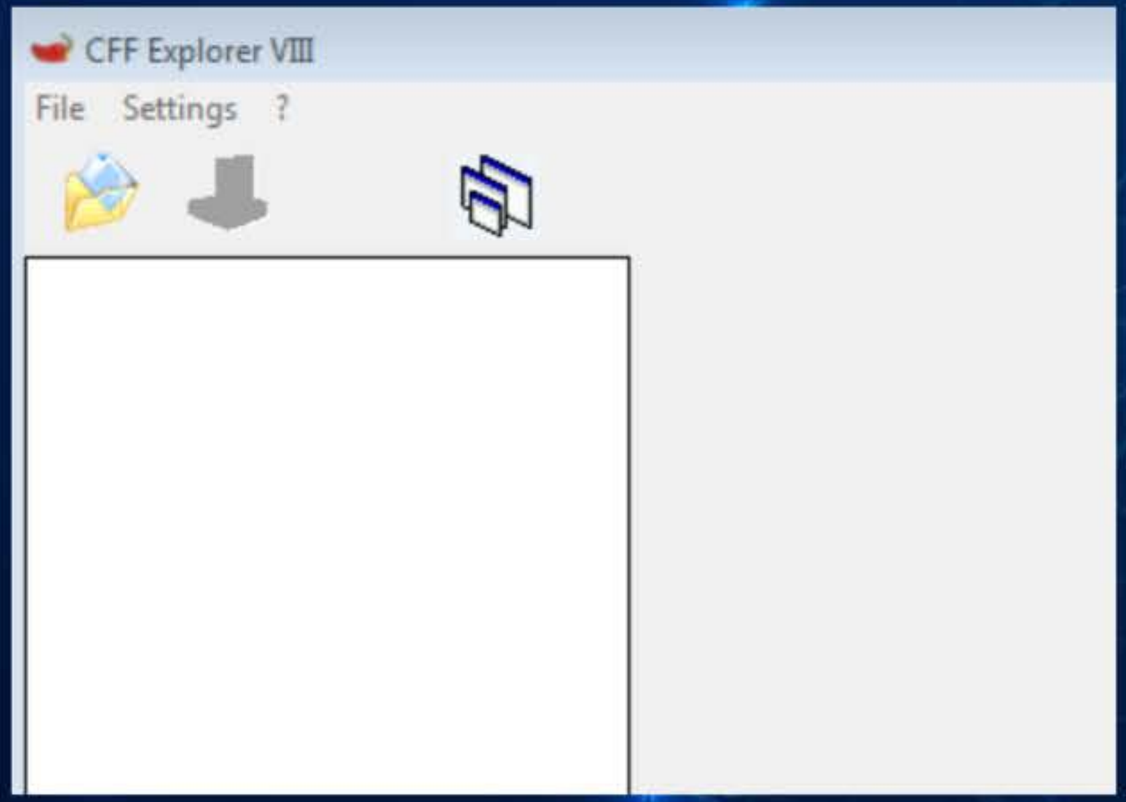
Parametri

VARIABILI

PARAMETRI

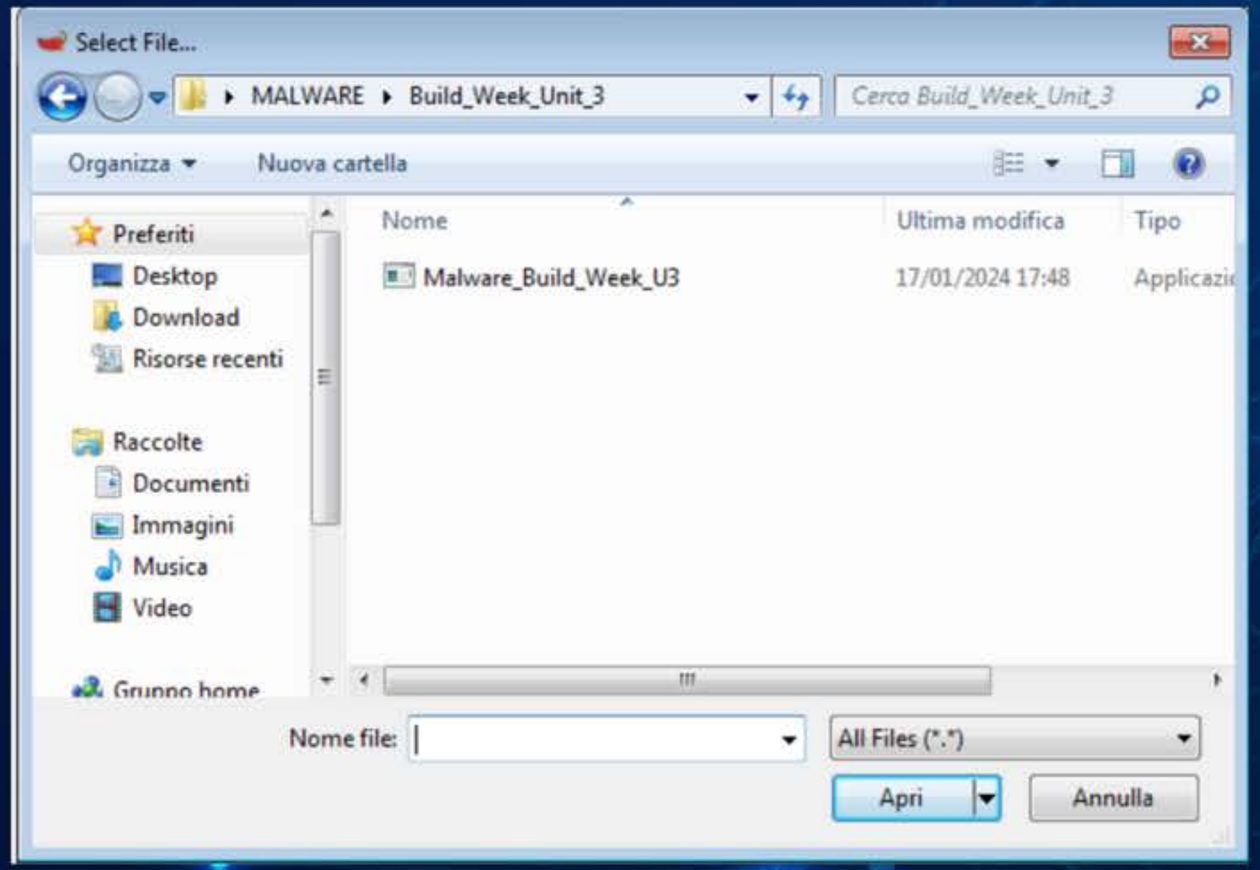
Per identificare quali librerie importa e di quante sezioni è composto il malware possiamo utilizzare il tool CFFexplorer.
Seguiamo le seguinti step per usare CFFexplorer:

- Aprire il CFFexplore
- Selezionando la cartella gialla sulla la pagina principale per importare il file eseguibile

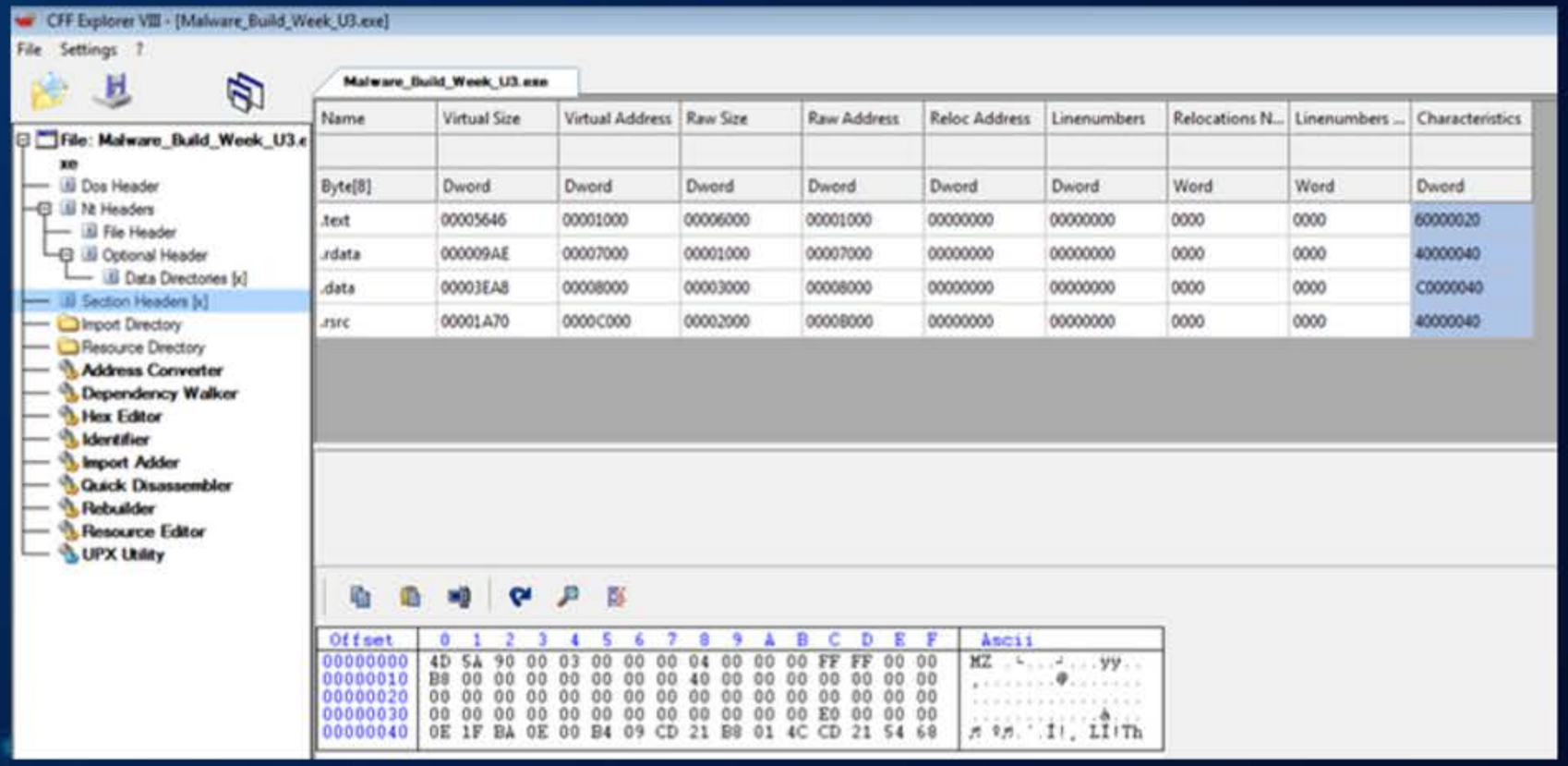


- Selezionare il file eseguibile **“Malware_Build_Week_U3”** nella cartella **“Malware”**

- Poi scelgliendo **“import directory”** dal panello principale a sinistra abbiamo le librerie importate dal file eseguibile



Quali sezioni sono presenti all'interno del file eseguibile? Descrivete brevemente almeno 2 di quelle identificate.



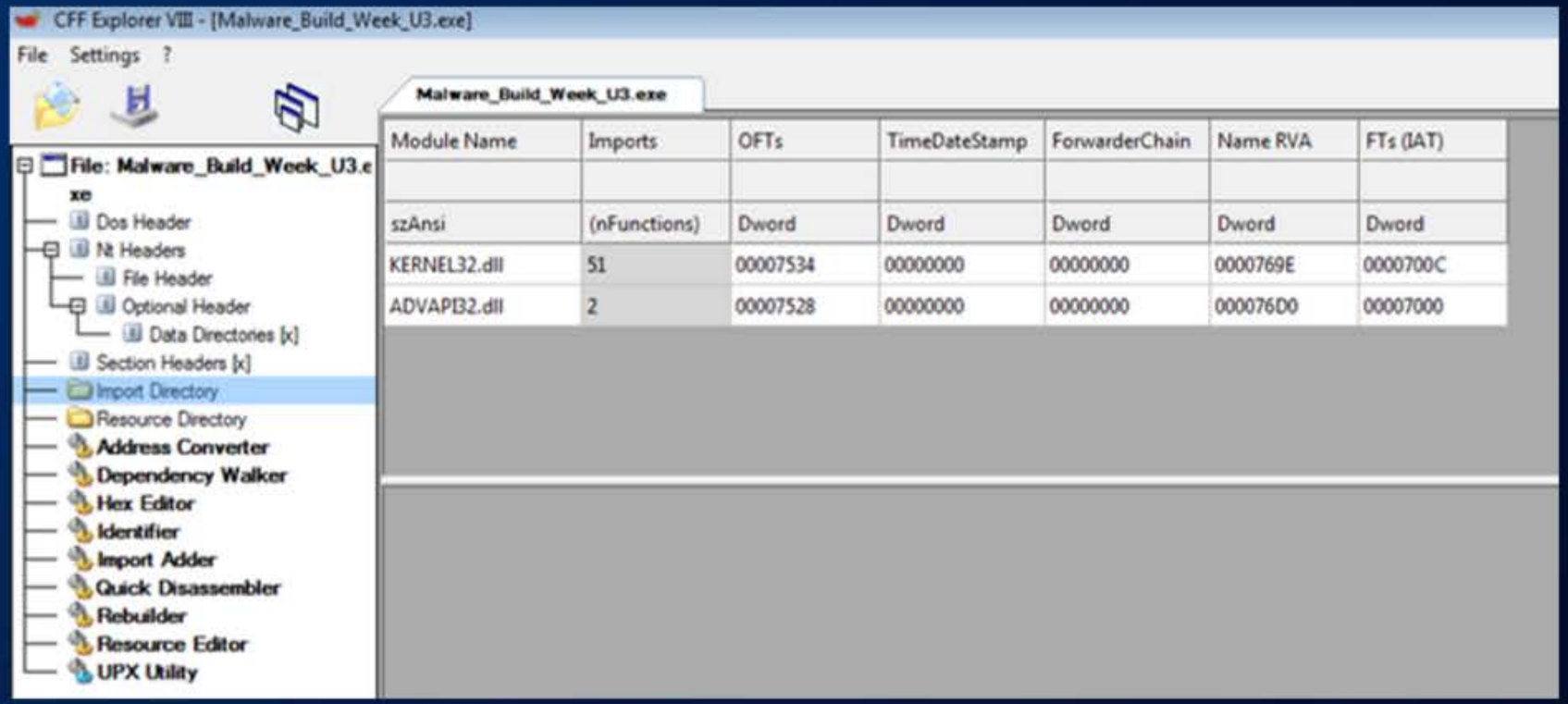
Il malware è composto da **4 sezioni**:

- **.text** : Questa è la sezione che contiene **il codice eseguibile del malware**. In altre parole, qui risiedono le istruzioni che la CPU eseguirà. La sezione è di sola **lettura** e di sola **esecuzione (RX)**, quindi non può essere **modificata** durante l'esecuzione del malware, il che protegge il codice eseguibile da alterazioni **non autorizzate**. Oltre al codice principale del malware, possono esserci delle routine critiche come l'**anti-debugging** o l'**offuscamento** per eludere l'analisi.
- **.data :** Questa sezione contiene **dati inizializzati** che il programma utilizza durante la sua esecuzione. Include **variabili globali** e **statiche** che sono state inizializzate con valori specifici. La sezione è **leggibile** e **scrivibile (RW)**, quindi i dati in questa sezione possono essere modificati durante l'esecuzione del programma. Può contenere configurazioni iniziali, **chiavi di cifratura**, **liste** di **obiettivi** o altri dati critici che il malware utilizza durante la sua esecuzione.

- **.rdata :** La sezione dei **dati di sola lettura (R)** contiene dati che non cambiano durante l'esecuzione del programma. Questo può includere stringhe, costanti, tabelle di funzioni di importazione, e altre strutture di dati che non necessitano di essere modificate. Non può essere modificata a **runtime**, garantendo che i dati rimangano invariati. Oltre ai dati di sola lettura standard, i malware possono utilizzare questa sezione per **memorizzare stringhe offuscate**, **tabelle di salti per funzioni critiche e altre informazioni che devono rimanere costanti.**

- **.rsrc** : La sezione delle risorse contiene risorse che il programma può utilizzare, come **icone, immagini, dialoghi, stringhe localizzate**, e altri tipi di dati che non fanno parte del codice eseguibile ma sono comunque necessari. Questa sezione può essere leggibile (R) e in alcuni casi scrivibile, ma di solito è di sola lettura. I malware possono nascondere payload aggiuntivi, dati di configurazione, o altre risorse offuscate all'interno di questa sezione. A volte vengono utilizzate tecniche per cifrare o comprimere queste risorse per evitare l'analisi.

Quali librerie importa il Malware ? Per ognuna delle librerie importate, fate delle sola analisi statica delle funzionalità che il Malware potrebbe implementare. Utilizzate le funzioni che sono richiamate all'interno delle librerie per supportare le vostre ipotesi.



Module Name	Imports	OFTs	TimeDateStamp	ForwarderChain	Name RVA	FTs (IAT)
szAnsi	(nFunctions)	Dword	Dword	Dword	Dword	Dword
KERNEL32.dll	51	00007534	00000000	00000000	0000769E	0000700C
ADVAPI32.dll	2	00007528	00000000	00000000	000076D0	00007000

Come possiamo notare le librerie che il malware importa sono:

- **KERNEL32.dll :** è una delle librerie principali nel sistema operativo Windows. Fornisce funzioni di base per la gestione delle risorse di sistema e l'interazione con il sistema operativo.

- **ADVAPI32.dll :** è un'altra libreria essenziale di Windows focalizzata principalmente sulla **sicurezza**, sulla **gestione dei servizi** e su altri aspetti avanzati della programmazione dell'applicazione.

Le librerie importate da un malware possono fornire indizi significativi sulle funzionalità che il malware cerca di sfruttare e rende più facile individuare il comportamento dell'eseguibile.

Il malware sembra utilizzare circa **51 funzioni** della libreria **KERNEL32.dll** e **2 funzioni** della **libreria ADVAPI32.dll.**

- **RegSetValueExA**: Questa funzione è usata per impostare il valore di un'entrata specifica nel registro di Windows. È comune nelle applicazioni Windows per salvare configurazioni o altri dati che devono persistere tra le sessioni. Tuttavia, nel contesto di un malware, questa funzione può essere usata per modificare o aggiungere chiavi di registro per garantire la persistenza del malware stesso, modificare configurazioni di sistema a favore delle attività malevole, o danneggiare configurazioni di sistema.

- **RegCreateKeyExA**: Questa funzione serve per creare una nuova chiave di registro o aprire una chiave esistente. I malware possono usare questa funzione per creare nuove chiavi di registro dove possono depositare i propri valori di configurazione o eseguibili. Questo è spesso un metodo per assicurarsi che il malware venga eseguito ad ogni avvio del sistema operativo.

Queste 2 funzioni appartengono alla libreria ADVAPI32.dll e sono utilizzate per scrivere o creare chiavi e valori nel Registro di sistema di Windows.

Come detto precedentemente il malware utilizza 51 funzioni importate dalla libreria KERNEL32.dll, troviamo funzioni per:

***1) Gestione delle Risorse*:**

- SizeofResource
- LockResource
- LoadResource
- FreeResource

Queste funzioni sono coinvolte nella **gestione e nel caricamento di risorse (come file o dati incorporati)**. Il malware potrebbe caricare risorse utili come payload, configurazioni o altri dati necessari per il funzionamento.

2) Gestione della Memoria:

- VirtualAlloc
- VirtualFree
- HeapAlloc
- HeapReAlloc
- HeapFree
- HeapCreate
- HeapDestroy

Queste funzioni sono utilizzate per **gestire la memoria nel processo del malware**. Potrebbero essere usate per allocare spazio per dati sensibili, per l'esecuzione di payload, o per effettuare

***3) Interazione con File System e Processi*:**

- GetModuleFileNameA
- GetModuleHandleA
- CloseHandle
- WriteFile
- TerminateProcess
- GetCurrentProcess
- ExitProcess

Queste funzioni permettono al malware di **interagire con i file (lettura, scrittura, gestione dei processi) e di manipolare processi e thread nel sistema operativo.**

***4) Gestione degli Errori e delle Eccezioni*:**

- GetLastError
- UnhandledExceptionFilter

Queste funzioni possono essere utilizzate per **gestire e monitorare gli errori durante l'esecuzione del malware.**

5) Interazione con l'Ambiente di Sistema:

- GetCommandLineA
- GetVersion
- GetEnvironmentVariableA
- GetVersionExA

Queste funzioni consentono al malware di **ottenere informazioni sul sistema, sull'ambiente di esecuzione e sulle variabili di sistema.**

***6) Manipolazione delle Stringhe e delle Codifiche*:**

- WideCharToMultiByte
- MultiByteToWideChar
- LCMapStringA
- LCMapStringW
- GetStringTypeA
- GetStringTypeW

Queste funzioni possono essere utilizzate per la manipolazione delle stringhe e delle codifiche per adattarsi alla configurazione e alle impostazioni del sistema.

***7) Caricamento dinamico di Librerie e Funzioni*:**

- GetProcAddress
- LoadLibraryA

Queste funzioni consentono al malware di caricare dinamicamente librerie e ottenere puntatori a funzioni, rendendo il suo comportamento più flessibile e potenzialmente evasivo.

***8) Operazioni di Input/Output sui File*:**

- FlushFileBuffers
- SetFilePointer
- CreateFileA
- ReadFile

Queste funzioni sono utilizzate per l'I/O su file, inclusa la lettura e la scrittura di dati da e verso file.

***9) Altre Operazioni di Sistema*:**

- SetHandleCount
- GetFileType
- GetStartupInfoA
- SetEndOfFile
- RtlUnwind
- GetCPInfo
- GetACP
- GetOEMCP

Queste funzioni supportano operazioni varie, come la **gestione dei handle**, informazioni di avvio, informazioni codifica dei caratteri, ecc.

CONCLUSIONI

- **Persistenza:** Il malware potrebbe scrivere nel Registro di sistema per garantire la sua esecuzione all'avvio.
- **Caricamento e Gestione delle Risorse**: Utilizzare risorse incorporate per eseguire il proprio payload o per configurazioni.
- **Manipolazione della Memoria**: Allocare spazio per dati sensibili o per eseguire operazioni avanzate di gestione della memoria.
- **Manipolazione dei File**: Leggere, scrivere o manipolare file nel sistema.
- **Interazione con il Sistema Operativo**: Ottenere informazioni di sistema, manipolare processi e thread, gestire errori.
- **Adattabilità e Dinamicità**: Caricare dinamicamente librerie e funzioni per rendere il malware più evasivo e adattabile alle condizioni del sistema.

Inoltre il file malevolo utilizza funzioni (pagina 19) particolari tipiche di un dropper ovvero un programma malevolo che contiene al suo interno un malware. Nel momento in cui viene eseguito, un dropper inizia la sua esecuzione ed estrae il malware che contiene per salvarlo sul disco.

GIORNO 2

Con riferimento al Malware in analisi, spiegare:

- Lo scopo della funzione chiamata alla locazione di memoria **00401021**
- Come vengono passati i parametri alla funzione alla locazione **00401021** ;
- Che oggetto rappresenta il parametro alla locazione **00401017**
- Il significato delle istruzioni comprese tra gli indirizzi un'altra o altre due righe assembly)
- Con riferimento all'ultimo quesito, tradurre il codice Assembly nel corrispondente costrutto C
- Valutate ora la chiamata alla locazione **00401047** , qual è il valore del parametro « **ValueName**»?

Nel complesso delle due funzionalità appena viste, spiegate quale funzionalità sta implementando il Malware in questa sezione

Lo scopo della funzione chiamata alla locazione di memoria 00401021

```
00401004  . 6A 00          PUSH 0                                   pDisposition = NULL
00401006  . 8D45 FC        LEA EAX,DWORD PTR SS:[EBP-4]
00401009  . 50             PUSH EAX                                 pHandle
0040100A  . 6A 00          PUSH 0                                   pSecurity = NULL
0040100C  . 68 3F000F00    PUSH 0F003F                              Access = KEY_ALL_ACCESS
00401011  . 6A 00          PUSH 0                                   Options = REG_OPTION_NON_VOLATILE
00401013  . 6A 00          PUSH 0                                   Class = NULL
00401015  . 6A 00          PUSH 0                                   Reserved = 0
00401017  . 68 54804000    PUSH Malware_.00408054                   Subkey = "SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon"
0040101C  . 68 02000080    PUSH 80000002                            hKey = HKEY_LOCAL_MACHINE
00401021  . FF15 04704000  CALL DWORD PTR DS:[<&ADVAPI32.RegCreate  RegCreateKeyExA
```

Per la risoluzione di questo esercizio abbiamo utilizzato il tool **OLLYdbg**. Esso è un debugger a livello di codice assembly per applicazioni Windows, noto per essere particolarmente utile per il reverse engineering.

Alla locazione di memoria **00401021** il malware chiama la funzione **RegCreateKeyExa**, essa appartiene alla famiglia delle **funzioni di gestione del registro di Windows** e permette di creare una nuova chiave o aprire una chiave esistente nel registro di sistema.
I malware possono usare questa funzione per creare nuove chiavi di registro dove possono depositare i propri valori di configurazione o eseguibili. Questo è spesso un metodo per assicurarsi che il malware venga eseguito ad ogni avvio del sistema operativo.

Come vengono passati i parametri alla funzione alla locazione 00401021 ?

I parametri vengono pushati nello stack in ordine inverso rispetto alla loro definizione nella firma della funzione **RegCreateKeyExA**. Questi valori vengono successivamente utilizzati dalla funzione chiamata per eseguire le operazioni specifiche. La chiamata a **RegCreateKeyExA** avviene quindi con tutti i parametri necessari correttamente posizionati nello stack.

Il motivo per cui i parametri vengono passati nello stack dall'ultimo al primo è legato alla convenzione di chiamata utilizzata. In particolare, questa convenzione si chiama **__cdecl**.

Che oggetto rappresenta il parametro alla locazione 00401017

Il parametro alla locazione **00401017** è **"Malware_00400054"**. Questo può essere dedotto dalla riga di codice di assembly che lo include come parametro per l'istruzione **PUSH**.

Il codice stia preparando i parametri per una chiamata alla funzione **RegCreateKeyExA**, i parametri passati alla funzione tramite lo stack includono diverse costanti e puntatori che rappresentano vari parametri della funzione stessa, tra cui il percorso della chiave di registro.

"Malware_00400054" sembra essere un'etichetta che rappresenta una stringa o una risorsa specifica utilizzata nella chiamata di funzione **RegCreateKeyExA**.

Il significato delle istruzioni comprese tra gli indirizzi 00401027 e 00401029 .

```
00401021   FF15 04704000   CALL DWORD PTR DS:[<&ADVAPI32.Reg
00401027   85C0            TEST EAX,EAX
00401029  v74 07           JE SHORT Malware_.00401032
0040102B   B8 01000000     MOV EAX,1
```

- **TEST eax, eax**: Questa istruzione esegue un'**operazione logica AND** tra il **registro eax** e se stesso. L'istruzione aggiorna i flag del processore in base al risultato di questa operazione. Se **eax è zero**, il flag **zero** (**ZF**) verrà impostato; altrimenti, sarà resettato. In sostanza, questa istruzione verifica se **eax è zero** senza alterarne il contenuto

- **JE short malware_.00401032**: Questa istruzione è un salto condizionale. **JE** (**Jump if Equal**) è vero se il flag zero (ZF) è impostato. In altre parole, JE esegue il **salto all'indirizzo 00401032** se il risultato dell'istruzione precedente (**TEST eax, eax**) era zero, cioè se **eax** è **zero**.

Con riferimento all'ultimo quesito, tradurre il codice Assembly nel corrispondente costrutto C

```c
int eax = /* valore di eax */;

if (eax == 0) {
    goto malware_00401032;
}

// Codice successivo alla condizione

malware_00401032:
// Codice all'etichetta 00401032.

```

IL codice Assembly in oggetto può essere tradotto in linguaggio **C** come una semplice condizione che controlla se una **variabile è zero** e, se lo è, salta a una particolare etichetta o esegue un blocco di codice specifico.

Valutate ora la chiamata alla locazione 00401047 , qual è il valore del parametro « ValueName»?

```
00401036  . 8B55 08        MOV EDX,DWORD PTR SS:[EBP+8]
00401039  . 52             PUSH EDX                                  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]
00401046  . 50             PUSH EAX                                  hKey
00401047  . FF15 00704000  CALL DWORD PTR DS:[<&ADVAPI32.RegSetValueExA>]   RegSetValueExA
```

OLLYdbg ci restituisce il valore del parametro "**ValueName**" ovvero "**GINADLL**"

Nel complesso delle due funzionalità appena viste, spiegate quale funzionalità sta implementando il Malware in questa sezione.

1)**Persistenza**: Utilizzando **RegCreateKeyExA** e **RegSetValueExA**, il malware sta cercando di configurarsi per eseguirsi automaticamente all'avvio del sistema. Questo è un comportamento comune nei malware per assicurarsi che possano essere eseguiti senza l'interazione dell'utente.

2)**Evasione** e **Persistenza Avanzata**: La scelta della chiave **Winlogon** indica un tentativo di nascondere la propria presenza tra i processi critici del sistema operativo, che sono fondamentali per l'accesso e l'autenticazione degli utenti.

3)**Potenziale per Attività Dannose**: Configurando il sistema per caricare una **DLL specifica** (**GinaDLL**) durante l'avvio, il malware potrebbe estendere le proprie capacità, come il **monitoraggio delle attività dell'utente, il furto di credenziali di accesso, o l'esecuzione di operazioni non autorizzate.**

In sintesi, il malware sta implementando la funzionalità di modifica del registro di Windows. Prima crea o apre una chiave di registro (**"SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon"**) e poi imposta un valore ("**GinaDLL**") all'interno di questa chiave. Queste azioni sono tipiche dei malware che tentano di persistere sul sistema o alterare il comportamento di Windows, in particolare il processo di **logon**.

GIORNO 3

Riprendete l'analisi del codice, analizzando le routine tra le locazioni di memoria ***00401080*** e ***00401128*** :

• Qual è il valore del parametro «ResourceName» passato alla funzione FindResourceA ();

• Il susseguirsi delle chiamate di funzione che effettua Malware in questa sezione di codice l'abbiamo visto durante le lezioni teoriche. Che funzionalità sta implementando il Malware ?

• È possibile identificare questa funzionalità utilizzando l'analisi statica basica?

• In caso di risposta affermativa, elencare le evidenze a supporto.

Entrambe le funzionalità principali del Malware viste finora sono richiamate all'interno della funzione Main(). Disegnare un diagramma di flusso (inserite all'interno dei box solo le informazioni circa le funzionalità principali) che comprenda le 3 funzioni.

Qual è il valore del parametro «ResourceName» passato alla funzione FindResourceA ()

```
sub_401299
_fclose
_fwrite
__fsopen
_fopen
_strchr

.text:004010BD    push    eax             ; lpType
.text:004010BE    mov     ecx, lpName
.text:004010C4    push    ecx             ; lpName
.text:004010C5    mov     edx, [ebp+hModule]
.text:004010C8    push    edx             ; hModule
.text:004010C9    call    ds:FindResourceA

; HRSRC __stdcall FindResourceA(HMODULE hModule, LPCSTR lpName, LPCSTR lpType)
extrn FindResourceA:dword ; CODE XREF: sub_401080+49↑p
                          ; DATA XREF: sub_401080+49↑r
```

La funzione FindResourceA accetta tre parametri:

1) hModule: un handle del modulo contenente la risorsa.
2) pName: il nome della risorsa.
3) lpType: il tipo della risorsa.

```
004010B1  . 33C0          XOR EAX,EAX
004010B3  .v E9 07010000  JMP Malware_.004011BF
004010B8  > A1 30804000   MOV EAX,DWORD PTR DS:[408030]
004010BD  . 50            PUSH EAX                          ResourceType => "BINARY"
004010BE  . 8B0D 34804000 MOV ECX,DWORD PTR DS:[408034]     Malware_.00408038
004010C4  . 51            PUSH ECX                          ResourceName => "TGAD"
004010C5  . 8B55 08       MOV EDX,DWORD PTR SS:[EBP+8]
004010C8  . 52            PUSH EDX                          hModule
004010C9  . FF15 28704000 CALL DWORD PTR DS:[<&KERNEL32.FindResou  FindResourceA
```

Il tool **OLLYdbg** ci restituisce il valore del parametro ResourceName passato alla funzione FindResourceA() che è = ***" TGAD"***

Il susseguirsi delle chiamate di funzione che effettua Malware in questa sezione di codice l'abbiamo visto durante le lezioni teoriche. Che funzionalità sta implementando il Malware ?

```
4010C5                  mov     edx, [ebp+hModule]
4010C8                  push    edx             ; hModule
4010C9                  call    ds:FindResourceA
4010CF                  mov     [ebp+hResInfo], eax
4010D2                  cmp     [ebp+hResInfo], 0
4010D6                  jnz     short loc_4010DF
4010D8                  xor     eax, eax
4010DA                  jmp     loc_4011BF
4010DF ; ---------------------------------------------------------------------------
4010DF
4010DF loc_4010DF:                              ; CODE XREF: sub_401080+56↑j
4010DF                  mov     eax, [ebp+hResInfo]
4010E2                  push    eax             ; hResInfo
4010E3                  mov     ecx, [ebp+hModule]
4010E6                  push    ecx             ; hModule
4010E7                  call    ds:LoadResource
4010ED                  mov     [ebp+hResData], eax
4010F0                  cmp     [ebp+hResData], 0
4010F4                  jnz     short loc_4010FB
4010F6                  jmp     loc_4011A5
4010FB loc_4010FB:                              ; CODE XREF: sub_401080+74↑j
4010FB                  mov     edx, [ebp+hResData]
4010FE                  push    edx             ; hResData
4010FF                  call    ds:LockResource
401105                  mov     [ebp+Str], eax
401108                  cmp     [ebp+Str], 0
40110C                  jnz     short loc_401113
40110E                  jmp     loc_4011A5
401113 ; ---------------------------------------------------------------------------
401113
401113 loc_401113:                              ; CODE XREF: sub_401080+8C↑j
401113                  mov     eax, [ebp+hResInfo]
401116                  push    eax             ; hResInfo
401117                  mov     ecx, [ebp+hModule]
40111A                  push    ecx             ; hModule
40111B                  call    ds:SizeofResource
401121                  mov     [ebp+Count], eax
```

Le funzionalità che sta implementando il malware sono:

FindResourceA:
Trova una risorsa incorporata nell'eseguibile dell'applicazione, restituendo un handle che può essere usato per caricarla.

LoadResource:
Carica la risorsa trovata in memoria, restituendo un handle della risorsa caricata.

LockResource:
Converte l'handle della risorsa caricata in un puntatore alla memoria, permettendo di accedere direttamente ai dati della risorsa.

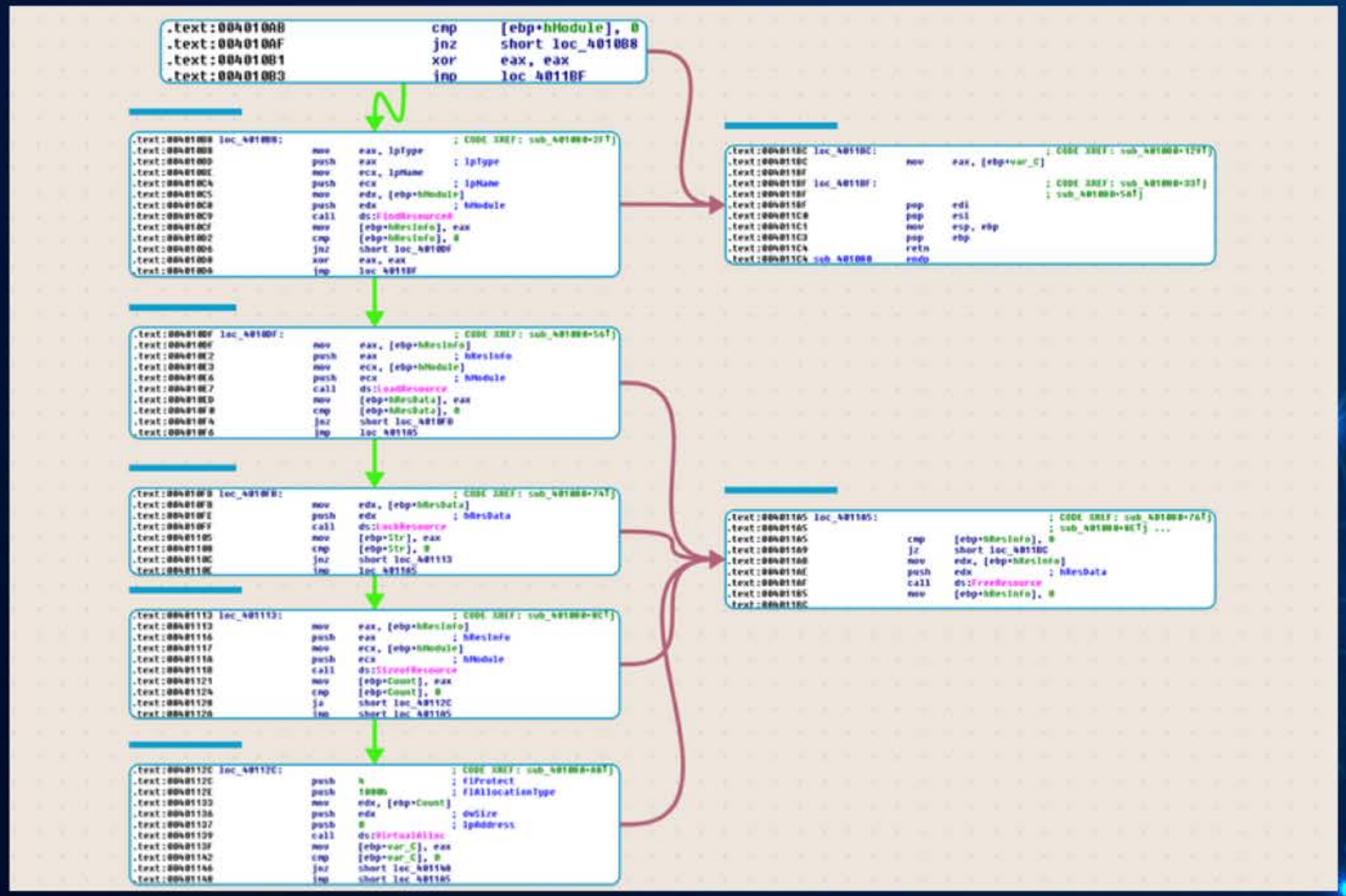
SizeofResource:
Ottiene la dimensione, in byte, della risorsa specificata.

- ***È possibile identificare questa funzionalità utilizzando l'analisi statica basica?***
- ***In caso di risposta affermativa, elencare le evidenze a supporto.***

OFTs	FTs (IAT)	Hint	Name
Dword	Dword	Word	szAnsi
00007632	00007632	0295	SizeofResource
00007644	00007644	01D5	LockResource
00007654	00007654	01C7	LoadResource
00007622	00007622	02BB	VirtualAlloc
00007674	00007674	0124	GetModuleFileNameA
0000768A	0000768A	0126	GetModuleHandleA
00007612	00007612	00B6	FreeResource
00007664	00007664	00A3	FindResourceA

E' possibile identificare le funzionalità utilizzando l'analisi statica basica con i tool CFFexplorer ed IDA pro, come mostrato in figura.

Entrambe le funzionalità principali del Malware viste finora sono richiamate all'interno della funzione Main(). Disegnare un diagramma di flusso (inserite all'interno dei box solo le informazioni circa le funzionalità principali) che comprenda le 3 funzioni.



GIORNO 4

Preparate l'ambiente ed i tool per l'esecuzione del Malware (suggerimento: avviate principalmente Esercizio Giorno 4 Process Monitor ed assicurate di eliminare ogni filtro cliccando sul tasto «reset» quando richiesto in fase di avvio). Eseguite il Malware , facendo doppio click sull'icona dell'eseguibile.

-Cosa notate all'interno della cartella dove è situato l'eseguibile del Malware? Spiegate cosa è avvenuto, unendo le evidenze che avete raccolto finora per rispondere alla domanda.

Filtrate includendo solamente l'attività sul **registro di Windows.**

-Quale chiave di registro viene creata?

-Quale valore viene associato alla chiave di registro creata?

Passate ora alla visualizzazione dell'attività sul **File System.**

- Quale chiamata di sistema ha modificato il contenuto della cartella dove è presente l'eseguibile del Malware ?

Unite tutte le informazioni raccolte fin qui sia dall'analisi statica che dall'analisi dinamica per delineare il funzionamento del Malware .

Malware Analysis

Giorno 4:

- Cosa notate all'interno della cartella dove è situato l'eseguibile del Malware? Spiegate cosa è avvenuto, unendo le evidenze che avete raccolto finora per rispondere alla domanda

Analizzate ora i risultati di Process Monitor (consiglio: utilizzate il filtro come in figura sotto per estrarre solo le modifiche apportate al sistema da parte del Malware). Fate click su «ADD» poi su «Apply» come abbiamo visto nella lezione teorica.

Malware_Build_Week_U3	17/01/2024 17:48	Applicazione	52 KB
msgina32.dll	25/06/2024 11:48	Estensione dell'ap...	7 KB

msgina32.dll è una componente essenziale dei vecchi sistemi operativi Windows, responsabile della gestione delle schermate di autenticazione e delle interfacce di accesso utente. La sua funzione primaria è garantire un processo di login sicuro e stabile, fornendo al contempo flessibilità per la personalizzazione delle interfacce di accesso.

REGISTRO DI WINDOWS .

*- Quale **chiave di registro** viene creata?-*

Quale valore viene associato alla chiave di registro creata?

11:48:31,3134589	Malware_Build_Week_U3.exe	3056	RegCreateKey	HKLM\SOFTWARE\Wow6432Node\Microsoft\Windows NT\CurrentVersion\Winlogon	SUCCESS	Desired Access: All Access, Disposition: REG
11:48:31,3135249	Malware_Build_Week_U3.exe	3056	RegSetInfoKey	HKLM\SOFTWARE\Wow6432Node\Microsoft\Windows NT\CurrentVersion\Winlogon	SUCCESS	KeySetInformationClass: KeySetHandleTagsI...
11:48:31,3135496	Malware_Build_Week_U3.exe	3056	RegQueryKey	HKLM\SOFTWARE\Wow6432Node\Microsoft\Windows NT\CurrentVersion\Winlogon	SUCCESS	Query: HandleTags, HandleTags: 0x400
11:48:31,3135647	Malware_Build_Week_U3.exe	3056	RegSetValue	HKLM\SOFTWARE\Wow6432Node\Microsoft\Windows NT\CurrentVersion\Winlogon\GinaDLL	ACCESS DENIED	Type: REG_SZ, Length: 520, Data: C:\Users

FILE SYSTEM

Quale chiamata di sistema ha modificato il contenuto della cartella dove è presente l'eseguibile del Malware ?

11:48:31,3121231	Malware_Build_Week_U3.exe	3056	IRP_MJ_CREATE	C:\Users\user\Desktop\MALWARE\Build_Week_Unit_3\msgina32.dll	SUCCESS
11:48:31,3125334	Malware_Build_Week_U3.exe	3056	IRP_MJ_WRITE	C:\Users\user\Desktop\MALWARE\Build_Week_Unit_3\msgina32.dll	SUCCESS
11:48:31,3126991	Malware_Build_Week_U3.exe	3056	FASTIO_WRITE	C:\Users\user\Desktop\MALWARE\Build_Week_Unit_3\msgina32.dll	FAST IO DISALLOWED
11:48:31,3127122	Malware_Build_Week_U3.exe	3056	IRP_MJ_WRITE	C:\Users\user\Desktop\MALWARE\Build_Week_Unit_3\msgina32.dll	SUCCESS
11:48:31,3127720	Malware_Build_Week_U3.exe	3056	IRP_MJ_CLEANUP	C:\Users\user\Desktop\MALWARE\Build_Week_Unit_3\msgina32.dll	SUCCESS
11:48:31,3128940	Malware_Build_Week_U3.exe	3056	IRP_MJ_CLOSE	C:\Users\user\Desktop\MALWARE\Build_Week_Unit_3\msgina32.dll	SUCCESS

Dalle analisi condotte, emerge che il malware tenta di alterare il meccanismo di autenticazione di Windows, manipolando specificamente la **DLL Gina (Graphical Identification and Authentication)**. Questo componente è cruciale nel gestire le interfacce grafiche di autenticazione, essendo fondamentale per la presentazione della schermata di login e la verifica delle credenziali degli utenti.

Modificando la DLL Gina, il malware può introdurre comportamenti altamente nocivi. Ad esempio, è possibile che intercetti e sottragga le credenziali degli utenti, facilitando accessi illeciti al sistema. Inoltre, la manipolazione di questa DLL può consentire al malware di eludere le misure di sicurezza esistenti, istituendo una **backdoor** che offre agli aggressori un accesso continuativo e privilegiato al sistema compromesso.

Questo tipo di attacco rappresenta una minaccia particolarmente insidiosa, operando a un livello profondo del sistema operativo e complicando significativamente la sua rilevazione e rimozione. La compromissione della DLL Gina può gravemente minare la sicurezza del sistema, mettendo a rischio l'integrità e la riservatezza dei dati archiviati.

Per contravvenire a tali minacce, è imperativo implementare contromisure robuste, includendo l'uso di software antivirus costantemente aggiornati, l'applicazione di politiche di sicurezza stringenti e l'adozione di tecniche avanzate per il rilevamento di anomalie. Queste ultime dovrebbero essere capaci di identificare comportamenti anomali a livello di autenticazione. È inoltre essenziale mantenere il sistema operativo aggiornato e applicare prontamente tutte le patch di sicurezza rilasciate dai fornitori

GIORNO 5

GINA (Graphical identification and authentication) è un componente lecito di Windows che permette l'autenticazione degli utenti tramite interfaccia grafica ovvero permette agli utenti di inserire username e password nel classico riquadro Windows, come quello in figura a destra che usate anche voi per accedere alla macchina virtuale.

• Cosa può succedere se il file . dll lecito viene sostituito con un file . dll malevolo, che intercetta i dati inseriti?

Sulla base della risposta sopra, delineate il profilo del Malware e delle sue funzionalità. Unite tutti i punti per creare un grafico che ne rappresenti lo scopo ad alto livello.

IL malware analizzato ,una volta avviato, sostituisce il file dll.lecito con un file dll.malevolo il cui scopo è quello di intercettare dati sensibili come username e password per loggare su GINA.
Una volta che il malcapitato andrà ad inserire le credenziali il malware salverà gli input su un file oppure li invierà in un server sotto il controllo dell'attaccante (quest'ultimo è possibile solo se il malware importa la libreria Wininet32.dll).

Quando il file .DLL lecito utilizzato da GINA viene sostituito con un file .DLL malevolo, le conseguenze possono essere gravi:

1) Intercettazione delle Credenziali di Accesso:
Il .DLL malevolo può registrare username e password inseriti dagli utenti durante il processo di login. Queste credenziali possono essere poi inviate a un attaccante o utilizzate per ulteriori accessi non autorizzati al sistema.
2) Accesso Non Autorizzato:
Con le credenziali intercettate, l'attaccante può ottenere accesso non autorizzato al sistema, permettendo potenzialmente l'accesso a dati sensibili, l'installazione di altri malware o la manipolazione dei dati.
3) Persistenza del Malware:
Il malware può impostare backdoor per mantenere l'accesso persistente al sistema anche dopo che l'utente cambia la propria password o rimuove altre parti del malware.
4) Modifica del Comportamento del Sistema:
Un .DLL malevolo può alterare il comportamento del sistema operativo, ad esempio disabilitando funzionalità di sicurezza, registrando attività dell'utente, o manipolando file di sistema.
5) Escalation dei Privilegi:
Utilizzando le credenziali amministrative, il malware può eseguire operazioni che richiedono alti livelli di accesso, come modificare i registri di sistema, installare altri software malevoli, o accedere a ulteriori risorse di rete.

Utilizzando il tool CFFexplorer siamo andati a vedere quali librerie e funzioni importi il file malevolo msgina32.dll.

msgina32.dll

Module Name	Imports	OFTs	TimeDateStamp	ForwarderChain	Name RVA	FTs (IAT)
00000E24	N/A	00000C7C	00000C80	00000C84	00000C88	00000C8C
szAnsi	(nFunctions)	Dword	Dword	Dword	Dword	Dword
KERNEL32.dll	12	000020F0	00000000	00000000	00002224	00002010
MSVCRT.dll	11	00002124	00000000	00000000	00002288	00002044
ADVAPI32.dll	3	000020E0	00000000	00000000	000022F2	00002000
USER32.dll	1	00002154	00000000	00000000	0000230C	00002074

Il tool ci restituisce questa schermata, il file malevolo importa 4 librerie:

-KERNEL32.DLL

-MSVCRT.DLL

-ADVAPI32.DLL

-USER32.DLL

MSVCRT.DLL = è una libreria di runtime di Microsoft C utilizzata da molte applicazioni Windows. Contiene implementazioni di funzioni standard del C, come manipolazione di stringhe, gestione della memoria, I/O, ecc. Il malware può utilizzare MSVCRT.DLL per sfruttare queste funzioni per attività dannose. L'uso di queste funzioni rende il malware più compatibile e leggero, dato che la libreria è ampiamente disponibile e utilizzata in molte versioni di Windows.

USER32.DLL = è una libreria di sistema di Windows che contiene funzioni per gestire l'interfaccia utente, le finestre, e gli input da tastiera e mouse. Questa DLL è spesso utilizzata dai malware per intercettare e manipolare le interazioni dell'utente con il sistema operativo.

Il file malevolo importa la libreria KERNEL32:DLL e utilizza 12 funzioni, esse vengono spesso utilizzate nei malware per una serie di operazioni, tra cui la manipolazione delle stringhe, la gestione delle librerie dinamiche (DLL), l'acquisizione di informazioni sul sistema e la gestione della memoria.

lstrcatW = Concatena due stringhe di caratteri wide (Unicode). Potrebbe essere utilizzata per costruire percorsi di file o URL combinando diverse parti di stringhe.

GetSystemDirectoryW = Recupera il percorso della directory di sistema ed è utile per individuare la posizione della DLL lecito di GINA per sostituirlo con la versione malevola.

DisableThreadLibraryCalls = Disabilita le notifiche DLL_THREAD_ATTACH e DLL_THREAD_DETACH per la DLL chiamante. Viene utilizzata per ridurre il carico sul processo e minimizzare la possibilità di rilevamento della DLL malevola.

lstrlenW = Calcola la lunghezza di una stringa di caratteri wide (Unicode). Necessaria per determinare la lunghezza delle stringhe quando si eseguono operazioni di manipolazione delle stesse.

LoadLibraryW = Carica una libreria dinamica (DLL) nel processo chiamante e viene usata per caricare DLL lecito o altre DLL necessarie per le operazioni del malware.

lstrcpyW = Copia una stringa di caratteri wide (Unicode) in un'altra e viene utilizzata per duplicare stringhe, come percorsi di file o dati.

LocalFree = Libera la memoria allocata localmente e viene usata per liberare memoria precedentemente allocata per prevenire perdite di memoria.

FormatMessageW = Formatta un messaggio di errore basato sul codice di errore fornito. Usata per ottenere messaggi di errore comprensibili durante l'esecuzione del malware, per il debug o la gestione degli errori.

FreeLibrary = Libera la DLL specificata e decrementa il contatore di riferimento del modulo. Usata per scaricare DLL non più necessarie dal processo, riducendo l'impronta del malware.

GetProcAddress = Recupera l'indirizzo di una funzione esportata da una DLL. Usata per ottenere puntatori a funzioni specifiche nelle DLL caricate dinamicamente.

ExitProcess = Termina il processo chiamante e tutti i thread. Usata per terminare il malware una volta completata la sua operazione, se necessario.

GetModuleFileNameW = Recupera il percorso completo del file eseguibile del modulo specificato. Usata per ottenere il percorso del file DLL lecito per la sostituzione con il DLL malevolo.

Il file malevolo utilizza 11 funzioni importate dalla libreria MSVCRT.DLL, sono ampiamente utilizzate nei programmi C/C++ per operazioni di I/O, gestione della memoria e altre operazioni di utilità. Quando queste funzioni vengono trovate nel codice di un malware, vengono utilizzate per eseguire operazioni malevole in modo efficiente.

_wfopen = Apre un file con un nome wide (Unicode) e una modalità specificata (lettura, scrittura, ecc.). Il malware potrebbe utilizzare _wfopen per aprire file di configurazione o file di log per lettura/scrittura di dati sensibili o per registrare informazioni rubate.

_vsnwprintf = Scrive una stringa formattata in un buffer, utilizzando un elenco di argomenti variabili. Usata per costruire stringhe dinamiche, come messaggi di errore, log di attività o costruzione di payload da inviare a un server remoto.

_initterm = Chiama un array di puntatori a funzioni di inizializzazione e terminazione C runtime. Il malware potrebbe usare _initterm per inizializzare variabili globali o eseguire codice necessario all'avvio.

malloc = Alloca memoria dinamicamente. Il malware utilizza malloc per allocare memoria per buffer temporanei, strutture di dati o altre risorse necessarie per le operazioni malevole.

free = Libera la memoria precedentemente allocata con malloc. Utilizzata per gestire la memoria dinamica allocata dal malware, prevenendo memory leak.

_adjust_fdiv = Funzione interna della libreria CRT utilizzata per correggere errori di divisione floating-point (può non essere sempre presente nei malware comuni). Non tipicamente utilizzata direttamente in un contesto di malware. Potrebbe essere parte del codice compilato che utilizza operazioni in virgola mobile.

_wstrdate = Recupera la data di sistema come stringa wide (Unicode). Il malware può utilizzare questa funzione per registrare timestamp nei log di attività o per determinare l'ora corrente per decisioni basate sul tempo.

fclose = Chiude un file aperto. Utilizzata per chiudere file aperti con _wfopen, garantendo che i dati siano scritti correttamente e liberando risorse.

??2@YAPAXI@Z (operator new) = Funzione di allocazione della memoria di C (equivalente a new). Viene usata per allocare memoria per oggetti in C in modo dinamico.

fwprintf = Scrive una stringa formattata in un file, utilizzando stringhe wide (Unicode). Utilizzata per scrivere log o altre informazioni formattate in file, utile per registrare dati rubati o attività del malware.

IL file msgina32.dll utilizza 3 funzioni importate dalla libreria ADVAPI32.DLL e sono utilizzate per interagire con il registro di sistema di Windows. Queste funzioni sono spesso usate dai malware per configurare l'ambiente di esecuzione, persistere nel sistema, o modificare impostazioni di sistema critiche.

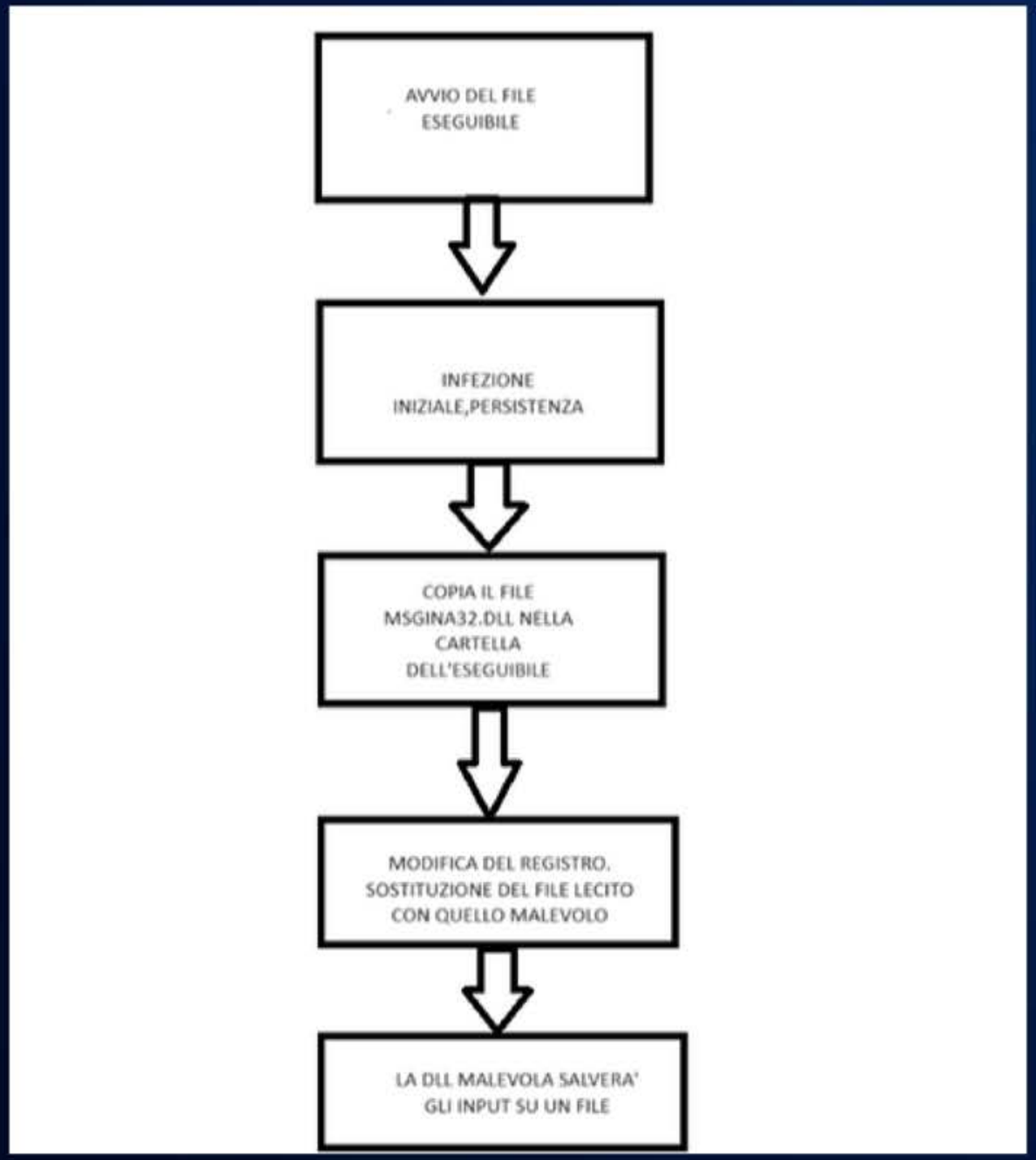
RegSetValueExW = Imposta il valore di una voce del registro specificata. Usata per modificare valori di configurazione nel registro, come l'aggiunta di voci per eseguire il malware all'avvio del sistema.

RegCreateKeyW = Crea una chiave del registro o apre una chiave esistente. Usata per creare nuove chiavi nel registro, spesso per impostare configurazioni o per garantire la persistenza del malware.

RegCloseKey = Chiude un handle aperto a una chiave del registro. Usata per chiudere chiavi del registro una volta che il malware ha finito di modificarle, liberando le risorse.

Il file malevolo importa la funzione wsprintfA inclusa nella libreria USER32.DLL di Windows. Questa funzione è utilizzata per formattare una stringa di caratteri in un buffer. La "A" alla fine del nome indica che la funzione accetta stringhe ASCII come input.

Può essere utilizzata per creare stringhe formattate dinamicamente. Questo è particolarmente utile per la creazione di log, messaggi di errore, comandi da eseguire, o per la costruzione di payload da inviare a un server remoto oppure potrebbe essere utilizzata per nascondere attività malevole o per comunicare con altri componenti del sistema senza essere facilmente rilevata.



Delineate il profilo del Malware e delle sue funzionalità.Unite tutti i punti per creare un grafico che ne rappresenti lo scopo ad alto livello.

Il malware in oggetto sembra comportarsi come un TROJAN che nasconde(molto probabilmente) delle funzionalità tipiche di un DROPPER.

Lo scopo di questo malware, come già indicato a PAGINA 45, è quello di sostituire un file .dll lecito con uno malevolo con lo scopo di entrare in possesso delle credenziali di accesso di GINA e di salvarli su un file.

Travelling around the

Epicode

By Stefano Cesaroni