BUILD WEEK 3

TEAM 3:
Valerio Zampone
Danilo Malagoli,
Carmela Ferrandina,
Federico Savi,
Giammarco Iorio

GIORNO 1

Con riferimento al file eseguibile Malware_Build_Week_U3, rispondere ai seguenti quesiti utilizzando i tool e le tecniche apprese nelle lezioni teoriche:

- Quanti parametri sono passati alla funzione Main()?

- Quante variabili sono dichiarate all'interno della funzione Main()?

- 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 ipotesi sulla base della 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.

TOOLS

Per completare l’esercizio di oggi utilizzeremo:

IDA pro 

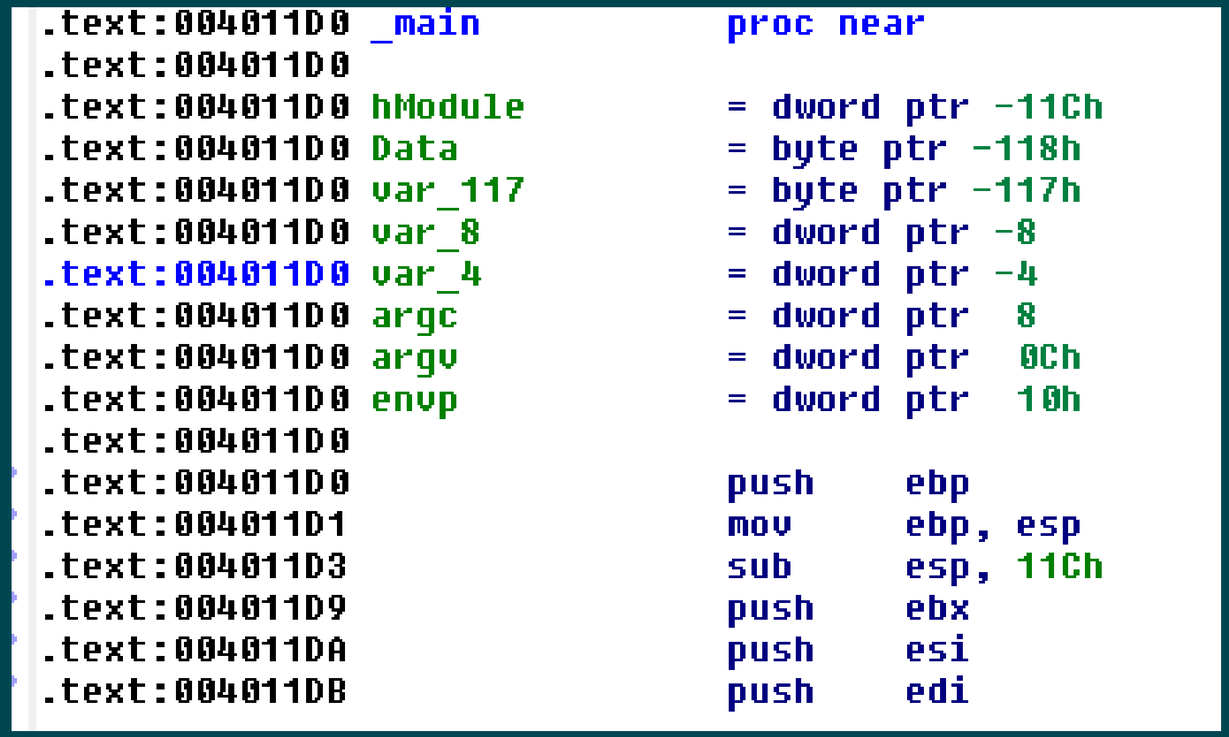
Avanzato disassemblatore e debugger utilizzato per analizzare il codice macchina e trasformarlo in codice assembly leggibile. È essenziale per il reverse engineering e supporta molte architetture.

CFF Explorer 🌶️

Strumento che permette di visualizzare e modificare file eseguibili Windows, come EXE e DLL. È utilizzato per esaminare le strutture dei file, analizzare le dipendenze e ottimizzare le prestazioni dei binari.

PARAMETRI E VARIABILI

Nell’immagine su IDA pro possiamo osservare la funzione “main” che include diversi parametri e variabili. I parametri individuati sono argc , argv, envp, ovvero quelli con offset positivo.
Invece le variabili sono hModule, Data, var117, var8 e var4, ovvero quelle con offset negativo.



PARAMETRI

- **argc** (argument count): È un intero che rappresenta il numero di argomenti passati alla funzione main. Include il nome del programma stesso, quindi il conteggio minimo sarà sempre almeno 1.
- **argv** (argument vector): È un array di stringhe (puntatori a carattere) che rappresenta gli argomenti passati alla funzione main. Ogni elemento dell'array è una stringa che rappresenta un argomento passato alla riga di comando.
- **envp** (environment pointer): È un array di stringhe che contiene le variabili d'ambiente. Ogni stringa nell'array rappresenta una variabile d'ambiente con la sua chiave e il suo valore.

```
.text:004011D0 argc            = dword ptr  8
.text:004011D0 argv            = dword ptr  0Ch
.text:004011D0 envp            = dword ptr  10h
```

VARIABILI

- **hModule**: È un puntatore a una variabile di tipo dword (32-bit) e sembra rappresentare un handle a un modulo. Questo è comunemente usato in Windows per rappresentare i moduli (eseguibili o DLL) caricati in un processo.
- **Data**: È un puntatore a una variabile di tipo byte. La notazione -118h suggerisce che questa variabile è situata a un offset negativo rispetto al punto di riferimento, il che implica che è una variabile locale all'interno dello stack della funzione.
- **var_117**: Anche questa è una variabile di tipo byte, situata a un offset negativo rispetto al punto di riferimento, simile a Data.
- **var_8**: È una variabile locale di tipo dword, usata probabilmente per scopo di calcolo o per memorizzare un valore temporaneo durante l'esecuzione della funzione.
- **var_4**: Un'altra variabile locale di tipo dword, simile a var_8, usata per scopi di calcolo o per conservare un valore temporaneo all'interno della funzione.

```
.text:004011D0 hModule          = dword ptr -11Ch
.text:004011D0 Data             = byte ptr -118h
.text:004011D0 var_117          = byte ptr -117h
.text:004011D0 var_8            = dword ptr -8
.text:004011D0 var_4            = dword ptr -4
```

SEZIONI

Utilizzando CFF Explorer analizziamo le sezioni che compongono il malware:

Name	Virtual Size	Virtual Address	Raw Size	Raw Address	Reloc Address	Linenumbers	Relocations N...	Linenumbers ...	Characteristics
00000250	00000258	0000025C	00000260	00000264	00000268	0000026C	00000270	00000272	00000274
Byte[8]	Dword	Dword	Dword	Dword	Dword	Dword	Word	Word	Dword
.text	00005646	00001000	00006000	00001000	00000000	00000000	0000	0000	60000020
.rdata	000009AE	00007000	00001000	00007000	00000000	00000000	0000	0000	40000040
.data	00003EA8	00008000	00003000	00008000	00000000	00000000	0000	0000	C0000040
.rsrc	00001A70	0000C000	00002000	0000B000	00000000	00000000	0000	0000	40000040

Abbiamo individuato le sezioni .text, .rdata, .data, .rsrc.
Sebbene queste sezioni siano presenti anche negli eseguibili legittimi, è sempre bene analizzarle poiché contengono le istruzioni che il programma dovrà eseguire. Di seguito vediamo lo scopo di ciascuna sezione nel dettaglio.

SEZIONI

.text

- Descrizione: Contiene il codice eseguibile del programma.
- Contenuto: Istruzioni macchina che il processore esegue.
- Caratteristiche: Tipicamente è una sezione di sola lettura ed eseguibile. È protetta per evitare che venga modificata durante l'esecuzione.

.rdata (Read-Only Data)

- Descrizione: Contiene dati di sola lettura.
- Contenuto: Può includere stringhe, costanti e puntatori a funzioni o dati.
- Caratteristiche: È una sezione di sola lettura, quindi i dati in essa contenuti non possono essere modificati a runtime.

SEZIONI

.data

- Descrizione: Contiene dati modificabili.
- Contenuto: Variabili globali e statiche che possono essere modificate durante l'esecuzione del programma.
- Caratteristiche: È una sezione di lettura e scrittura, permettendo la modifica dei dati a runtime.

.rsrc (Resource Data)

- Descrizione: Contiene le risorse del programma.
- Contenuto: Include risorse come icone, cursori, immagini, menu, dialoghi e stringhe utilizzate dal programma.
- Caratteristiche: È una sezione di sola lettura. Le risorse vengono utilizzate principalmente dall'interfaccia grafica del programma.

LIBRERIE

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

Sempre su CFF Explorer, ci spostiamo in “Import Directory“ per vedere quali librerie vengono importate dal malware. Ne abbiamo trovate 2: KERNEL32.dll e ADVAPI32.dll

Di seguito vediamo quali funzioni svolgono.

LIBRERIE

KERNEL32.dll: Questa libreria comprende le funzioni fondamentali per interagire con il sistema operativo, come la gestione della memoria e la manipolazione dei file.

ADVAPI32.dll: Questa libreria contiene le funzioni necessarie per l'interazione con il registro di sistema del sistema operativo.

Nelle slide successive analizziamo le funzioni contenute all’interno delle librerie per comprenderne meglio il funzionamento.

LIBRERIA KERNEL32.dll

All’interno di questa libreria troviamo 51 funzioni. Analizziamo le più importanti:

- ***VirtualAlloc***: alloca, riserva o impegna una regione di pagine nella memoria virtuale del processo chiamante. Un malware potrebbe usare questa funzione per allocare memoria per il codice iniettato o per i dati.
- ***WriteFile***: scrive dati in un file o in un dispositivo di input/output. Un malware potrebbe utilizzare questa funzione per scrivere dati rubati in un file o per modificare file di sistema.
- ***LoadLibraryA***: carica una libreria dinamica (DLL) nella memoria del processo chiamante. Un malware potrebbe usare questa funzione per caricare librerie aggiuntive contenenti codice malevolo.
- ***GetProcAddress***: recupera l'indirizzo di una funzione esportata da una DLL. Un malware può usare questa funzione per risolvere gli indirizzi delle funzioni che intende chiamare dinamicamente.
- ***CreateFileA***: crea o apre un file o un dispositivo I/O. Questa funzione può essere usata da un malware per creare, aprire o manipolare file di sistema o file di dati.

LIBRERIA KERNEL32.dll

- ***ReadFile***: legge dati da un file o da un dispositivo di input/output. Un malware può usare questa funzione per leggere dati da file di sistema o da file di configurazione.
- ***VirtualFree***: libera, decommette o rilascia una regione di pagine nella memoria virtuale del processo chiamante. Un malware può usare questa funzione per liberare la memoria allocata precedentemente con VirtualAlloc.
- ***GetVersion***: recupera il numero di versione del sistema operativo. Un malware può usare questa funzione per adattare il suo comportamento a diverse versioni di Windows.
- ***ExitProcess***: termina un processo e tutti i suoi thread. Un malware potrebbe usare questa funzione per terminare il proprio processo o quello di altri processi.
- ***HeapAlloc***: alloca un blocco di memoria da un heap. Utilizzata da un malware per allocare memoria necessaria per le sue operazioni.

LIBRERIA KERNEL32.dll

- ***HeapFree***: libera un blocco di memoria precedentemente allocato dall'heap. Un malware potrebbe usare questa funzione per gestire la memoria allocata.
- ***GetCurrentProcess***: recupera un pseudo-handle per il processo chiamante. Un malware può usare questa funzione per ottenere informazioni o manipolare il proprio processo.
- ***TerminateProcess***: termina un processo specificato e tutti i suoi thread. Un malware potrebbe usare questa funzione per terminare processi di sicurezza o antivirus.
- ***GetModuleHandleA***: recupera un handle per il modulo specificato caricato nel processo chiamante. Un malware può usare questa funzione per ottenere l'indirizzo base di una DLL caricata.
- ***SetFilePointer***: imposta la posizione del puntatore del file per un file aperto. Utilizzata per leggere o scrivere dati in posizioni specifiche di un file.

LIBRERIA ADVAPI32.dll

All’interno di quessta libreria sono presenti solo 2 funzioni:

- *RegSetValueExA*
- *RegCreateKeyExa*

RegSetValueExA e RegCreateKeyExA sono funzioni dell'API di Windows utilizzate per interagire con il registro di sistema. Queste funzioni sono utilizzate appunto per creare e impostare valori all'interno del registro di sistema di Windows.

Nell'ambito dei malware, le funzioni RegSetValueExA e RegCreateKeyExA possono essere utilizzate per diverse ragioni legate alla persistenza, alla configurazione del malware, o alla manipolazione delle impostazioni del sistema operativo per facilitarne l'attività.

IPOTESI

Il malware analizzato utilizza le librerie Kernel32.dll e Advapi32.dll, suggerendo una sofisticata operazione di manipolazione del sistema operativo Windows. Ecco alcune ipotesi sul suo funzionamento:

1. Persistenza nel Sistema: Il malware usa RegCreateKeyExA e RegSetValueExA per creare e modificare chiavi di registro, garantendo l'esecuzione automatica all'avvio del sistema. Questo permette al malware di rimanere attivo anche dopo un riavvio del computer.
2. Allocazione di Memoria: Con VirtualAlloc e HeapAlloc, il malware alloca memoria per caricare ed eseguire codice malevolo o per memorizzare dati temporanei necessari per le sue operazioni.
3. Manipolazione di File: Funzioni come CreateFileA, WriteFile, e ReadFile suggeriscono che il malware può creare, modificare, e leggere file. Questo potrebbe includere la creazione di file di log per attività malevole o la modifica di file di sistema per nascondere la propria presenza.
4. Caricamento Dinamico di Codice: L'uso di LoadLibraryA e GetProcAddress permette al malware di caricare DLL aggiuntive e risolvere funzioni specifiche in modo dinamico, estendendo così le sue capacità senza dover includere tutto il codice necessario all'interno del proprio eseguibile.

IPOTESI

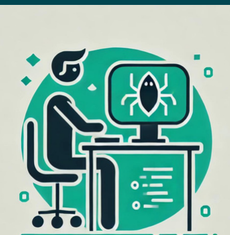
5. Terminazione di Processi: Con TerminateProcess, il malware potrebbe terminare processi di sicurezza o antivirus che ostacolano la sua esecuzione, aumentando così la sua resilienza contro la rilevazione e la rimozione.
6. Recupero di Informazioni di Sistema: Funzioni come GetVersion e GetCurrentProcess indicano che il malware raccoglie informazioni sull'ambiente di esecuzione per adattare il suo comportamento a diverse versioni di Windows e configurazioni di sistema.
7. Modifica del Registro di Sistema: Usando le funzioni di Advapi32.dll, il malware potrebbe alterare impostazioni di sicurezza, configurazioni di rete o altre chiavi critiche per facilitare ulteriori compromissioni del sistema.
8. Interazione con Altri Processi: Funzioni come ReadProcessMemory e WriteProcessMemory suggeriscono che il malware può interagire con la memoria di altri processi, consentendo operazioni di injection o estrazione di informazioni.

Queste ipotesi indicano un malware altamente capace, progettato per persistere, espandersi dinamicamente, e manipolare il sistema operativo e i dati dell'utente per raggiungere i suoi obiettivi malevoli.

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.



SCOPO DELLA FUNZIONE

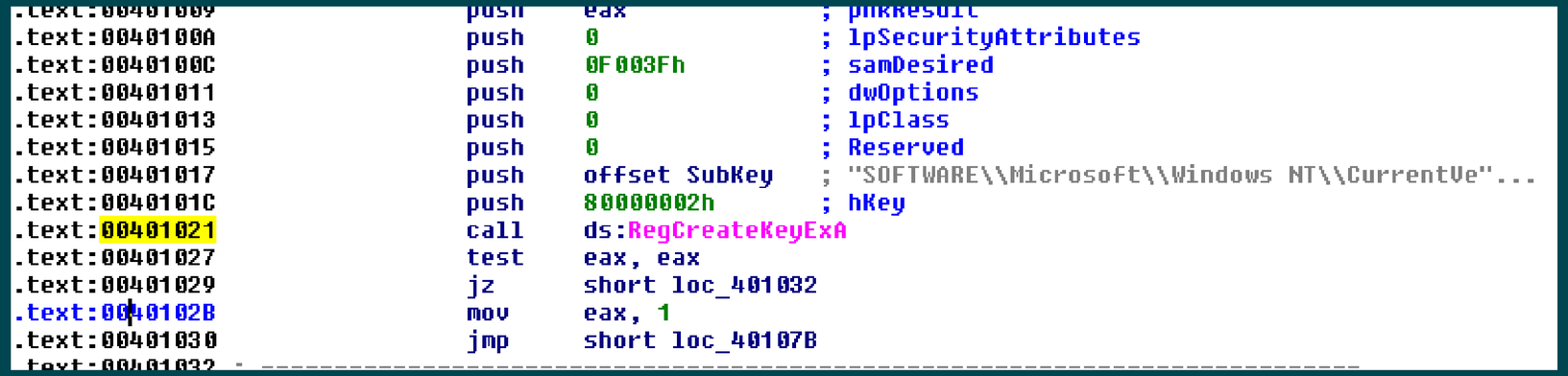
Osserviamo la funzione alla locazione di memoria **00401021**:

La funzione **RegCreateKeyExA** crea una nuova chiave di registro (altrimenti, se presente, la apre per effettuare un’eventuale manipolazione).

Nel contesto di un malware, la funzione può avere lo scopo di ottenere persistenza, ovvero essere avviata come processo legittimo all’avvio del sistema operativo.

PASSAGGIO PARAMETRI

Dalla seguente figura, possiamo vedere come i parametri di cui la funzione ha bisogno vengono passati tramite funzioni “**push**”.



Troviamo funzioni importanti durante la creazione di un nuovo file come **dwoptions** e **lpSecurityAttributes**. Da notare come quest’ultima sia stata impostata a 0, il che significa che la funzione avrà i privilegi di amministratore.

INDIRIZZO CHIAVE DI REGISTRO

All’indirizzo di memoria **00401017** troviamo il seguente parametro:

L’oggetto che si trova alla posizione di memoria 00401017 è:
“SOFTWARE\Microsoft\Windows NT\CurrentVersione\Winlogon“.

Si tratta dell‘indirizzo alla chiave di registro che il malware sta tentando di modificare. Questo parametro verrà passato alla funzione "**RegCreateKeyExA**"(utilizzato per ottenere persistenza) tramite un comando push.

ISTRUZIONI AGGIUNTIVE

Istruzione all'indirizzo **00401027**:Questa istruzione esegue un'operazione AND logica tra il registro eax e se stesso. Lo scopo di questa operazione è di aggiornare i flag del processore basati sul valore di eax, senza modificare il valore di eax.

Istruzione all'indirizzo **00401029:** Questa istruzione è una condizione di salto (Jump if Zero). Salta all'etichetta loc_401032 se il flag Zero (ZF) è impostato a 1. Questo significa che il salto verrà eseguito se il valore di eax era zero quando è stata eseguita l'istruzione test eax, eax.

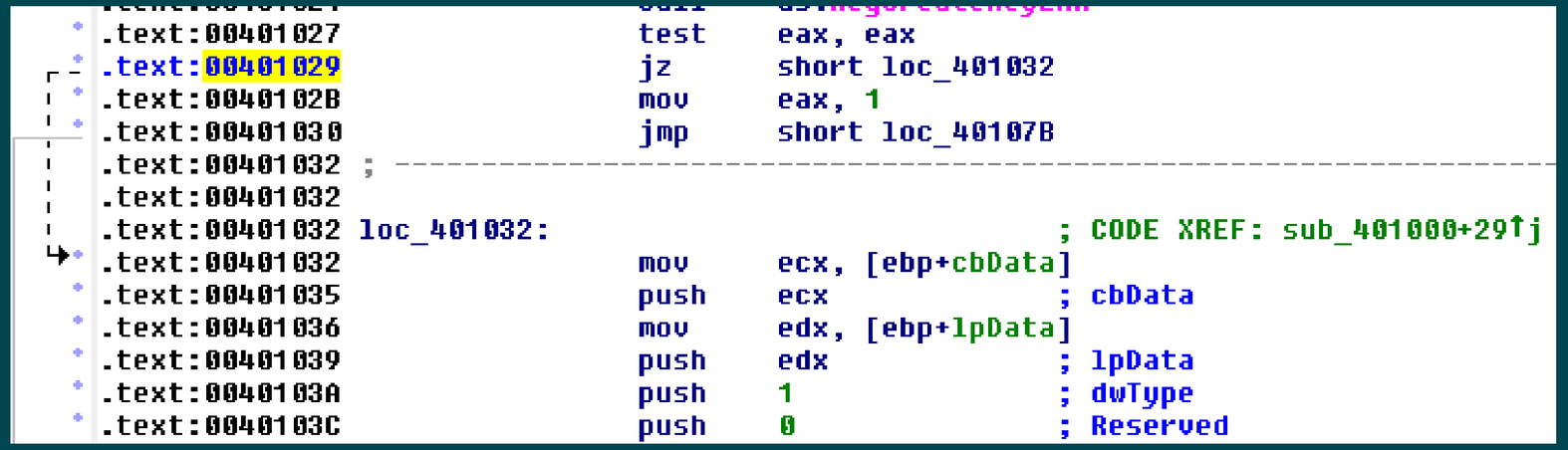
ISTRUZIONI AGGIUNTIVE

TEST: Questa operazione è comunemente utilizzata per verificare se un registro contiene zero senza modificare il suo contenuto. Nel nostro caso, Il registro EAX viene confrontato con se stesso attraverso un'operazione di AND. Il risultato non viene memorizzato, ma gli effetti dell'operazione sono riscontrabili sui flag dell'architettura (EFLAGS). Il flag ZF verrà impostato a 1 se il risultato della comparazione restituirà zero; altrimenti, ZF avrà valore 0 se il risultato della comparazione sarà diverso da zero.

JZ: L'istruzione jz (Jump if Zero) è una delle istruzioni di salto condizionato disponibili nell'assembly x86. Questa istruzione salta a una determinata etichetta (label) se il flag Zero (ZF) è impostato. Come suggerisce lo stesso nome, questa funzione effettua un salto se il parametro che gli viene passato (EAX) risulterà uguale a 0.

TRADUZIONE IN LINGUAGGIO C

IDA ci suggerisce il comportamento delle funzioni tramite una freccia.



Possiamo tradurre le istruzioni agli indirizzi di memoria 00401027 e 00401029 in linguaggio C come:

```
if (eax == 0) {
  // Salta alla locazione 401032 se la chiamata a RegCreateKeyExA() è riuscita
 } else { eax = 1; // Imposta il valore di eax a 1 se la chiamata non è riuscita
 }
```

ISTRUZIONI AGGIUNTIVE

In poche parole, se il registro **EAX** è uguale a **zero** (come verificato dall'istruzione test), viene effettuato un salto alla locazione **401032** (come specificato dall'istruzione **jZ**). Altrimenti, il codice continua nell'istruzione successiva.

VALUE NAME

All’indirizzo di memoria **00401047** abbiamo una chiamata di funzione di tipo “**RegSetValueExA**”.

```
.text:00401032 loc_401032:                             ; CODE XREF: sub_401000+29↑j
.text:00401032                 mov     ecx, [ebp+cbData]
.text:00401035                 push    ecx             ; cbData
.text:00401036                 mov     edx, [ebp+lpData]
.text:00401039                 push    edx             ; lpData
.text:0040103A                 push    1               ; dwType
.text:0040103C                 push    0               ; Reserved
.text:0040103E                 push    offset ValueName ; "GinaDLL"
.text:00401043                 mov     eax, [ebp+hObject]
.text:00401046                 push    eax             ; hKey
.text:00401047                 call    ds:RegSetValueExA
.text:0040104D                 test    eax, eax
.text:0040104F                 jz      short loc_401062
.text:00401051                 mov     ecx, [ebp+hObject]
.text:00401054                 push    ecx             ; hObject
.text:00401055                 call    ds:CloseHandle
.text:0040105B                 mov     eax, 1
```

Il valore che viene “pushato” alla funzione è la stringa ”**GinaDLL**” che punta (grazie all’offset Value name) ad un registro leggittimo di windows che gestisce il **login** utente.

GINA DLL

GINA.DLL è responsabile per la presentazione della schermata di login, la raccolta delle credenziali dell'utente e l'interazione con l'utente durante il processo di autenticazione.

Nel contesto di un malware, quest’ultimo può sostituire la **GINA.DLL** di default con una versione **modificata** che intercetta le credenziali dell'utente (nome utente e password) al momento del login. Queste credenziali possono poi essere inviate a un server controllato dagli attaccanti.

Si tratta, quindi di un modo per ottenere le credenziali di accesso dell‘utente vittima.

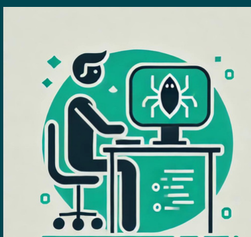
FUNZIONALITA’ IMPLEMENTATE

Riassumendo, possiamo dire che il malware sta cercando di inserire una propria DLL di monitoraggio nel sistema.

In questo modo tenta di ottenere credenziali, eseguire un codice malevolo al momento del login, oppure alterare altri comportamenti di autenticazione di Windows.

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 il 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? (dal giorno 1 in pratica) 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



TOOLS

Per completare l’esercizio di oggi utilizzeremo:

OllyDBG 

OllyDbg è un debugger utilizzato per il reverse engineering, noto per la sua capacità di visualizzare dettagliate informazioni del codice, facilitando l'identificazione di vulnerabilità e bug nel software.

CFF Explorer 

Strumento che permette di visualizzare e modificare file eseguibili Windows, come EXE e DLL. È utilizzato per esaminare le strutture dei file, analizzare le dipendenze e ottimizzare le prestazioni dei binari.

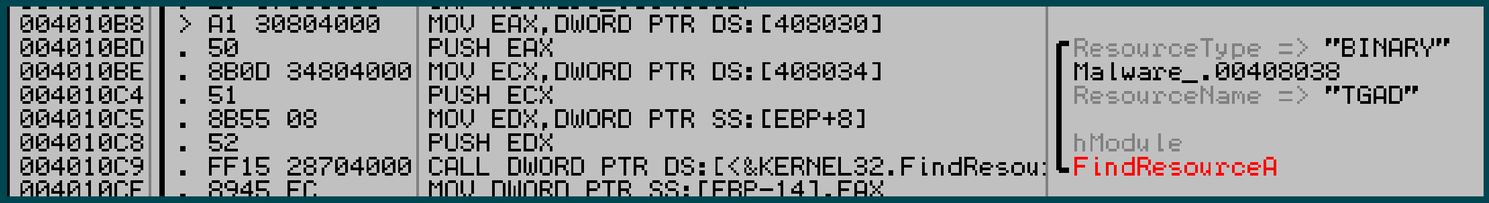
RESOURCE NAME

Per prima cosa, apriamo Olly DBG ed analizziamo il codice tra le righe 00401080 e 00401128.

```
00401080 $ 55              PUSH EBP
00401081 . 8BEC            MOV EBP,ESP
00401083 . 83EC 18         SUB ESP,18
00401086 . 56              PUSH ESI
00401087 . 57              PUSH EDI
00401088 . C745 EC 00000(  MOV DWORD PTR SS:[EBP-14],0
0040108F . C745 E8 00000(  MOV DWORD PTR SS:[EBP-18],0
00401096 . C745 F8 00000(  MOV DWORD PTR SS:[EBP-8],0
0040109D . C745 F0 00000(  MOV DWORD PTR SS:[EBP-10],0
004010A4 . C745 F4 00000(  MOV DWORD PTR SS:[EBP-C],0
004010AB . 837D 08 00      CMP DWORD PTR SS:[EBP+8],0
004010AF .v75 07           JNZ SHORT Malware_.004010B8
004010B1 . 33C0            XOR EAX,EAX
004010B3 .vE9 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
004010CF . 8945 EC         MOV DWORD PTR SS:[EBP-14],EAX
004010D2 . 837D EC 00      CMP DWORD PTR SS:[EBP-14],0
004010D6 .v75 07           JNZ SHORT Malware_.004010DF
004010D8 . 33C0            XOR EAX,EAX
004010DA .vE9 E0000000     JMP Malware_.004011BF
004010DF > 8B45 EC         MOV EAX,DWORD PTR SS:[EBP-14]
004010E2 . 50              PUSH EAX                          hResource
004010E3 . 8B4D 08         MOV ECX,DWORD PTR SS:[EBP+8]
004010E6 . 51              PUSH ECX                          hModule
004010E7 . FF15 14704000   CALL DWORD PTR DS:[<&KERNEL32.LoadResou  LoadResource
004010ED . 8945 E8         MOV DWORD PTR SS:[EBP-18],EAX
004010F0 . 837D E8 00      CMP DWORD PTR SS:[EBP-18],0
004010F4 .v75 05           JNZ SHORT Malware_.004010FB
004010F6 .vE9 AA000000     JMP Malware_.004011A5
004010FB > 8B55 E8         MOV EDX,DWORD PTR SS:[EBP-18]
004010FE . 52              PUSH EDX                          hResource
004010FF . FF15 10704000   CALL DWORD PTR DS:[<&KERNEL32.LockResou  LockResource
00401105 . 8945 F8         MOV DWORD PTR SS:[EBP-8],EAX
00401108 . 837D F8 00      CMP DWORD PTR SS:[EBP-8],0
0040110C .v75 05           JNZ SHORT Malware_.00401113
0040110E .vE9 92000000     JMP Malware_.004011A5
00401113 > 8B45 EC         MOV EAX,DWORD PTR SS:[EBP-14]
00401116 . 50              PUSH EAX                          hResource
00401117 . 8B4D 08         MOV ECX,DWORD PTR SS:[EBP+8]
0040111A . 51              PUSH ECX                          hModule
0040111B . FF15 0C704000   CALL DWORD PTR DS:[<&KERNEL32.SizeofRes  SizeofResource
00401121 . 8945 F0         MOV DWORD PTR SS:[EBP-10],EAX
00401124 . 837D F0 00      CMP DWORD PTR SS:[EBP-10],0
00401128 .v77 02           JA SHORT Malware_.0040112C
```

RESOURCE NAME

Il valore del parametro Resource Name passato alla funzione FindResourceA() è contenuto all'interno della sezione nella figura. Come possiamo vedere il suo valore è TGAD.



FUNZIONALITA’

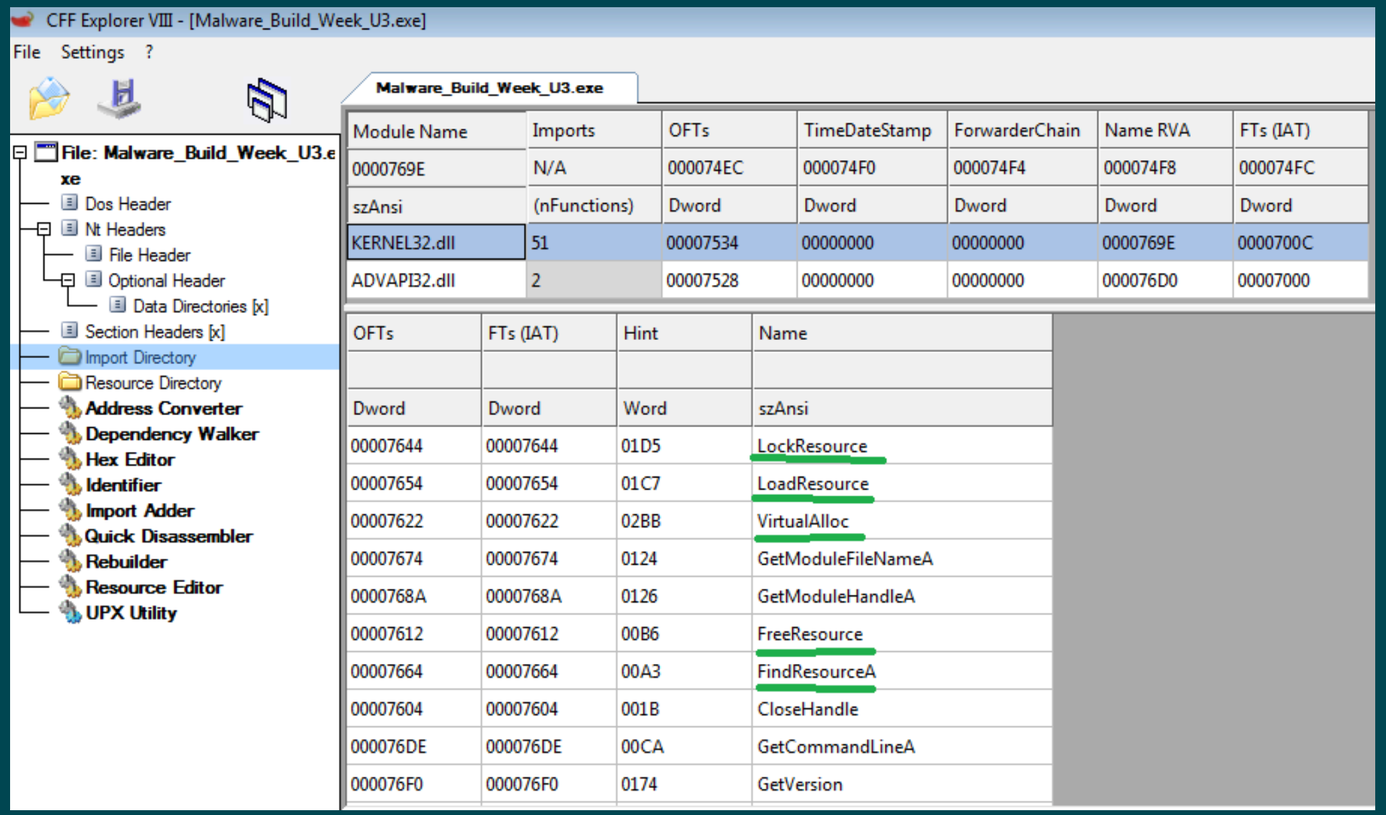
Osservando le chiamate di funzione effettuate dal malware, possiamo intuire quali funzionalità verranno implementate:

- **Find Resource**: Trova una risorsa inclusa in un modulo
- **Load Resource**: Carica la risorsa in memoria
- **Lock Resource**: Punta ad una risorsa in memoria
- **Size of Resource**: restituisce la dimensione della risorsa caricata
- **Virtual Alloc**: Riserva un determinata area della memoria per il processo caricato

FUNZIONALITA’

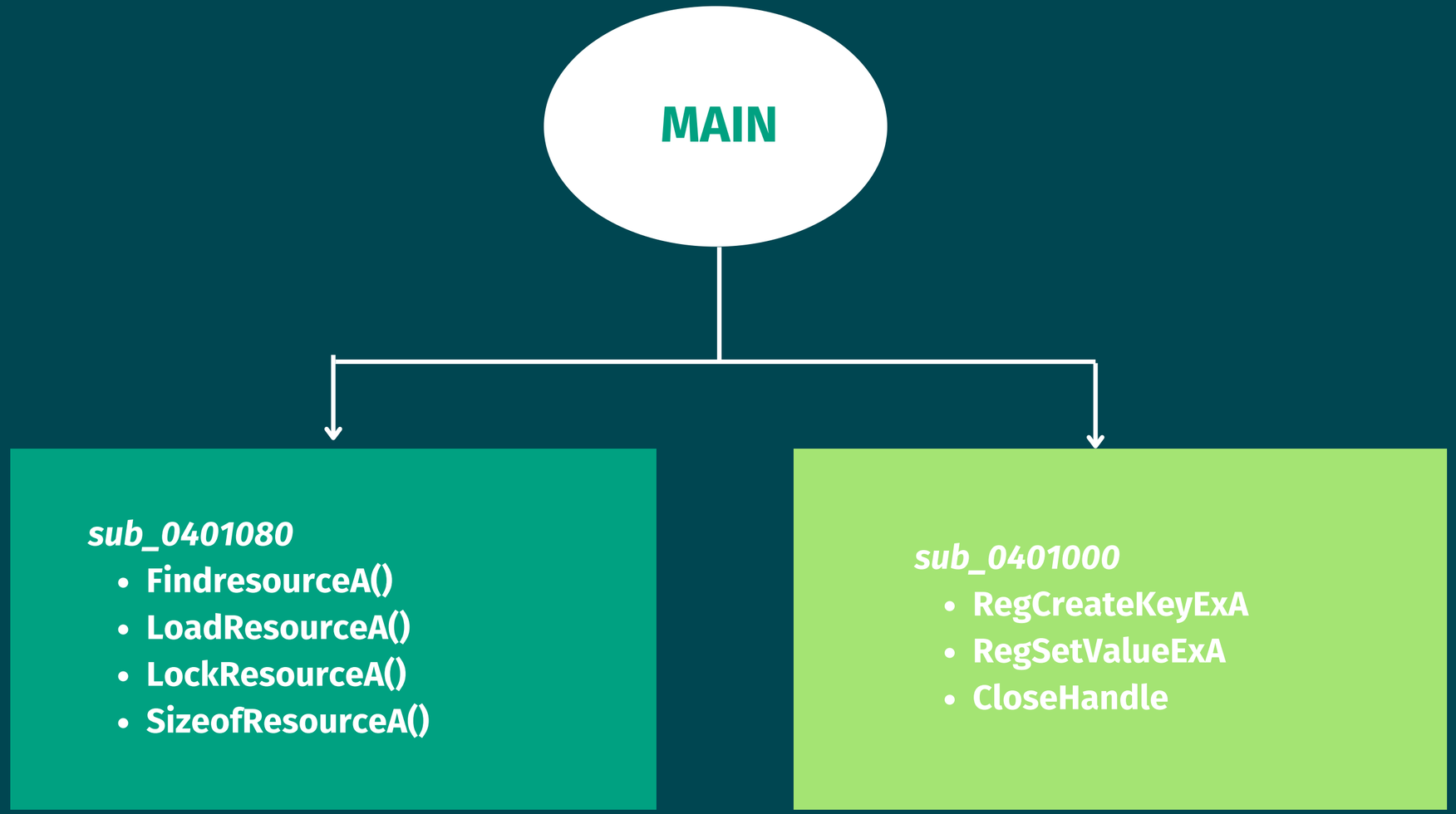
Il malware esegue una sequenza operativa tipica dei **dropper**, utilizzando le chiamate API di Windows: FindResourceA, LoadResource, e SizeOfResource. La funzione FindResourceA localizza una risorsa incorporata nell'eseguibile, in questo caso identificata dal nome "TGAD". Questo indica che il malware cerca un componente specifico occultato, che potrebbe essere un secondo stadio di un attacco o un payload aggiuntivo. Successivamente, LoadResource carica il contenuto della risorsa in memoria, pratica comune dei dropper per eseguire o manipolare componenti dannosi. SizeOfResource restituisce la dimensione della risorsa, garantendo una corretta gestione della memoria durante l'estrazione e l'eventuale esecuzione del payload. Queste operazioni di ricerca, caricamento e dimensionamento di componenti nascosti sono utilizzate per eludere la rilevazione basata su firma, poiché il payload non appare come un file separato sul disco ma viene gestito interamente in memoria. L'identificazione di una risorsa con un nome come "TGAD" potrebbe implicare l'uso di tecniche di steganografia o compressione per mascherare la natura della risorsa. In sintesi, il malware implementa funzionalità di estrazione di risorse nascoste, allocazione di memoria e manipolazione di dati, e potenzialmente manipolazione di file, indicando un intento di distribuire componenti dannosi nascosti e garantire persistenza sul sistema compromesso.

FUNZIONALITA’



E’ effettivamente possibile determinare le funzionalità precedentemente citate con l’analisi statica basica. Utilizzando CFF Explorer e spostandoci nella sezione **Import Directory** analizziamo le librerie importate. Al loro interno, possiamo notare la presenza delle funzioni in questione.

DIAGRAMMA DI FLUSSO



GIORNO 4

Preparate l’ambiente ed i tool per l’esecuzione del Malware (suggerimento: avviate principalmente 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. 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. 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 .

PROCESS MONITOR

Process Monitor (spesso abbreviato come ProcMon) è uno strumento avanzato per il monitoraggio del sistema operativo Windows, particolarmente utile nell'analisi dinamica dei malware, poiché consente agli analisti di osservare in tempo reale il comportamento dei programmi malevoli nel sistema. Process Monitor traccia tutte le operazioni sui file, le **interazioni con il Registro di sistema**, e le attività dei processi e dei thread, fornendo una visualizzazione dettagliata di tutte le azioni effettuate dal malware. Include anche filtri avanzati per isolare attività sospette, log dettagliati per analisi successive e visualizzazione delle stack trace per comprendere il flusso di esecuzione. Dopo aver configurato un ambiente isolato (sandbox), possiamo eseguire il malware in sicurezza e osservare la sua attività in tempo reale. Possiamo anche utilizzare dei filtri per focalizzarci sulle operazioni rilevanti. Ad esempio, possiamo filtrare le attività di scrittura sui file e le modifiche al registro per individuare rapidamente le azioni dannose.

PROCESS MONITOR

Prima di avviare il malware, settiamo in modo sicuro l'ambiente di lavoro; per farlo eseguiamo le seguenti operazioni:

Utilizzo Rete interna: per evitare che il malware possa comunicare con l'esterno, caricando file malevoli o trasmettere informazioni sulla macchina colpita

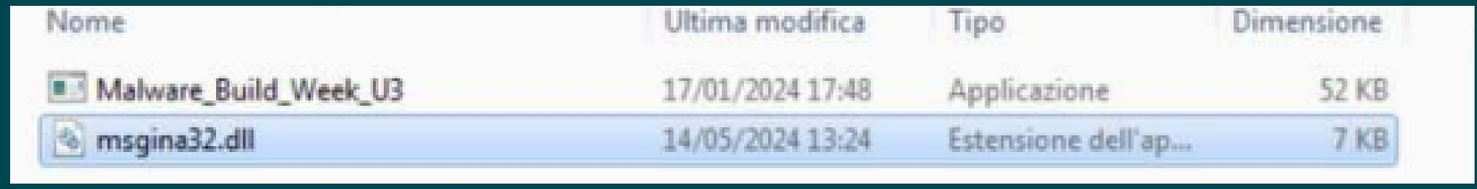
Disabilitazione porte USB: spesso un malware è in grado di rilevare dispositivi di archiviazione esterna ed utilizzarli come vettore per propagarsi su altri dispositivi

Disabilitazione cartelle condivise tra VM e PC Host: spesso ce ne dimentichiamo, ma la presenza di queste cartelle permette al malware di agire anche al di fuori della macchina virtuale ed infettare la macchina Host.

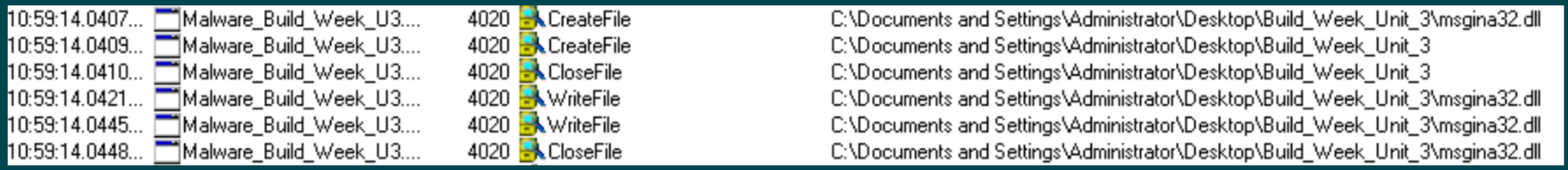
Creazione di istantanee: La creazione di istantanee ci permette non solo di capire quali parti del sistema operativo vengono compromesse(attraverso un confronto tra il prima e il dopo l'esecuzione del malware), ma anche di riportare la macchina ad uno stato precedente qualora vi fossero compromissioni importanti.

Possiamo, dunque, eseguire il malware e monitorare i processi ad esso associati tramite il tool Procmon.

PROCESS MONITOR



Nome	Ultima modifica	Tipo	Dimensione
Malware_Build_Week_U3	17/01/2024 17:48	Applicazione	52 KB
msgina32.dll	14/05/2024 13:24	Estensione dell'ap...	7 KB



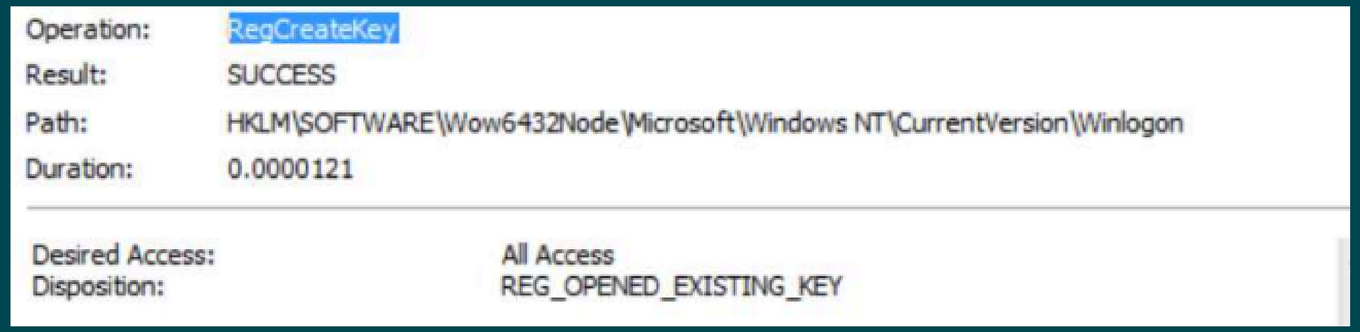
10:59:14.0407...	Malware_Build_Week_U3....	4020	CreateFile	C:\Documents and Settings\Administrator\Desktop\Build_Week_Unit_3\msgina32.dll
10:59:14.0409...	Malware_Build_Week_U3....	4020	CreateFile	C:\Documents and Settings\Administrator\Desktop\Build_Week_Unit_3
10:59:14.0410...	Malware_Build_Week_U3....	4020	CloseFile	C:\Documents and Settings\Administrator\Desktop\Build_Week_Unit_3
10:59:14.0421...	Malware_Build_Week_U3....	4020	WriteFile	C:\Documents and Settings\Administrator\Desktop\Build_Week_Unit_3\msgina32.dll
10:59:14.0445...	Malware_Build_Week_U3....	4020	WriteFile	C:\Documents and Settings\Administrator\Desktop\Build_Week_Unit_3\msgina32.dll
10:59:14.0448...	Malware_Build_Week_U3....	4020	CloseFile	C:\Documents and Settings\Administrator\Desktop\Build_Week_Unit_3\msgina32.dll

La prima cosa che possiamo notare, dopo l'avvio del maware, è la comparsa di un nuovo file all'interno della stessa cartella del malware. Si tratta di un file chiamato **msgina.dll**; la cosa ci insospettisce poichè il nome è identico ad una delle librerie di windows(responsabile dell'autenticazione dell'user).

La presenza di questo file può spesso indicare che un eseguibile sta tentando di sostituire una propria libreria con una libreria leggittima di windows.

PROCESS MONITOR

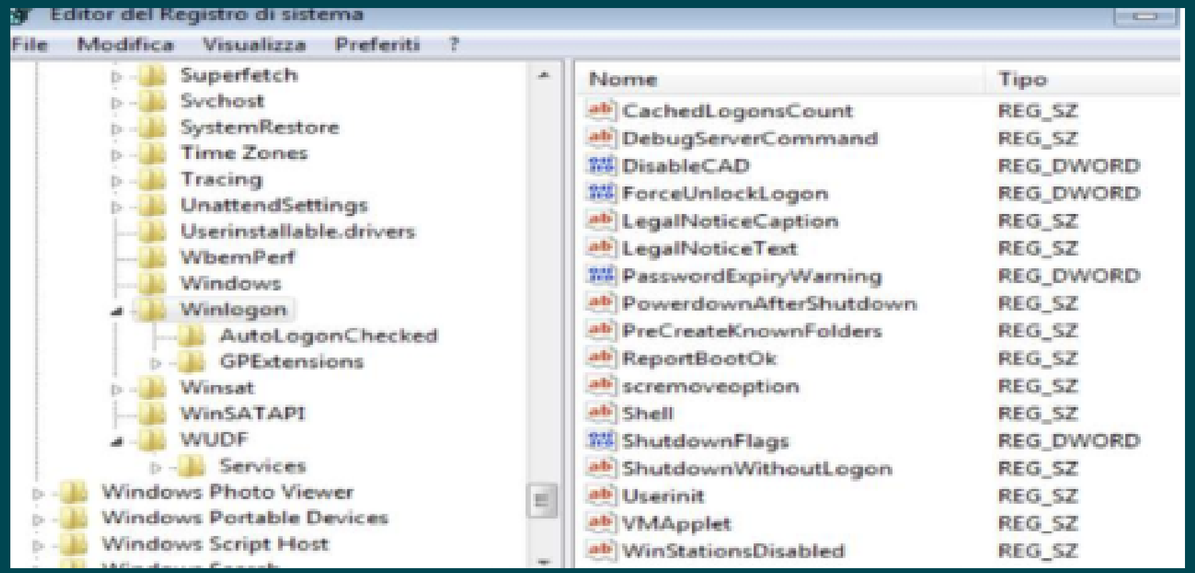
Chiave di registro



Nell’operazione RegCreatekey, possiamo vedere che la chiave di registro creata è REG_OPENED_EXISTING_KEY.

PROCESS MONITOR

Chiave di registro

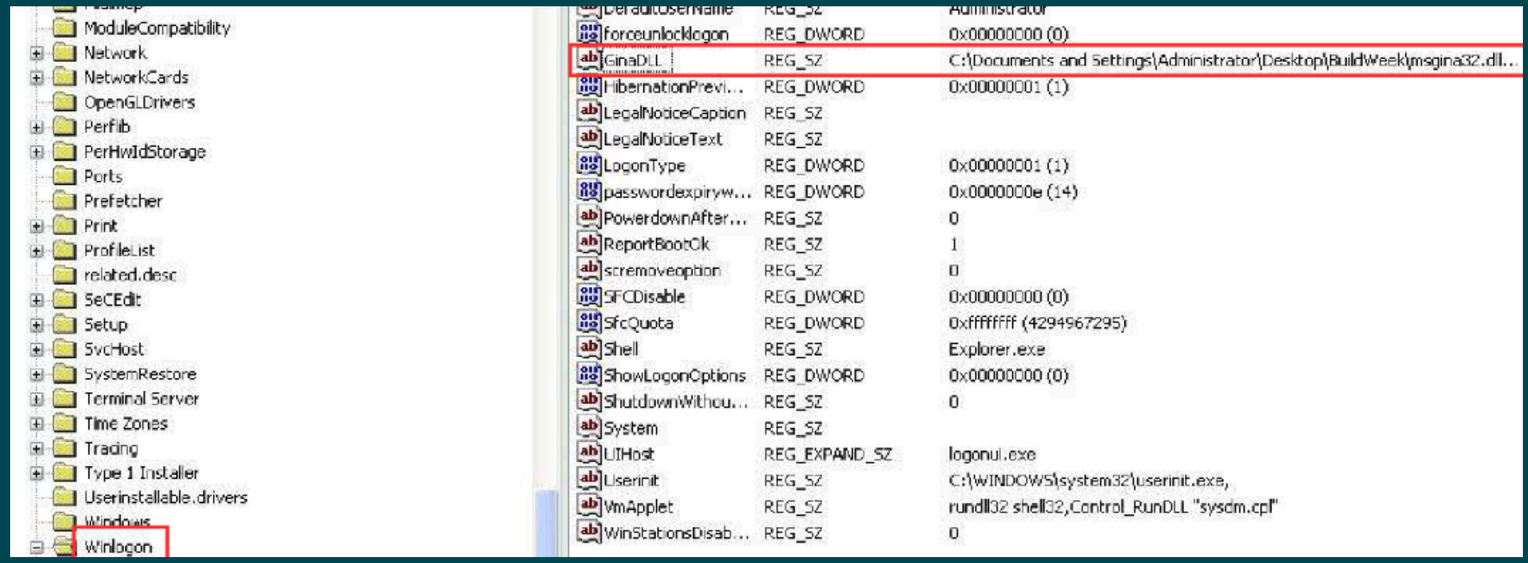


Utilizzando il tool regedit proviamo a cercare il valore della chiave di registro. Come possiamo vedere, non troviamo GinaDLL poichè con l'introduzione di Windows Vista e versioni successive, incluso Windows 7, Microsoft ha sostituito la Gina con un nuovo modello di autenticazione.

PROCESS MONITOR

Chiave di registro

Proviamo dunque a testare il malware su Windows XP. Come si può osservare, è stata correttamente creata la chiave di registro "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\GinaDLL". All'interno di questa chiave, è stato copiato il percorso del DLL msgina32.dll, situato nella stessa directory dell'eseguibile del malware, in questo caso "C:\Documents and Settings\Administrator\Desktop\Buildweek\msgina32.dll". Questo consente di inserire un DLL malevolo che cattura le credenziali di accesso, le quali possono essere poi sfruttate da un utente malintenzionato per prendere il controllo del computer.

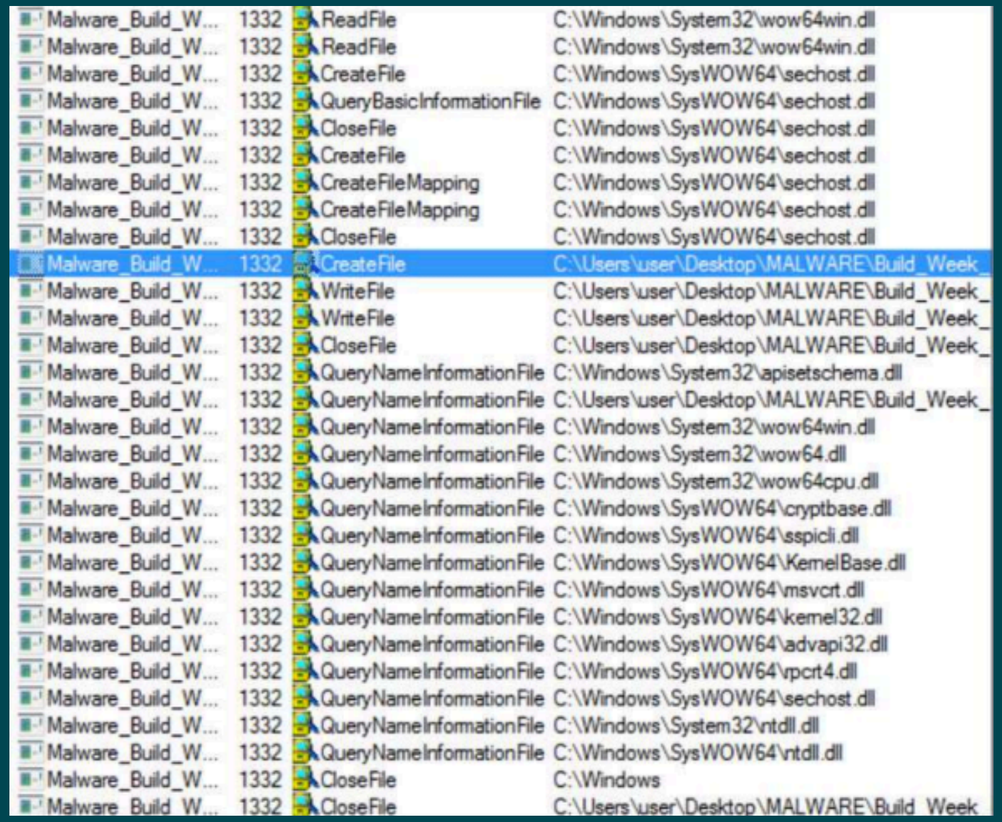


PROCESS MONITOR

File System

Il contenuto della cartella in cui si trova il malware viene modificato attraverso la chiamata di sistema **CreateFile** (come si vede in figura). Attraverso questo comando avviene la creazione del file **msgina.dll** (file malevolo).

E’ presente anche la chiamata **CloseFile**; quest’ultima, nel contesto di un malware, serve a chiudere specifici processi leggittimi oppure a chiudere il processo malevolo, quando questo ha finito di operare, per rendersi invisibile.



Malware_Build_W...	1332	ReadFile	C:\Windows\System32\wow64win.dll
Malware_Build_W...	1332	ReadFile	C:\Windows\System32\wow64win.dll
Malware_Build_W...	1332	CreateFile	C:\Windows\SysWOW64\sechost.dll
Malware_Build_W...	1332	QueryBasicInformationFile	C:\Windows\SysWOW64\sechost.dll
Malware_Build_W...	1332	CloseFile	C:\Windows\SysWOW64\sechost.dll
Malware_Build_W...	1332	CreateFile	C:\Windows\SysWOW64\sechost.dll
Malware_Build_W...	1332	CreateFileMapping	C:\Windows\SysWOW64\sechost.dll
Malware_Build_W...	1332	CreateFileMapping	C:\Windows\SysWOW64\sechost.dll
Malware_Build_W...	1332	CloseFile	C:\Windows\SysWOW64\sechost.dll
Malware_Build_W...	1332	CreateFile	C:\Users\user\Desktop\MALWARE\Build_Week
Malware_Build_W...	1332	WriteFile	C:\Users\user\Desktop\MALWARE\Build_Week_
Malware_Build_W...	1332	WriteFile	C:\Users\user\Desktop\MALWARE\Build_Week_
Malware_Build_W...	1332	CloseFile	C:\Users\user\Desktop\MALWARE\Build_Week_
Malware_Build_W...	1332	QueryNameInformationFile	C:\Windows\System32\apisetschema.dll
Malware_Build_W...	1332	QueryNameInformationFile	C:\Users\user\Desktop\MALWARE\Build_Week_
Malware_Build_W...	1332	QueryNameInformationFile	C:\Windows\System32\wow64win.dll
Malware_Build_W...	1332	QueryNameInformationFile	C:\Windows\System32\wow64.dll
Malware_Build_W...	1332	QueryNameInformationFile	C:\Windows\System32\wow64cpu.dll
Malware_Build_W...	1332	QueryNameInformationFile	C:\Windows\SysWOW64\cryptbase.dll
Malware_Build_W...	1332	QueryNameInformationFile	C:\Windows\SysWOW64\sspicli.dll
Malware_Build_W...	1332	QueryNameInformationFile	C:\Windows\SysWOW64\KernelBase.dll
Malware_Build_W...	1332	QueryNameInformationFile	C:\Windows\SysWOW64\msvcrt.dll
Malware_Build_W...	1332	QueryNameInformationFile	C:\Windows\SysWOW64\kernel32.dll
Malware_Build_W...	1332	QueryNameInformationFile	C:\Windows\SysWOW64\advapi32.dll
Malware_Build_W...	1332	QueryNameInformationFile	C:\Windows\SysWOW64\rpcrt4.dll
Malware_Build_W...	1332	QueryNameInformationFile	C:\Windows\SysWOW64\sechost.dll
Malware_Build_W...	1332	QueryNameInformationFile	C:\Windows\System32\ntdll.dll
Malware_Build_W...	1332	QueryNameInformationFile	C:\Windows\SysWOW64\ntdll.dll
Malware_Build_W...	1332	CloseFile	C:\Windows
Malware_Build_W...	1332	CloseFile	C:\Users\user\Desktop\MALWARE\Build_Week

FUNZIONAMENTO DEL MALWARE

Il malware descritto è un dropper, un tipo di malware progettato per introdurre e installare ulteriori payload malevoli nel sistema della vittima. In particolare, utilizza la chiave di registro GinaDLL per inserire una DLL malevola che cattura le credenziali di accesso degli utenti. L'inserimento di una DLL malevola tramite la chiave GinaDLL compromette le credenziali di login degli utenti, esponendole a potenziali furti. Una volta che un attaccante avrà accesso alle credenziali, potrà accedere al sistema con i privilegi dell'utente compromesso, potenzialmente eseguendo ulteriori attacchi o installando un altro malware. Inoltre, modificando la chiave di registro per caricare la propria DLL, il malware garantisce la persistenza nel sistema, poiché la DLL verrà caricata ad ogni login dell'utente.

Questa tecnica rende difficile la rimozione del malware, poiché anche dopo un riavvio del sistema, il malware rimane attivo. Tuttavia, il malware è progettato per colpire sistemi operativi obsoleti (Windows XP e precedenti), suggerendo che potrebbe essere stato sviluppato in un periodo in cui questi sistemi erano più diffusi o che gli attaccanti stiano prendendo di mira sistemi legacy ancora in uso. Per proteggersi da questo tipo di attacco, è fondamentale aggiornare i sistemi operativi a versioni più recenti e mantenere aggiornati gli strumenti di sicurezza. Inoltre, monitorare e analizzare le modifiche alle chiavi di registro critiche può aiutare a rilevare e prevenire tali attacchi.

GIORNO 5

GINA (Graphical identification and authentication) è un componente lecito di Windows che permette l’autenticazione degli utenti tramite interfaccia grafica 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.

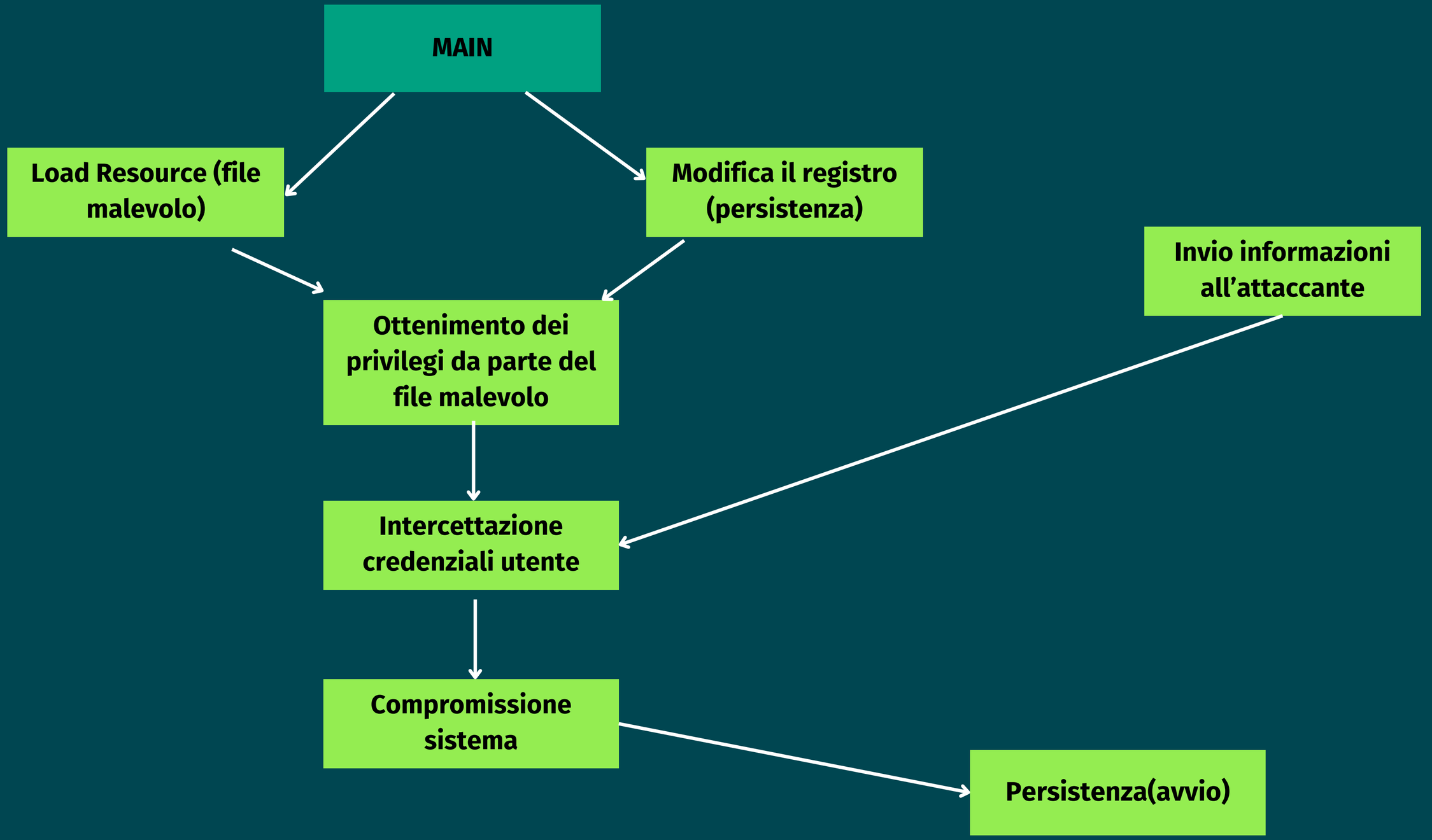
- Cosa può succedere se il file . dll lecito viene sostituito con un file . dll malevolo, che intercetta i dati inseriti?

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

La sostituzione della libreria “gina.dll” leggittima con quella malevola può avere diverse conseguenze negative:

- **Furto di credenziali**: Un programma di login malevolo può registrare e inviare le credenziali (username e password) dell'utente a un attaccante. Questo permette all'attaccante di ottenere accesso non autorizzato al sistema e ai dati personali o aziendali.
- **Accesso non autorizzato**: Con le credenziali rubate, un attaccante può accedere a risorse sensibili sul computer, come file, database, e-mail, e altre informazioni riservate. Questo può portare a violazioni della privacy e alla compromissione di informazioni confidenziali.
- **Installazione di ulteriori malware**: Un programma di login malevolo può fungere da porta di ingresso per ulteriori malware, come trojan, ransomware, o keylogger, che possono causare danni aggiuntivi al sistema e ai dati.
- **Escalation dei privilegi:** Il malware può tentare di ottenere privilegi di amministratore per eseguire operazioni dannose con maggiore autorità, compromettendo ulteriormente la sicurezza del sistema.

- **Manomissione del sistema:** Il programma di login malevolo può alterare o eliminare file di sistema critici, rendendo il sistema instabile o inutilizzabile. Questo può causare downtime e perdita di produttività.
- **Sorveglianza e spionaggio**: Un login malevolo può includere funzionalità di sorveglianza, registrando le attività dell'utente, inclusi i dati sensibili e le comunicazioni private.


MAIN
Load Resource (file malevolo)
Modifica il registro (persistenza)
Invio informazioni all’attaccante
Ottenimento dei privilegi da parte del file malevolo
Intercettazione credenziali utente
Compromissione sistema
Persistenza(avvio)

CONSIGLI

Di seguito, alcuni consigli per prevenire il diffondersi di file malevoli come quello appena visto:

- **Usare Software Antivirus e Antimalware**: Installare e mantenere aggiornato un software antivirus e antimalware affidabile. Questi strumenti possono rilevare e bloccare dropper e altri tipi di malware prima che possano infettare il sistema.

- **Mantenere il Sistema e le Applicazioni Aggiornate**: Assicurarsi che il sistema operativo, i browser e tutte le applicazioni siano sempre aggiornati con le ultime patch di sicurezza. I dropper spesso sfruttano vulnerabilità note nei software per infiltrarsi nei sistemi.

- **Implementare l'Autenticazione a Due Fattori (2FA)**: Abilitare l'autenticazione a due fattori per tutti gli account importanti. Anche se un dropper riesce a ottenere le credenziali, 2FA aggiunge un ulteriore livello di sicurezza che rende più difficile per gli attaccanti accedere al sistema.

CONSIGLI

- **Eseguire Backup Regolari**: eseguire questa operazione su supporti esterni o cloud sicuri. In caso di infezione, è possibile ripristinare i propri dati senza dover pagare un riscatto o perdere informazioni importanti.

- **Evitare di Aprire Allegati Sospetti e Link Non Verificati**: Fare sempre attenzione agli allegati e ai link presenti in e-mail o messaggi provenienti da mittenti sconosciuti o sospetti. I dropper sono spesso distribuiti tramite phishing e allegati e-mail infetti.

- **Limitare i Privilegi degli Utenti:** Configurare i sistemi in modo che gli utenti abbiano solo i privilegi necessari per svolgere il loro lavoro. Limitare i privilegi può impedire ai dropper di eseguire modifiche critiche al sistema se riescono a infettare un account con privilegi limitati.

GRAZIE PER L'ATTENZIONE