EPICODE

Progetto Finale

TRACCIA

Malware Analysis

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

Analisi statica

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

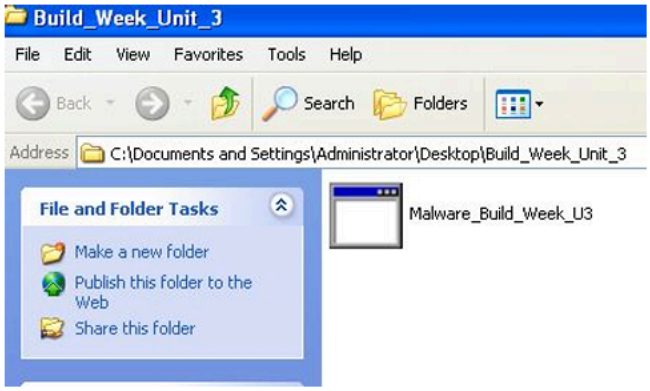
- Quanti parametri sono passati alla funzione Main()?
- Quante variabili sono dichiarate all'interno della funzione Main()?
- Quali sezioni sono presenti all'interno del file eseguibile? Descrivete brevemente almeno 2 di quelle identificate
- Quali librerie importa il Malware? Per ognuna delle librerie 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.

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 **00401027** e **00401029**.
- 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»?

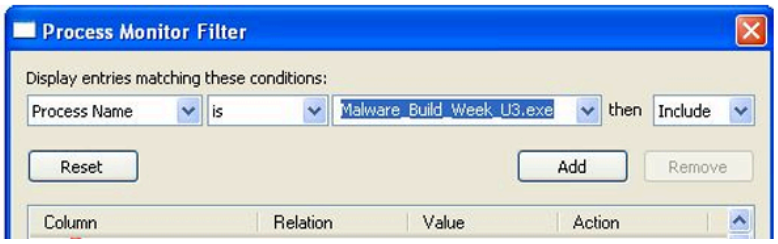
Analisi dinamica

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.

INTRODUZIONE

Analisi statica e dinamica dei malware: una panoramica completa

Entrando nel mondo dell'analisi dei malware, ci imbattiamo in due approcci fondamentali:

Analisi statica: Come un detective che esamina la scena del crimine, l'analisi statica osserva il malware "a riposo", senza eseguirlo. Utilizza tecniche e strumenti per carpire il comportamento del software malevolo senza attivarlo, fornendo informazioni preziose sulla sua natura e potenziali minacce.

Analisi dinamica: In questo scenario, il malware viene "eseguito" in un ambiente controllato, simile a un'operazione sotto copertura. Osservando il comportamento del malware in azione, possiamo scoprire come interagisce con il sistema, quali dati modifica e quali azioni intraprende.

Entrambe le tecniche, statica e dinamica, si suddividono in due livelli:

Analisi basica: Il primo passo in entrambe le metodologie. Nell'analisi statica basica, si esamina un file eseguibile senza approfondire le singole istruzioni. L'obiettivo è determinare se il file è malevolo e ricavare informazioni generiche sulle sue funzionalità. L'analisi dinamica basica, invece, esegue il malware in un ambiente sandbox per osservarne il comportamento e tentare di neutralizzarlo.

Analisi avanzata: Tuffandosi nel "codice sorgente" del malware, l'analisi statica avanzata utilizza tecniche di reverse engineering per identificare il comportamento del malware analizzando le sue istruzioni. I disassembler, strumenti che convertono il file eseguibile in linguaggio assembly, sono alleati preziosi in questa fase. L'analisi dinamica avanzata, invece, impiega debugger per monitorare lo stato del programma durante l'esecuzione, raccogliendo informazioni dettagliate sulle sue azioni e interazioni con il sistema.

Combinando i risultati dell'analisi statica e dinamica, gli analisti ottengono una visione completa e accurata delle minacce poste dal malware. L'analisi statica fornisce un'analisi preliminare rapida e identifica potenziali segnali di pericolo, mentre l'analisi dinamica conferma e approfondisce tali informazioni osservando il malware in azione. Insieme, queste due tecniche sono cruciali per la lotta efficace contro i malware sofisticati in continua evoluzione.

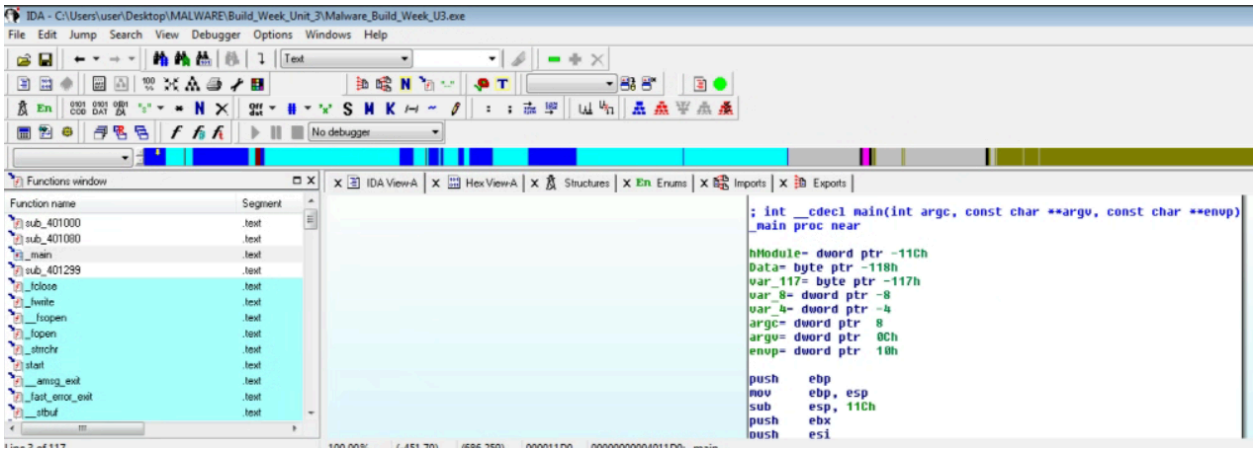
Oltre alla suddivisione in livelli basico e avanzato, è importante sottolineare che l'analisi statica e dinamica sono complementari. I risultati dell'una possono guidare e perfezionare l'altra, offrendo una comprensione più completa del malware e delle sue insidie.

Per padroneggiare a fondo l'analisi dei malware, è essenziale acquisire familiarità con i concetti di reverse engineering, linguaggio assembly e debugger. Questi strumenti saranno la vostra chiave per svelare i segreti del malware e neutralizzare le sue minacce.

PROCEDIMENTO

Analisi statica

Utilizzando IDA Pro, possiamo notare che i parametri nella funzione main() sono tre, un int e due char, e il numero delle variabili dichiarate sono cinque



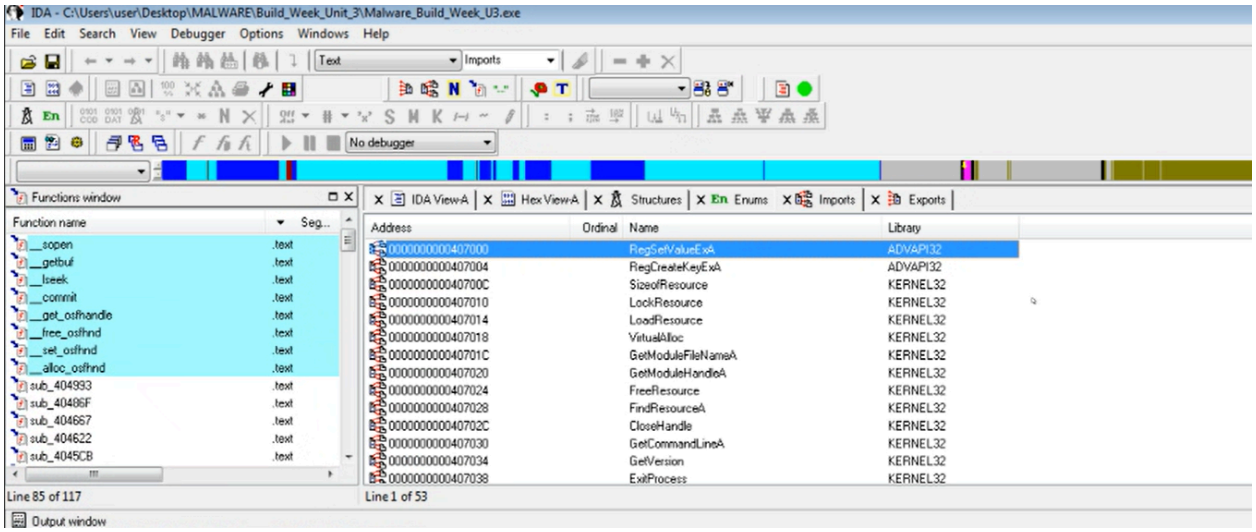
Di seguito ci sono le sezioni di un file eseguibile, i punti che ho voluto approfondire sono:

- **.text** è una sezione fondamentale di un file eseguibile PE di Windows. Contiene il codice macchina effettivo che viene eseguito dal processore quando il programma viene avviato. La sezione .text può essere utilizzata per comprendere il funzionamento di un programma e per identificare potenziali vulnerabilità di sicurezza.
- **.rdata** è una sezione importante del file eseguibile che contiene dati in sola lettura utilizzati dal programma durante l'esecuzione. I dati nella sezione ".rdata" possono includere costanti di programma, tabelle di dati, risorse del programma e informazioni sul debug.

Malware_Build_Week_U3.exe

Name	Virtual Size	Virtual Address	Raw Size	Raw Address	Reloc Address	Linenumbers	Relocations N...	Linenumbers ...	Characteristics
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

Le librerie importate dal malware sono la KERNEL32 e ADVAPI32;

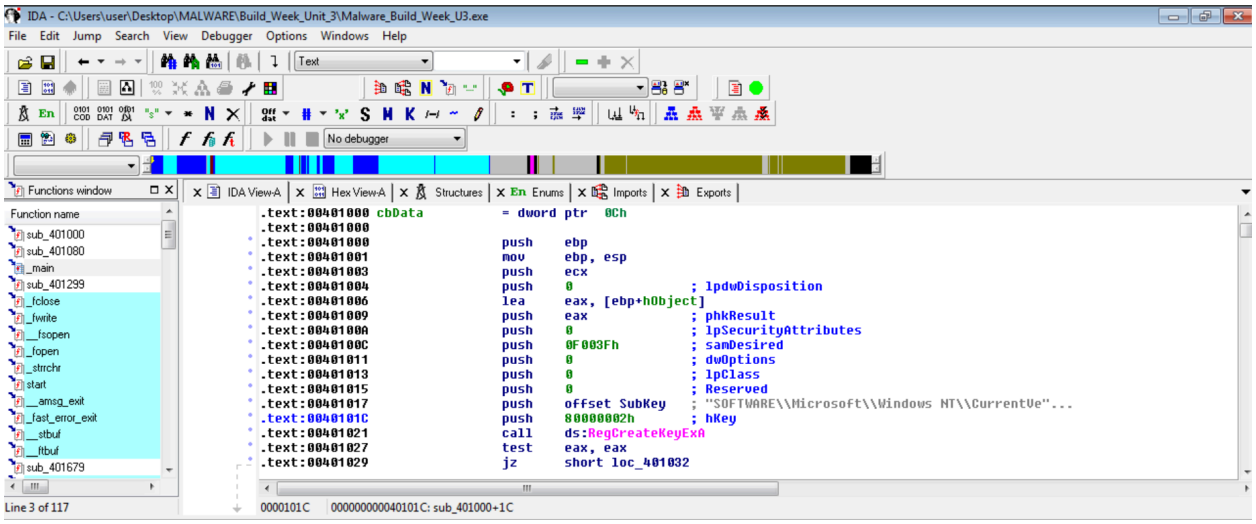


in base alle funzioni richiamate all'interno delle librerie posso ipotizzare che si tratta di un **dropper**, ovvero un programma malevolo che al suo interno contiene un malware.
I dropper rappresentano una tipologia di malware con caratteristiche ben precise che li distinguono da altri tipi di minacce informatiche. La loro funzione primaria è quella di distribuire e attivare altri malware all'interno di un sistema infetto. Per compiere questo compito, i dropper sfruttano diverse tecniche, tra cui l'utilizzo di specifiche API per estrarre il malware contenuto al loro interno.

Le API chiave per l'estrazione del malware:

- *FindResource():* Questa funzione permette di identificare la risorsa contenente il malware all'interno del file eseguibile del dropper.
- *LoadResource():* Una volta individuata la risorsa, LoadResource() la carica in memoria, rendendola disponibile per l'estrazione.
- *LockResource():* Questa funzione blocca la risorsa in memoria, impedendo che venga modificata o sovrascritta durante il processo di estrazione.
- *SizeOfResource():* Determina la dimensione della risorsa contenente il malware, garantendo che venga estratta la quantità corretta di dati.

La funzione chiamata all'indirizzo di memoria 00401021 sembra essere responsabile della *creazione di una chiave di registro,* e utilizza la funzione **RegCreateKeyExA** della libreria **ADVAPI32.DLL**.



I parametri passati dalla funzione sono:

- hKEY
- SubKey
- dwFlags (dwOptions)
- lpSecurityAttributes
- Reserved
- lpdwDisposition

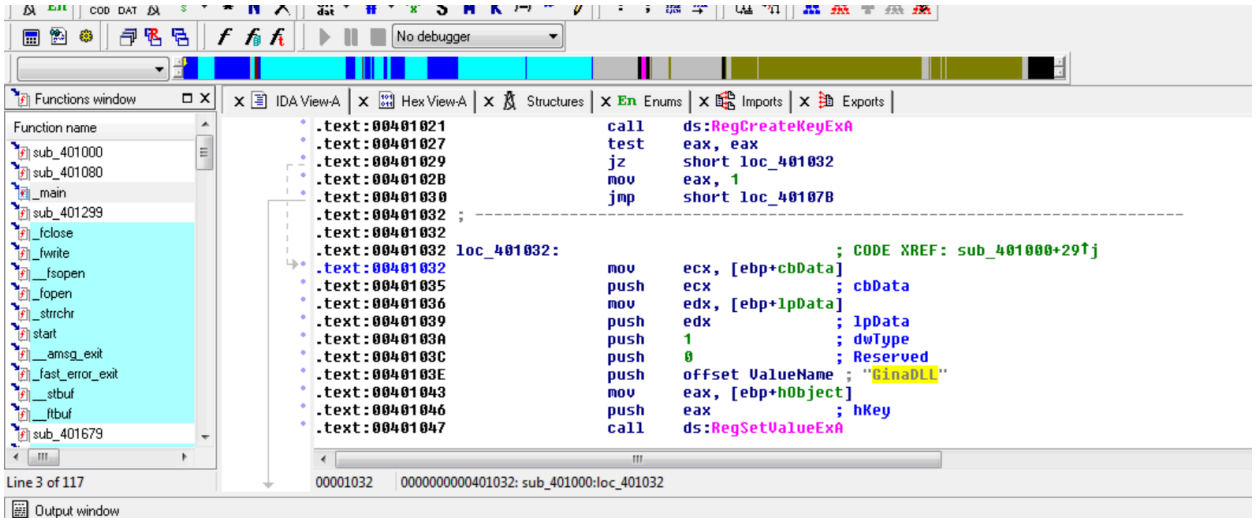
L'oggetto rappresentato dal parametro alla locazione di memoria 00401017 è molto probabilmente una stringa. Questa stringa rappresenta con tutta probabilità il percorso della chiave di registro che la funzione alla locazione 00401021 sta tentando di creare, ed è probabilmente memorizzata in memoria come stringa null-terminata, ovvero termina con un carattere nullo (\0).

Le istruzioni comprese tra gli indirizzi 00401027 e 00401029 controllano il valore di ritorno della chiamata alla funzione RegCreateKeyExA e saltano a una routine di gestione degli errori se la chiamata ha fallito:

- 00401027: Questa istruzione esegue un confronto tra il valore contenuto nel registro EAX e il valore zero.
- 00401029: Questa istruzione è un'istruzione condizionale di salto.
- Istruzione successiva: Se il valore in EAX è uguale a zero, il programma salterà all'istruzione successiva situata a otto byte di distanza. Se il valore in EAX non è uguale a zero, il programma continuerà a eseguire l'istruzione successiva a questa istruzione.

Traduzione in C:

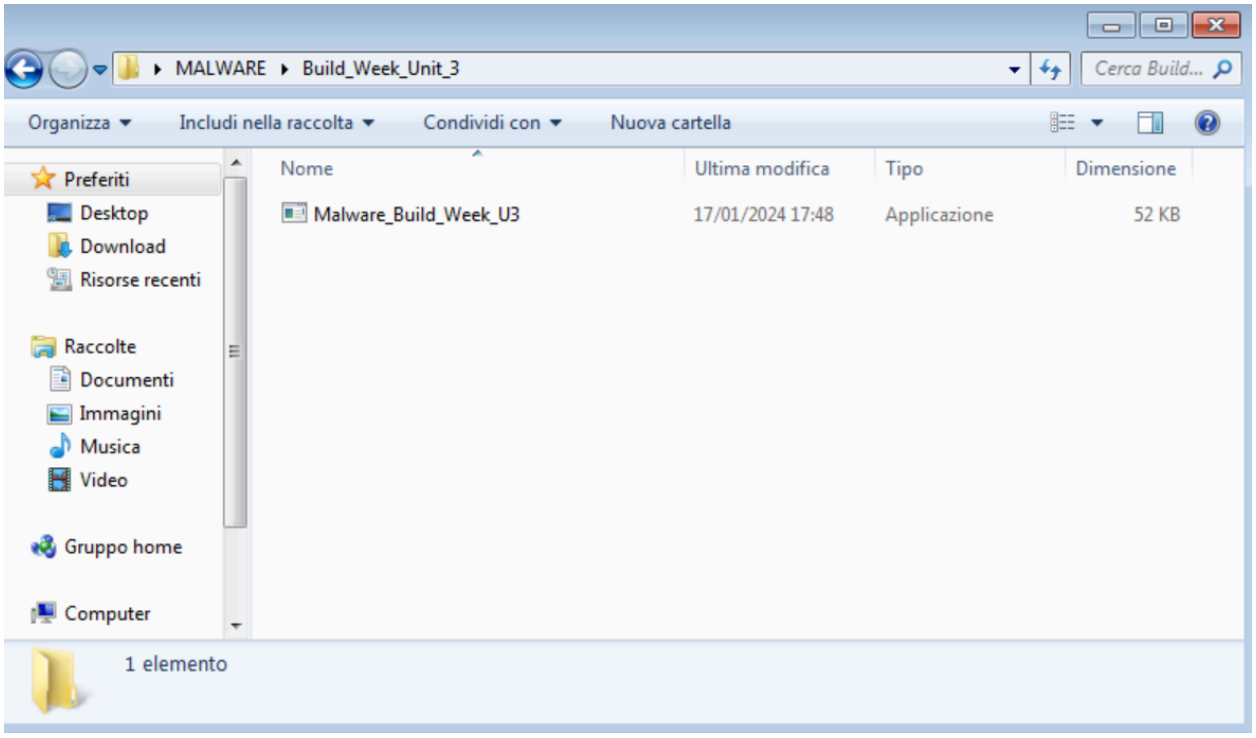
```
if (EAX == 0) {

  // Fai una cosa

} else {

  // Fai un'altra cosa

}
```



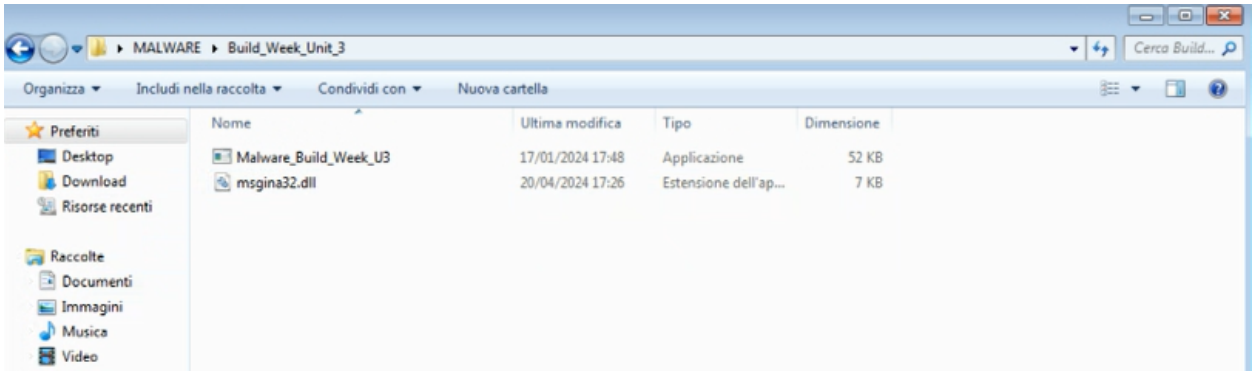
Il valore del parametro "ValueName" per la chiamata alla locazione 00401047 è "GinaDLL". Questo significa che la funzione sta impostando il valore di una chiave di registro denominata "GinaDLL" su un valore stringa. Il valore della stringa è il percorso del modulo "GinaDLL", che è un componente del sistema operativo Windows.

Analisi dinamica

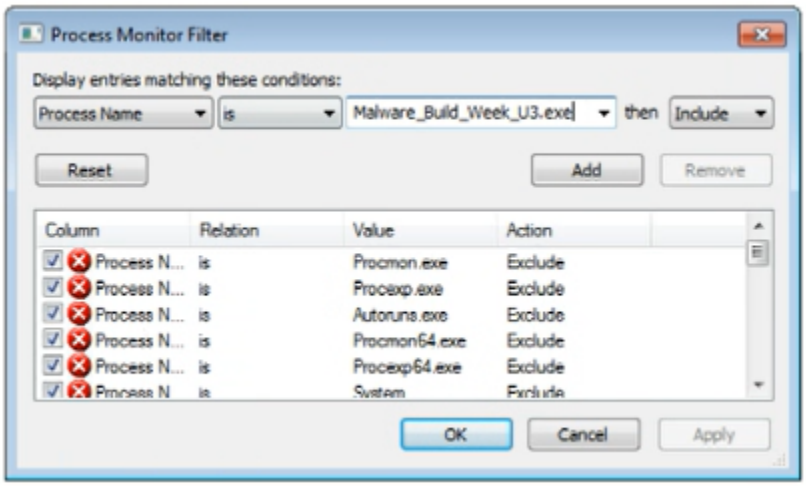
Prima di procedere questo è il contenuto della cartella del malware:



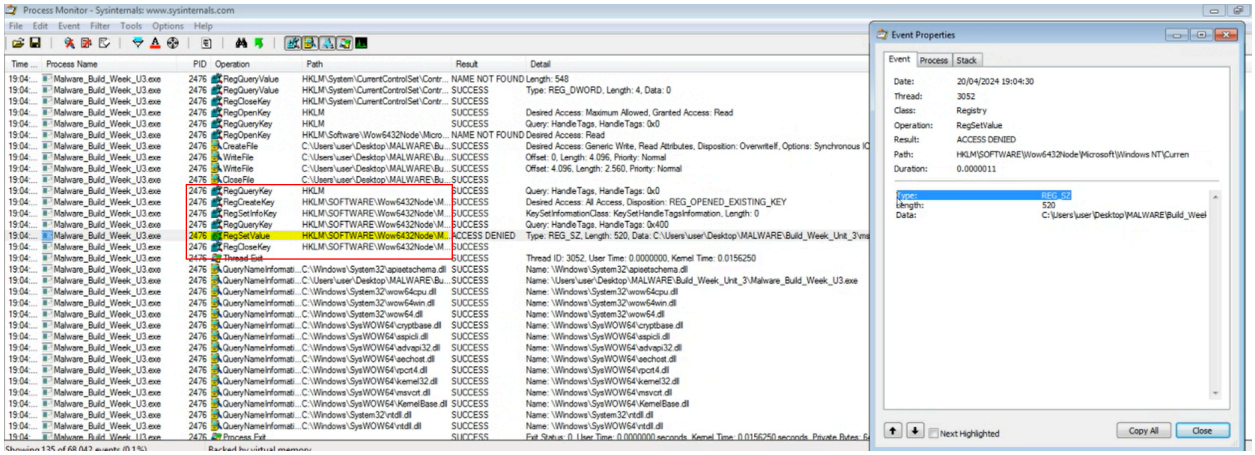
Dopo avere fatto il doppio click è apparsa l'estensione dell'app *msgina32.dll,* questo file è un componente critico del processo di accesso a Windows.



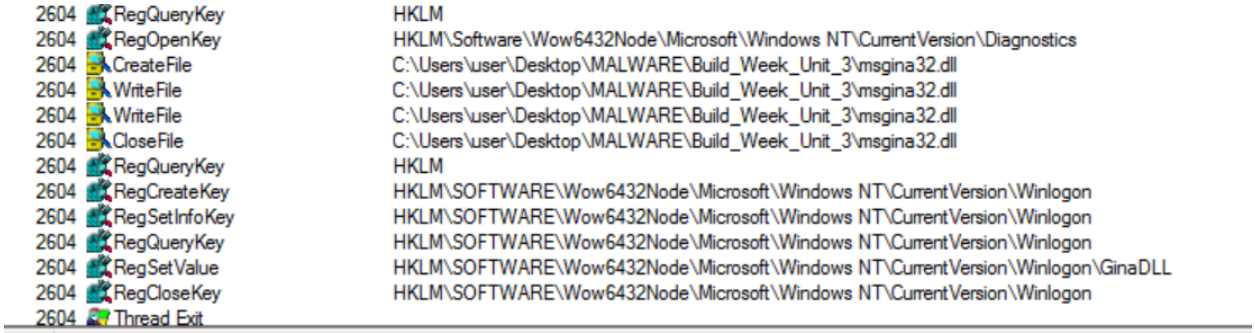
Avviamo Process Monitor e andiamo a filtrare:



possiamo notare che viene creata la chiave di registro HKLM, abbreviazione di HKEY_LOCAL_MACHINE, è una delle chiavi principali del registro di sistema di Windows. Contiene informazioni di configurazione cruciali per il funzionamento del sistema operativo e dei programmi installati. A differenza di altre chiavi del registro di sistema che riguardano profili utente specifici, HKLM memorizza impostazioni globali accessibili a tutti gli utenti del computer.



Viene fatta una chiamata di sistema (CreateFile) che crea la msgina.dll nella cartella del malware e a seguire vediamo la write file che inserisce il contenuto malevolo e poi la close file



Conclusione

Il comportamento globale del malware è quello della creazione e apertura della chiave di registro ***Winlogon*** con inserimento nella stessa il valore che punta alla dll malevola creata dopo la sua esecuzione. Essendo la dll in questione fondamentale per l'autenticazione degli utenti su windows, possiamo presupporre che lo scopo del software malevolo sia quello di registrare le autenticazioni da parte degli utenti al sistema per poi impossessarsene.