Report progetto

W24-D5

Redatto da Andrea Sciattella

04/08/2024

INDICE

1. TRACCIA

ll 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:

1. Quanti parametri sono passati alla funzione Main()?
2. Quante variabili sono dichiarate all'interno della funzione Main()?
3. Quali sezioni sono presenti all'interno del file eseguibile? Descrivete brevemente almeno due di quelle identificate.
4. 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:

5. Lo scopo della funzione chiamata alla locazione di memoria 00401021;
6. Come vengono passati i parametri alla funzione alla locazione 00401021;
7. Che oggetto rappresenta il parametro alla locazione 00401017
8. Il significato delle istruzioni comprese tra gli indirizzi 00401027 e 00401029.
9. Con riferimento all'ultimo quesito, tradurre il codice Assembly nel corrispondente costrutto C.
10. 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.

11. Quale chiave di registro viene creata?
12. Quale valore viene associato alla chiave di registro creata?

Passate ora alla visualizzazione dell'attività sul file system.

13. 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.

2. INTRODUZIONE

La crescente complessità e diffusione dei malware rappresenta una sfida significativa per il campo della Cyber Security. Per contrastare efficacemente queste minacce, è fondamentale comprendere a fondo il funzionamento dei malware attraverso un processo sistematico di analisi.

Questo progetto si propone di eseguire un'analisi completa di un campione di malware presente sulla nostra macchina preparata all'apposita "vivisezione", applicando metodologie di **analisi statica e dinamica**:

- L'analisi statica implica **l'esame del malware senza eseguirlo**, attraverso lo studio del codice sorgente, delle librerie importate e delle strutture interne del file eseguibile. Questo approccio ci permette di identificare le **caratteristiche e le funzionalità del malware**, come le sezioni del file, le funzioni chiamate e i parametri utilizzati, senza rischiare di attivare il codice dannoso.
- L'analisi dinamica, invece, consiste nell'osservare il **comportamento del malware in un ambiente controllato** durante la sua esecuzione. Utilizzando strumenti di monitoraggio, possiamo tracciare le **modifiche apportate al sistema**, come la creazione di file, la modifica di chiavi di registro e l'attività di rete. Questo ci fornisce una visione pratica e concreta degli effetti del malware sul sistema ospite.

Combinando queste due tecniche, saremo in grado di ottenere una comprensione completa del malware, identificando non solo le sue caratteristiche interne, ma anche il suo impatto operativo.

Questo progetto fornirà un quadro dettagliato del malware in esame, contribuendo alla mia capacità di sviluppare contromisure efficaci e migliorare le pratiche di isolamento e rimedio.

3. ANALISI STATICA

Per questa prima fase di analisi statica, useremo uno dei *disassembler* (programma che traduce da linguaggio macchina ad assembly) più potenti e versatili per l'analisi di malware: ***IDA Pro Free***.

Questo software è ampiamente utilizzato da ricercatori di sicurezza, analisti di malware e sviluppatori per **esaminare e comprendere il funzionamento interno** di file binari ed eseguibili senza eseguirli, creando così un ambiente di analisi in totale sicurezza.

Rispetto ai suoi competitor IDA Pro Free offre diversi vantaggi, tra cui l'identificazione di:

- Funzioni / chiamate di funzione;
- Analisi dello stack;
- Variabili locali e parametri.

3.1 Quanti parametri sono passati alla funzione Main()?

Nello screenshot di IDA sono evidenziati i parametri passati alla funzione Main().

Si possono definire parametri per due motivi:

1. I tre valori sono dichiarati all'interno della funzione
2. L'offset del valore è positivo rispetto al suo EBP (Base Pointer)

Nel nostro caso ne abbiamo tre che sono **argc**, **argv** ed **envp**.

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

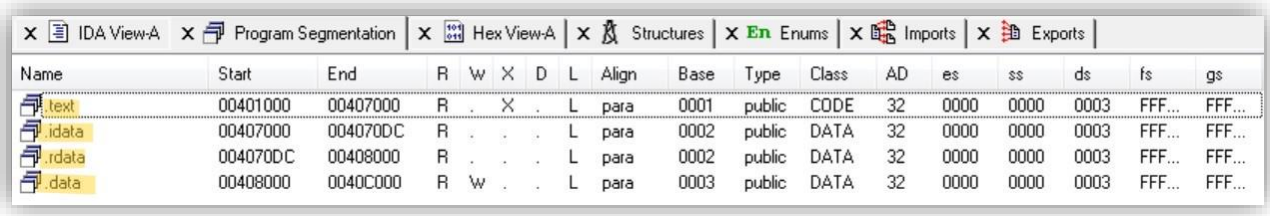
```
; int __cdecl main(int argc, const char **argv, const char **envp)
_main proc near

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
```

Nel caso delle variabili invece, il valore dell'offset rispetto al EBP (Base Pointer) è sempre positivo, quindi ci sono 5 variabili: **hModule, Data, var_117, var_8, var_4**.

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

Per visualizzare le sezioni presenti in un file su IDA Pro Free seguiamo il percorso della scheda View ->Open subviews ->Segments: sono presenti **.text**, **.idata**, **.rdata**, **.data**.



Name	Start	End	R	W	X	D	L	Align	Base	Type	Class	AD	es	ss	ds	fs	gs
.text	00401000	00407000	R	.	X	.	L	para	0001	public	CODE	32	0000	0000	0003	FFF...	FFF...
.idata	00407000	004070DC	R	.	.	.	L	para	0002	public	DATA	32	0000	0000	0003	FFF...	FFF...
.rdata	004070DC	00408000	R	.	.	.	L	para	0002	public	DATA	32	0000	0000	0003	FFF...	FFF...
.data	00408000	0040C000	R	W	.	.	L	para	0003	public	DATA	32	0000	0000	0003	FFF...	FFF...

- **Sezione .text**: La sezione .text contiene il codice eseguibile del programma, funzioni, procedure, e qualsiasi altra istruzione eseguibile. Memorizza le istruzioni macchina del programma che vengono eseguite dalla CPU ed è solitamente marcata come read-only ed eseguibile.
- **Sezione .data**: La sezione .data contiene dati globali e statici che possono essere modificati durante l'esecuzione del programma (Come variabili globali inizializzate, array, strutture e qualsiasi dato che il programma può modificare durante il suo

funzionamento). Questa sezione memorizza variabili inizializzate definite nel programma e tutti questi dati sono allocati nello spazio di memoria del programma e possono essere letti e scritti.

- **Sezione .rdata (Read-Only Data):** La sezione .rdata contiene dati che non vengono modificati durante l'esecuzione del programma (come Stringhe di testo, tabelle di costanti, indirizzi di funzioni). Memorizza dati costanti e stringhe che non devono essere alterati dal programma. Questa sezione è spesso marcata come read-only dalla protezione di memoria del sistema operativo.
- **Sezione .idata:** La sezione .idata è utilizzata per la gestione delle importazioni delle funzioni da altri moduli (come tabelle di importazione, nomi delle funzioni importate, nomi delle DLLs). Contiene le informazioni necessarie per risolvere le funzioni importate, come le tabelle di importazione ed includono i nomi delle funzioni e dei moduli da cui le funzioni vengono importate.

3.4 *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.*

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

Questo file eseguibile importa due librerie principali:

- **ADVAPI32**, un modulo di sistema presente nei sistemi operativi Microsoft Windows, che fornisce una serie di funzioni avanzate per la **gestione delle applicazioni, la sicurezza e il registro di sistema**. Questa libreria è essenziale per molte operazioni

a livello di sistema, specialmente quelle legate alla sicurezza e alla gestione delle risorse. Le due funzioni importate (*RegSetValueExA, RegCreateKeyExA*) sono legate probabilmente alla **creazione di persistenza nel sistema** da parte del file, quindi alla creazione, modifica o cancellazione delle chiavi di registro Windows.

- **KERNEL32,** è una delle componenti cruciali del sistema operativo Microsoft Windows. Essa fa parte del Kernel, il nucleo del sistema operativo, e fornisce diverse **funzioni di base a basso livello** necessarie per il funzionamento del sistema. Nella scheda imports sono presenti funzioni come *SizeOfResource, LockResource, LoadResource e VirtualAlloc, CreateFile, WriteFile* che sono strettamente legate ai malware categorizzati come dropper.

3.5 Lo scopo della funzione chiamata alla locazione di memoria 00401021.

```
.text:0040101C                push    80000002h       ; hKey
.text:00401021                call    ds:RegCreateKeyExA
.text:00401027                test    eax, eax
```

La funzione chiamata alla locazione 00401021 è parte della DLL ADVAPI32, come già specificato nella domanda precedente. Questa funzione permette di **creare o aprire una chiave di registro specificata**. Nel caso la chiave esistesse già, la funzione la andrebbe ad aprire.

3.6 Come vengono passati i parametri alla funzione alla locazione 00401021.

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

I parametri vengono passati alla funzione *RegCreateKeyExA* tramite lo stack. Come evidenziato nel testo del codice notiamo prima la creazione dello stack tramite il push di EBP (Base Pointer) ed ESP (Stack Pointer) e successivamente il push dei valori:

- **NULL,** (Disposizione);
- **NULL,** (Attributi di sicurezza);
- **KEY_ALL_ACCESS**, (Accesso);
- **NULL** ,(Opzioni);
- **NULL,** (Classe);
- **0** (Riservato);
- "**SOFTWARE\\Microsoft\Windows NT\\CurrentVersion\Winlogon**" (Sottochiave);
- "**HKEY_LOCAL_MACHINE**" (Handle della chiave).

3.7 Che oggetto rappresenta il parametro alla locazione 00401017.

Il parametro alla locazione 00401017 è parte dei parametri impostati per la chiamata alla funzione RegCreateKeyExA. In questo parametro viene stabilita la subkey da aprire o creare (*SOFTWARE\\Microsoft\Windows NT\\CurrentVersion\\Winlogon*, che nel caso del nostro OS esiste).

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

- Nella locazione *00401027* viene usato "*test eax, eax*", cioè viene effettuata **un'operazione di confronto bit a bit tra due valori**, senza modificare alcuno dei valori coinvolti (simile all'operatore logico AND che a differenza di "test" può memorizzare il risultato).
- Nella locazione 00401029 invece troviamo *"jz short loc_0040132*". Questa operazione ci indica che se nella parte precedente **viene impostato uno ZERO FLAG** (ZF) in true (corrispondente ad 1 in bit), si **esegue un salto condizionale** alla locazione 00401032.

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

Il corrispondente costrutto in C sarebbe:

If (RegCreateKeyExA(HKEY_LOCAL_MACHINE, "SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\Winlogon", 0, NULL, 0, KEY_ALL_ACCESS, NULL, &hKey, NULL) != ERROR_SUCCESS) {

EAX = 1;

} else {

// Che corrisponde al salto a 00401032

}

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

```
.text:00401032
.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
```

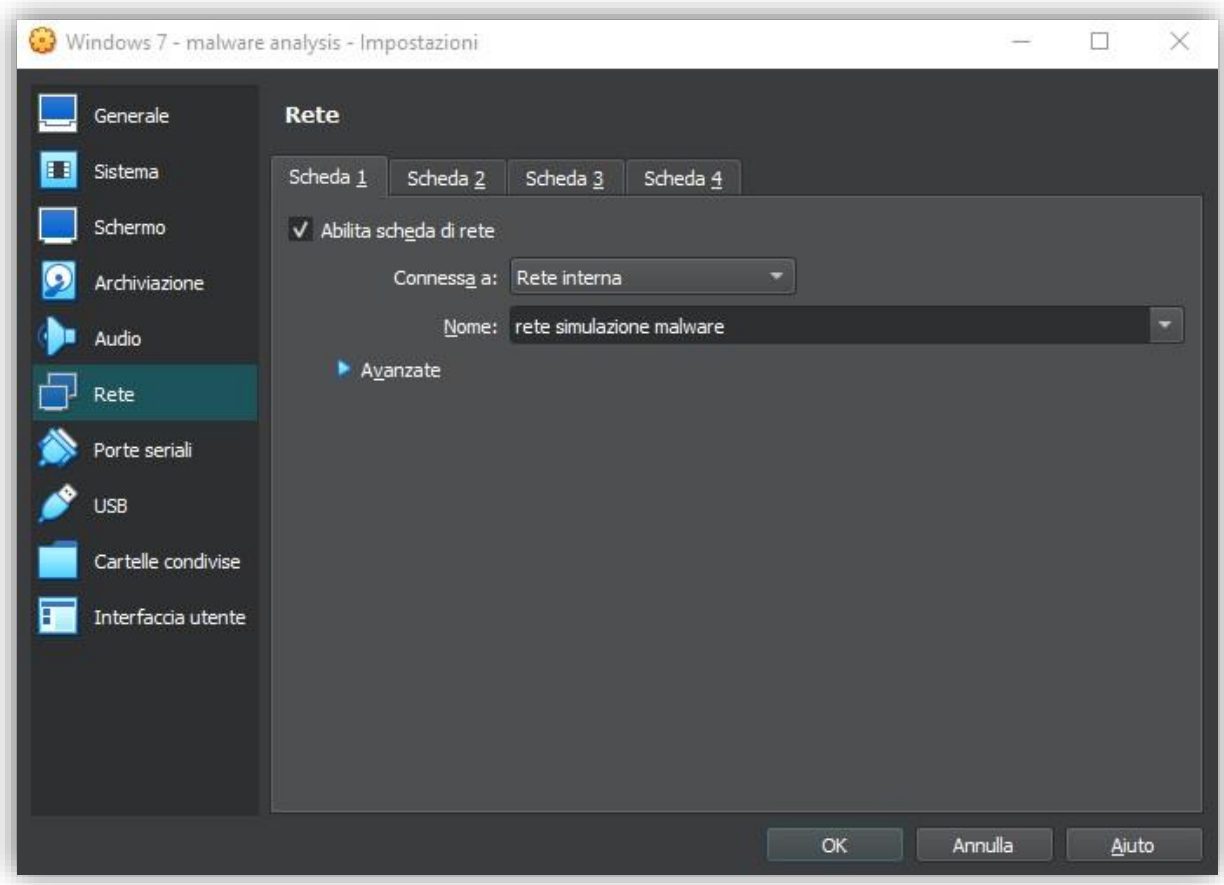
La chiamata di funzione a 00401047 è a *RegSetValueExA* sempre parte della **DLL ADVAPI32**. I parametri spinti tramite push prima di questa chiamata mostrano che *ValueName* è "**GinaDLL**" come vediamo evidenziato.

4. ANALISI DINAMICA

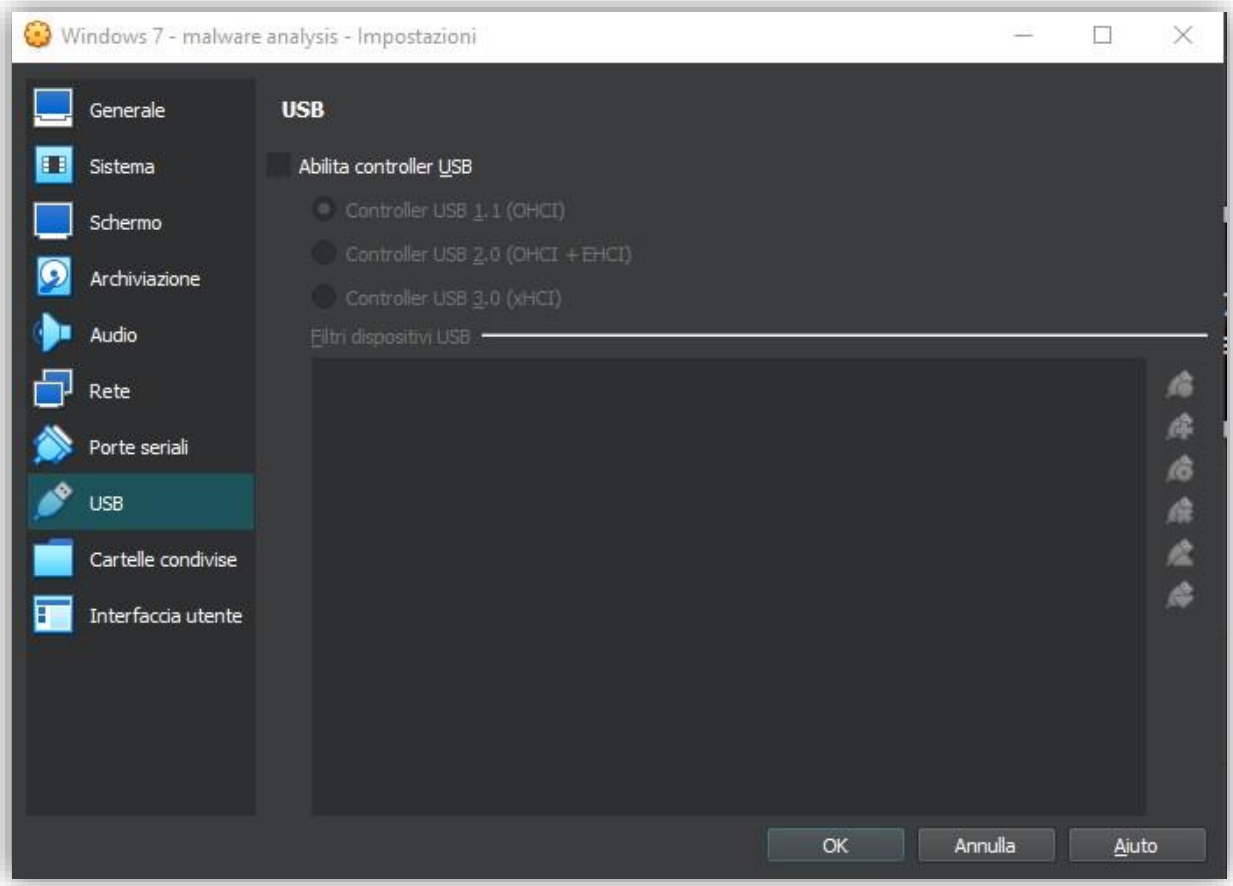
Possiamo passare ora alla fase di analisi dinamica sul dispositivo, per poter analizzare il file malevolo durante l'esecuzione.

Prima di tutto dovremo impostare il nostro laboratorio per evitare il diffondersi del malware alla nostra macchina host e creare quindi danni irriversibili.

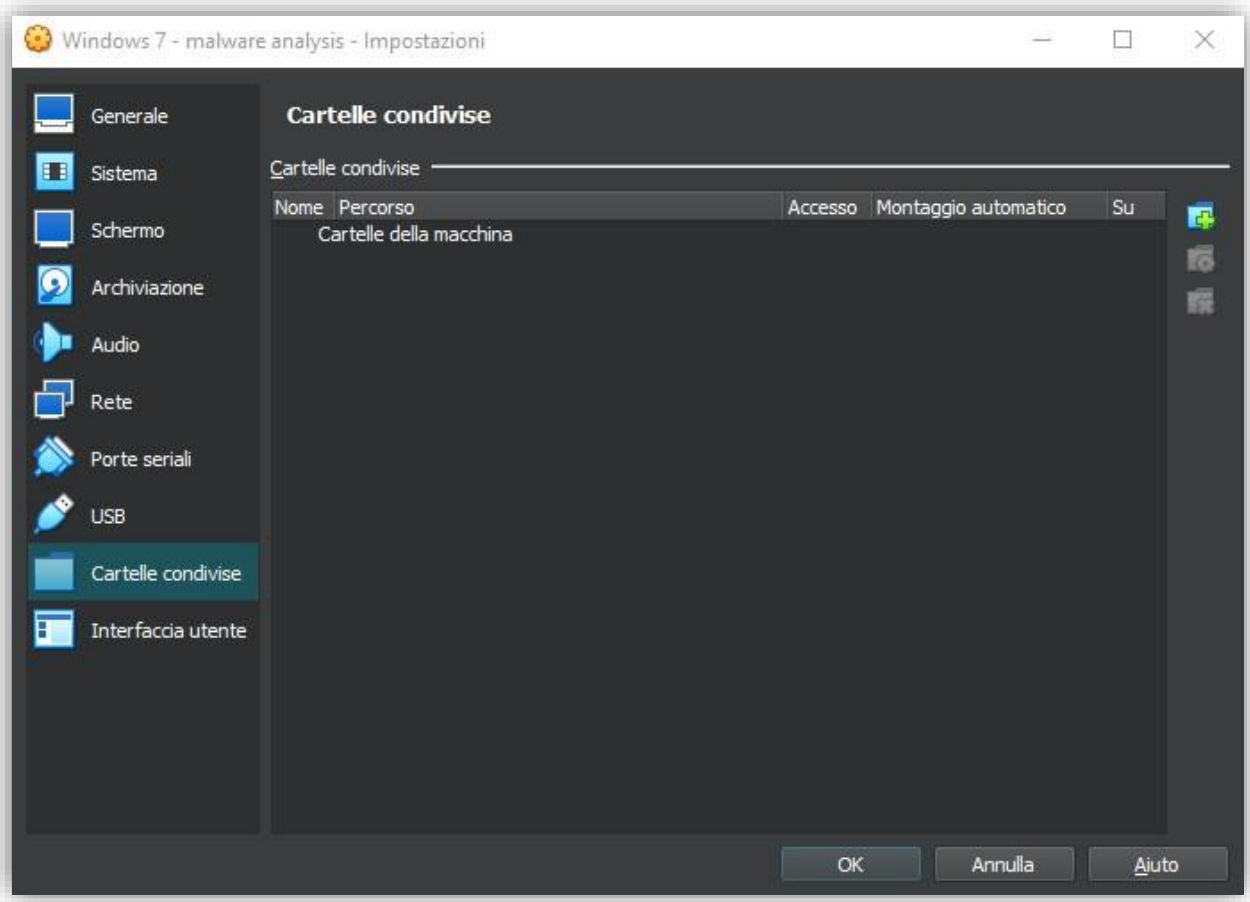
4.1 Setup e configurazione sicura della macchina host



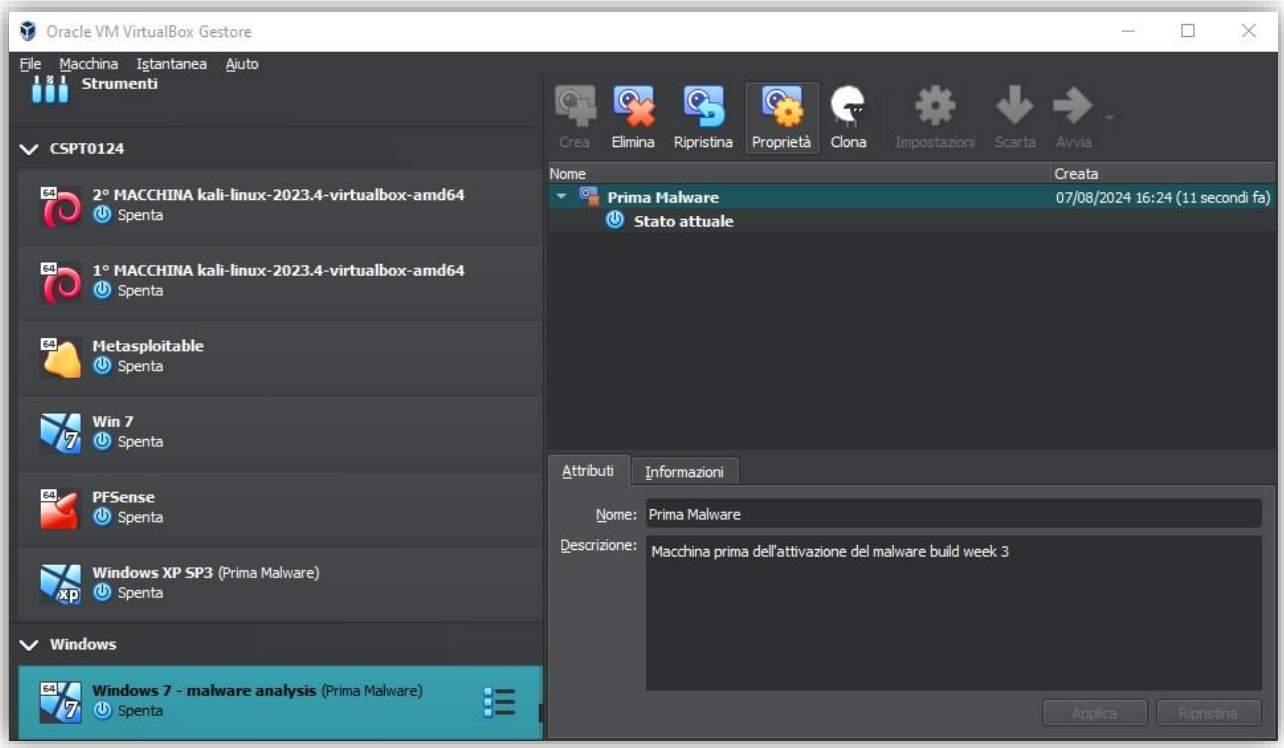
- La configurazione schede di rete nell'ambiente di test **non deve permettere alla macchina di avere accesso diretto ad Internet ed altre macchine**. La miglior configurazione è abilitare una sola scheda di rete impostata su rete interna per monitorare il traffico che genera potenzialmente il malware.



- I dispositivi USB collegati alla macchina fisica possono essere **riconosciuti anche dall'ambiente di test**. Per isolare completamente la macchina da qualsiasi collegamento esterno a sé, si consiglia di non abilitare o disabilitare il controller USB. Infatti, alcuni malware utilizzano i dispositivi USB per propagarsi su altre macchine.



- Le cartelle condivise potrebbero essere usate allo **stesso modo dei dispositivi USB**. È sempre ottima pratica accertarsi di rimuovere ogni cartella e collegamento con la macchina host per evitare la propagazione di file dannosi al di fuori del laboratorio.

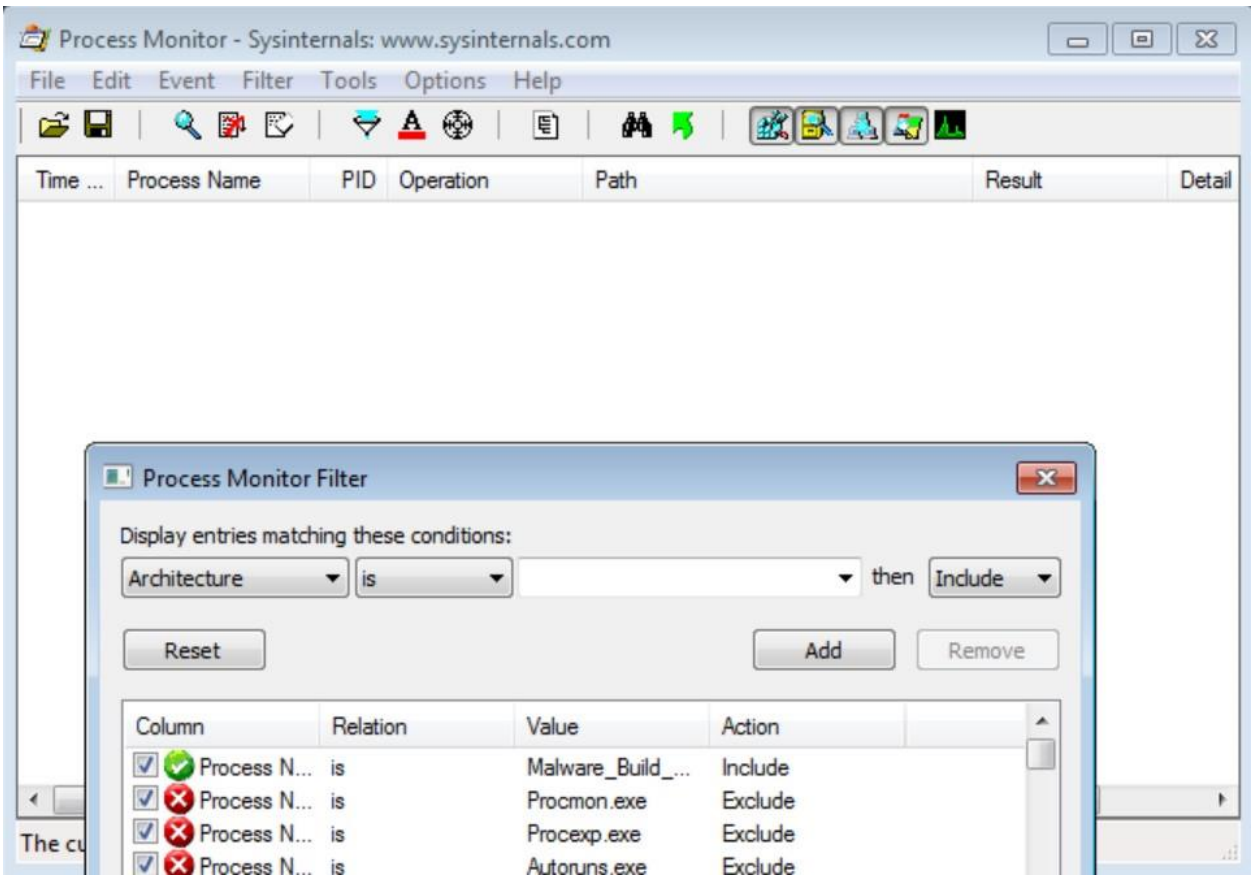


- Infine, ma non per importanza abbiamo la funzione addizionale creazione di istantanee nella macchina virtual box. Buona pratica è creare un'istantanea prima dell'utilizzo di qualsiasi file considerato malevolo in modo tale da poter **ripristinare la macchina al momento dell'istantanea** qualora si causino danni irreversibili alla macchina.

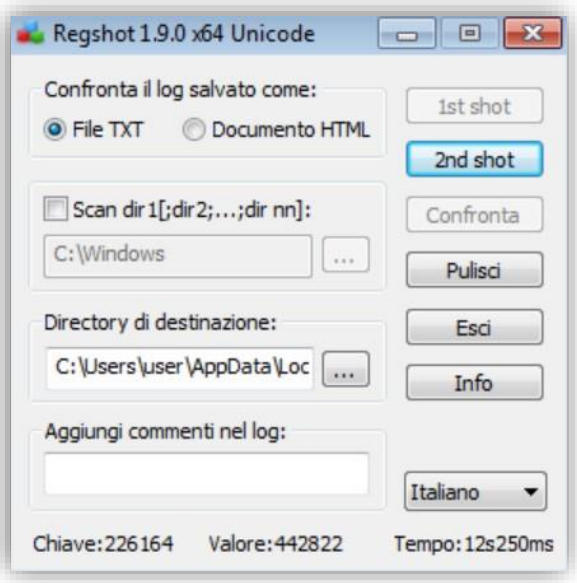
La macchina è stata **configurata correttamente**, ed ora possiamo procedere con l'attivazione dei tools necessari all'analisi del file e delle modifiche da esso comportate:

1. Il primo tool è **Procmon** uno strumento di monitoraggio avanzato per Windows fornito direttamente da Windows Sysinternals, **che mostra in tempo reale l'attività del file system, del Registro di sistema e dei processi/thread**.
2. Al secondo posto abbiamo **Regshot** un programma Open-source, portable, che ci permette di **confrontare le modifiche avvenute nel nostro computer** prima e dopo una qualsiasi operazione effettuata.

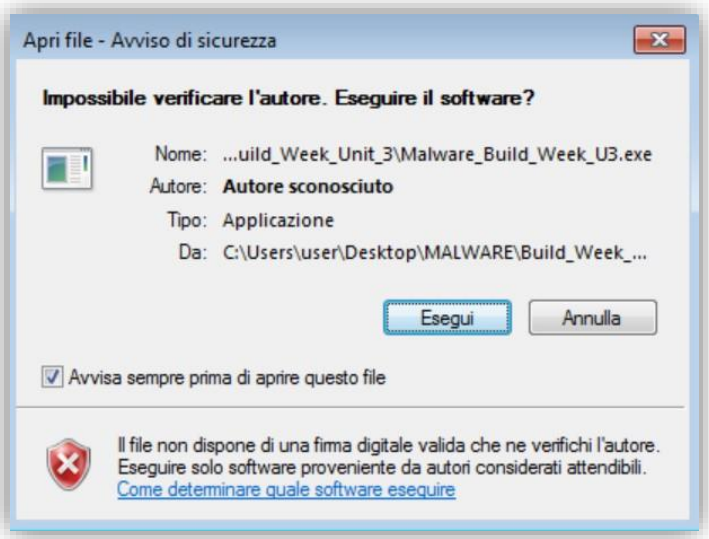
4.2 Esecuzione del malware



- Impostiamo il filtro su Procmon per poter visualizzare solo le azioni compiute dal file a cui siamo interessati.



- Effettuiamo il primo scatto tramite *regshot* a tutte le chiavi del nostro dispositivo, e teniamo il tool da parte per analisi successive all'avvio del malware.



Ora possiamo **eseguire** il malware.

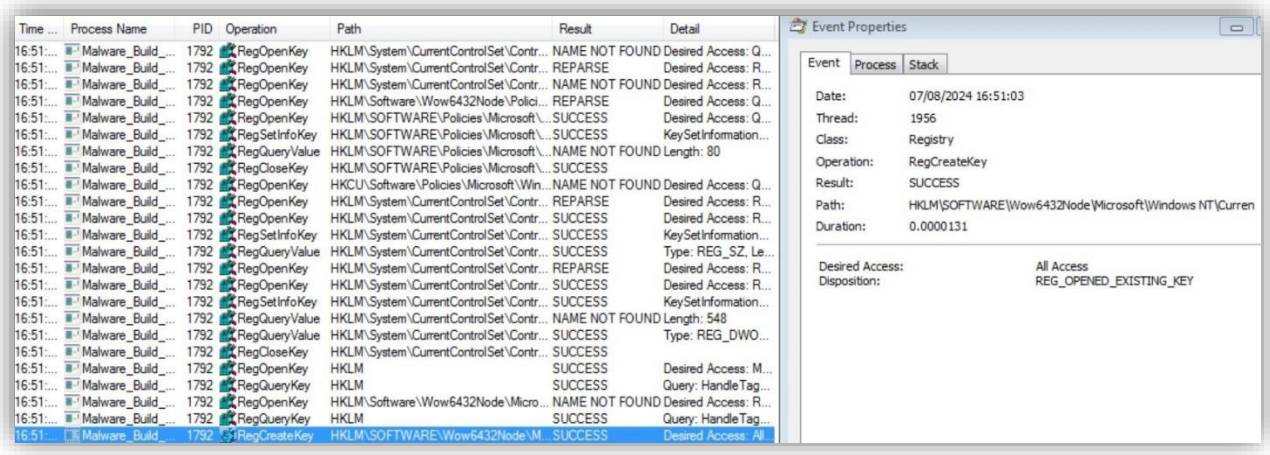
- Come risultato dall'esecuzione abbiamo un file denominato "**msgina32.ddl**", una Dynamic Link Library o libreria dinamica. Ma a cosa serve?

4.3 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

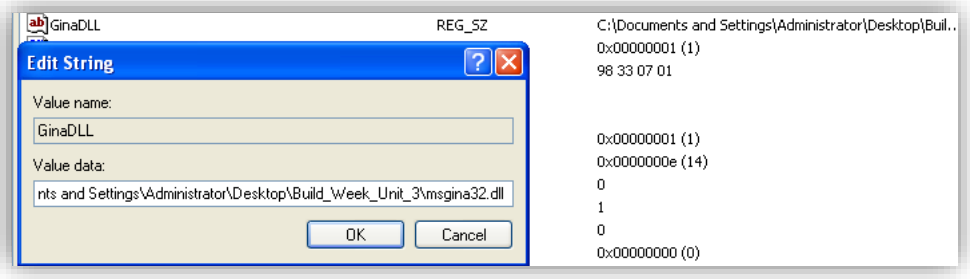
Fino ad ora siamo riusciti a capire che il malware è un dropper: una volta attivato modifica le chiavi di registro ed infine crea il file "msgina32.dll".

Tutto ciò fa pensare ad un caso di GINA interception, dove viene sostituita la vera libreria del componente Graphical Identification and Authentication, che gestisce gli accessi di windows.

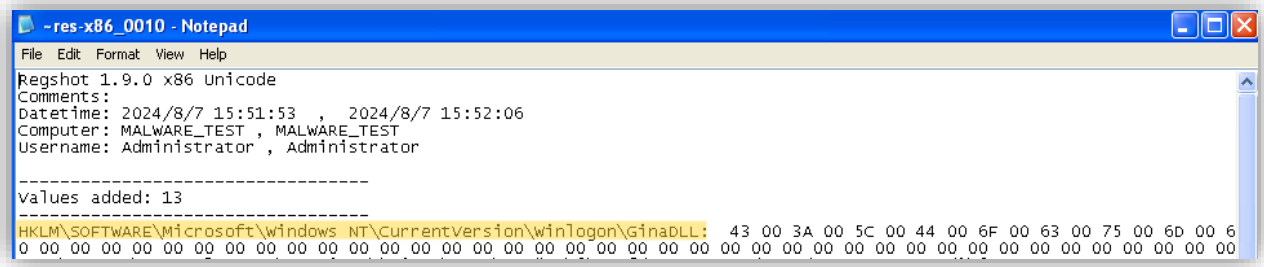
4.4 Filtrate includendo solamente l'attività sul registro di Windows, quale chiave di registro viene creata e quale valore viene associato alla chiave di registro creata?



- Su Procmon attiviamo il filtro in alto a destra per separare le operazioni di registro e troviamo un'operazione "RegCreateKey" con esito "SUCCESS" indicando che la chiave è stata creata con successo, ma approfondiamo.

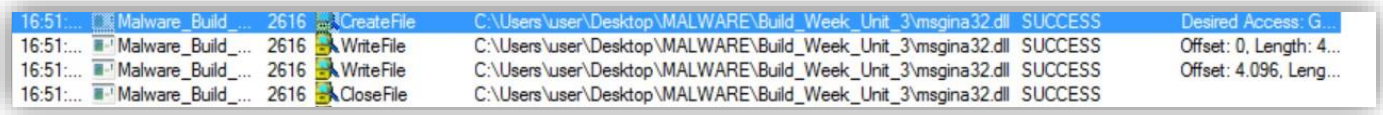


- Premiamo Tasto Win+R e digitiamo RegEdit, in modo da poter visualizzare i nostri registri Windows. Dirigiamoci alla posizione "**SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\Winlogon"** e verifichiamo il valore della chiave appena creata.
- La nostra chiave ha ora impostato come GinaDLL il file creato dal malware utilizzato in precedenza (msgina32.dll presente nel nostro Desktop, nella cartella Build_Week_Unit_3) .



- Tramite Regshot abbiamo l'ennesima prova di creazione della chiave da parte del file.

4.5 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?

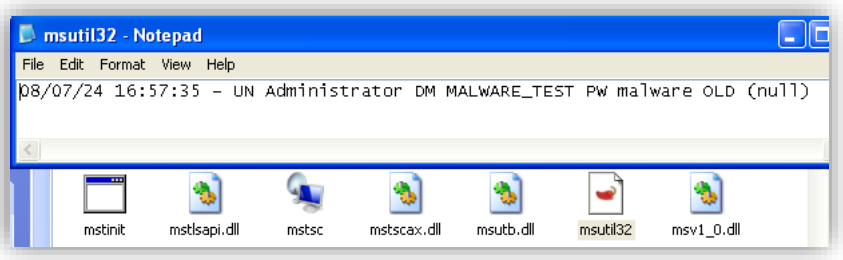


- Possiamo vedere evidenziato di blu la chiamata di funzione CreateFile dove il file viene creato, WriteFile dove viene impostato e modificato con i valori importati dal malware e CloseFile per la sua chiusura.

5. CONCLUSIONI

Il malware analizzato è al 100% un **dropper**, che installa nel sistema operativo un file malevolo che effettua un esempio di **GinaDLL interception**, una tecnica malevola utilizzata per compromettere il processo di autenticazione in sistemi operativi Windows XP. Il malware **opera sostituendo la DLL di autenticazione predefinita**, **GinaDLL**, con un file malevolo chiamato *msgina32.dll*. Questa sostituzione viene realizzata **modificando le chiavi di registro** Winlogon nel sistema, assicurando così la **persistenza del malware** ad ogni spegnimento ed accensione della macchina.

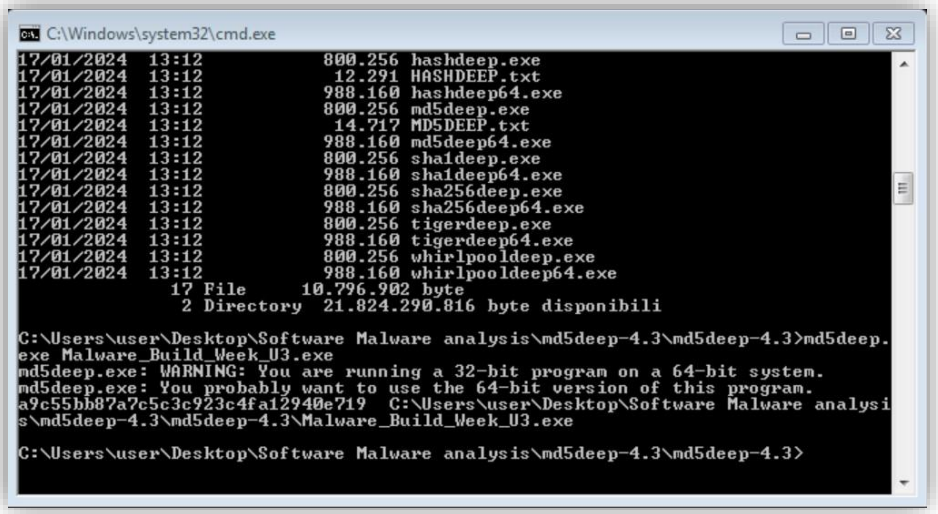
Una volta che msgina32.dll è attiva, il malware **intercetta le credenziali degli utenti durante il processo di logon**, registrandole in un file denominato *msutil32.sys*.



(File msutil32.sys aperto tramite notepad rileva le informazioni catturate degli utenti che effettuano l'accesso nel sistema operativo)

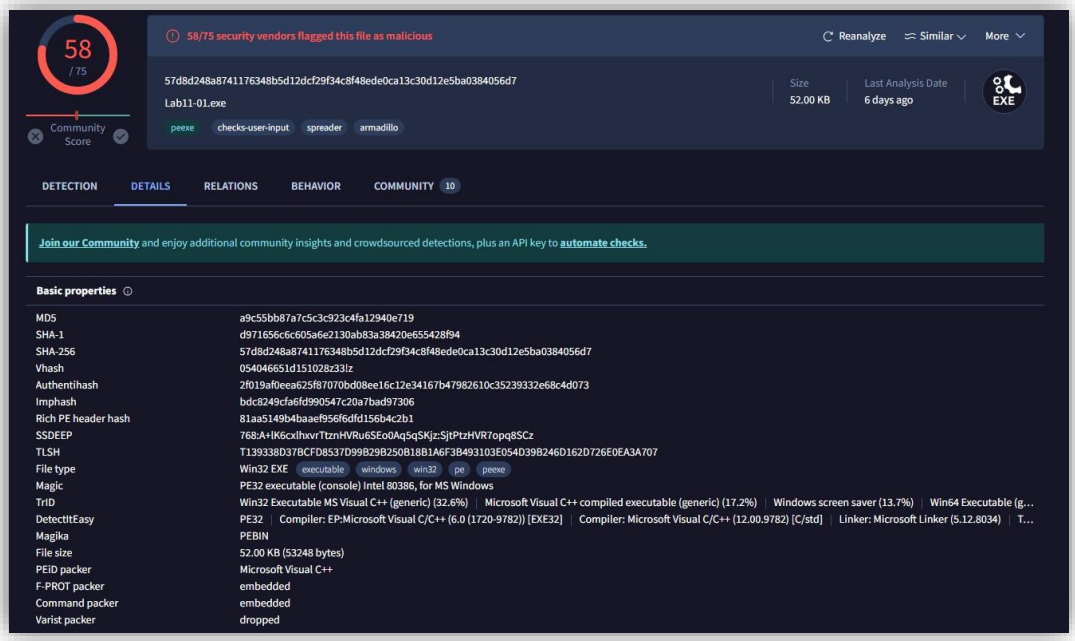
Questa tecnica permette all'attaccante di ottenere informazioni sensibili come nomi utente e password, compromettendo la sicurezza dell'intero sistema.

L'analisi evidenzia l'importanza di proteggere i sistemi legacy come Windows XP, che sono particolarmente vulnerabili a questo tipo di attacchi. L'utilizzo di sistemi aggiornati e il monitoraggio costante delle chiavi di registro critiche sono essenziali per prevenire la compromissione di queste piattaforme.



Un altro metodo efficiente per analizzare file, se non si è sicuri se siano innocui o meno, è la threat intelligence tramite siti online.

- Effettuiamo l'hash dell'eseguibile tramite md5deep.exe da CLI Windows (valore hashato del malware= a9c55bb87a7c5c3c923c4fa12940e719).
- Accediamo al sito online di VirusTotal, che ci permette di inserire un eseguibile, un URL o un hash di eseguibili. Inseriamo l'hash ottenuto da md5deep.



- Et voilà! Il file viene etichettato come malevolo da 58 vendor su 75, indicandoci altre informazioni preziose per la futura analisi del file malevolo.