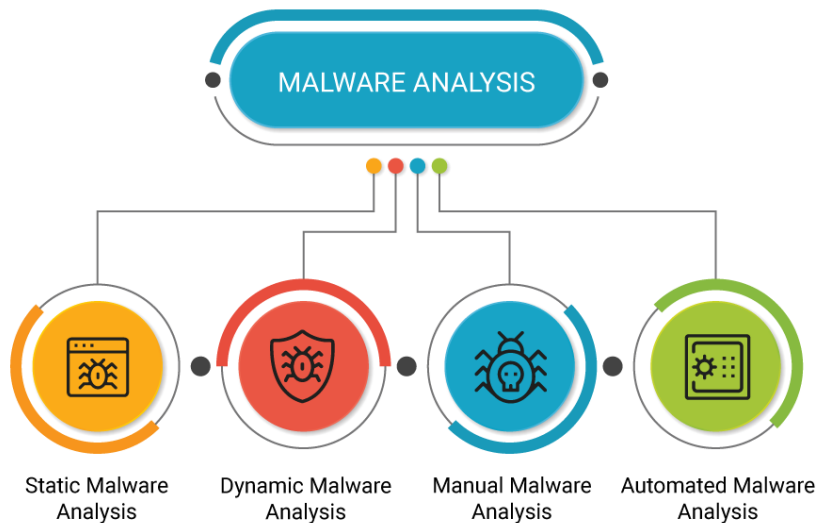


Weekly Project Report

Analisi Statica e Dinamica

TYPES OF MALWARE ANALYSIS



Il progetto di questa settimana consiste nell'effettuare **varie operazione di malware analysis**. In particolare ci viene **richiesta la risoluzione di tre task**:

Task n.1

In riferimento ad un malware presente sulla macchina Windows XP:

- identificare quali sono le sezioni di cui è composto**
- quali sono le librerie importate**

Task n.2

In riferimento ad uno spezzone di codice rinvenuto e potenzialmente appartenente ad un malware:

- identificare i costrutti noti**
- ipotizzare il comportamento**

Task n.3

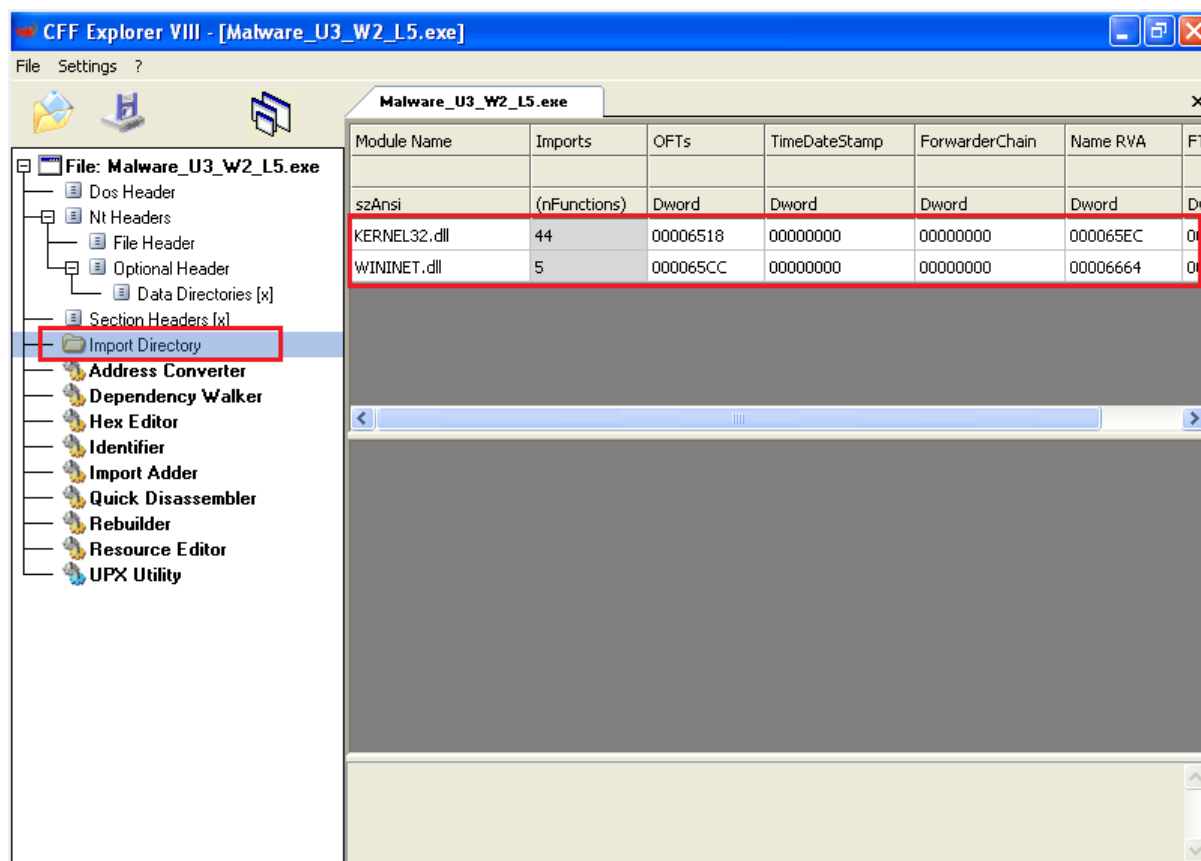
In quanto **membro senior del SOC** è richiesto di **effettuare un'analisi completa** di un file rinvenuto da un nuovo dipendente e che lui pensa essere una **potenziale minaccia**.

TASK n.1 - Malware_U3_W2_L5

Per la task n.1 andremo ad **utilizzare il tool CFF Explorer** che è un software dedicato **all'analisi dei PE** (portable execution) ovvero il formato utilizzato per la maggior parte degli eseguibili sul sistema Windows.

Una volta aperto il programma basterà caricare il file .exe del malware e lui provvederà in automatico ad analizzarlo e a **mostrarci poi una struttura ad albero** contenente diverse sezioni tra cui: **Import Directory e Section Headers**, dove la prima contiene le librerie e per ogni libreria tutte le funzioni utilizzate dal malware mentre la seconda ci mostra le sezioni di cui è composto il software.

a) Di seguito lo screen della sezione Import Directory

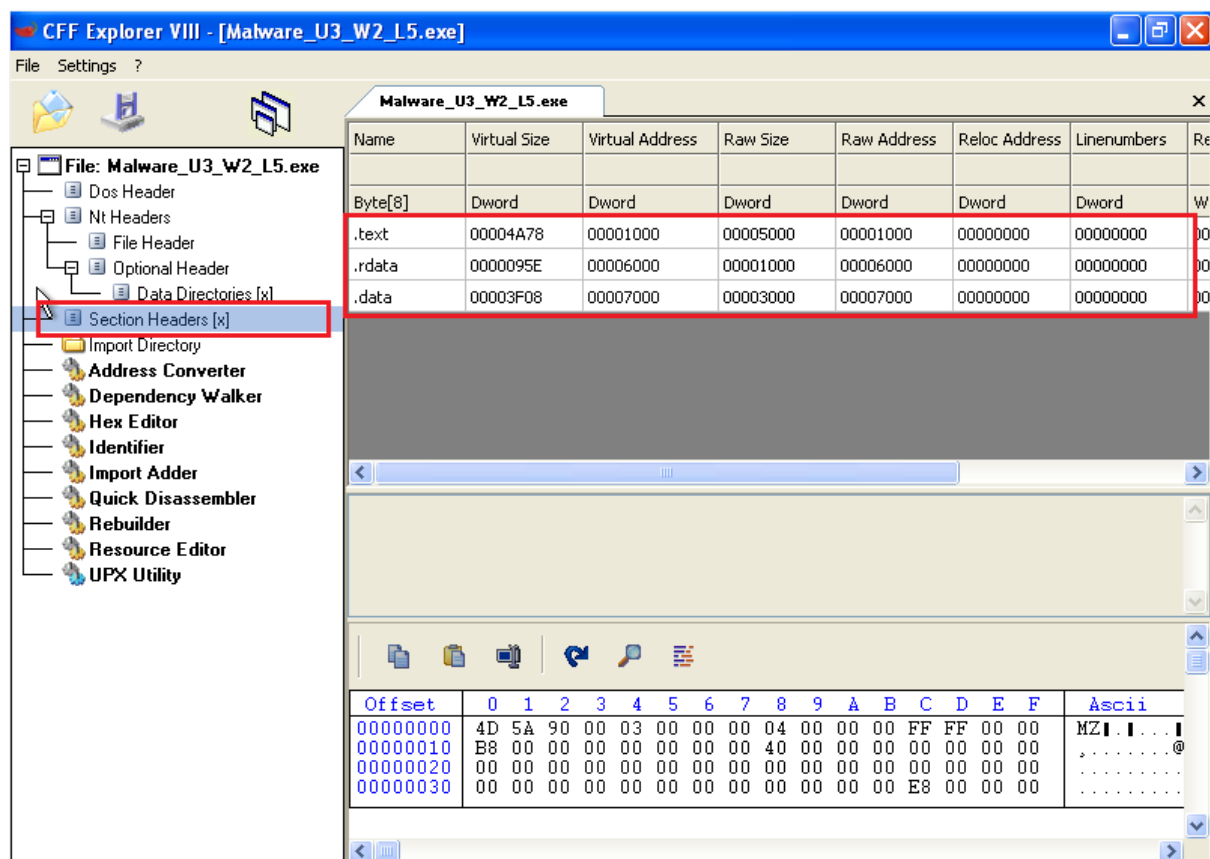


Descrizione delle librerie:

a) **Kernel32.dll** che contiene le funzioni principali per interagire con l'os come la manipolazione dei file, gestione della memoria ecc..

b) **Wininet.dll** che contiene le funzioni per l'implementazione di protocolli di rete come HTTP, FTP, NTP

Di seguito lo screen della sezione Section Headers



Descrizione delle sezioni:

- a) **.text** che include le istruzioni che la CPU eseguirà una volta avviato il software
- b) **.rdata** che include le informazioni circa le librerie e le funzioni importate ed esportate dall'eseguibile
- c) **.data** che include i dati e le variabili globali del programma eseguibile che devono essere disponibili da qualsiasi parte del programma; una variabile si dice globale quando non è definita all'interno di una funzione ma è accessibile da tutte le funzioni dell'eseguibile.

TASK n.2 - Analisi Assembly

Per la seconda task invece **procediamo con l'analisi del codice** cercando di **suddividere prima il codice in sezioni** che corrispondono ai vari costrutti e dopo aver fatto ciò **ipotizziamo un possibile comportamento del software**.

Di seguito il codice da analizzare

```
*.text:00401000      push    ebp |
*.text:00401001      mov     ebp, esp
*.text:00401003      push    ecx
*.text:00401004      push    0          ; dwReserved
*.text:00401006      push    0          ; lpdwFlags
*.text:00401008      call   ds:InternetGetConnectedState
*.text:0040100E      mov     [ebp+var_4], eax
*.text:00401011      cmp     [ebp+var_4], 0
*.text:00401015      jz      short loc_40102B
*.text:00401017      push    offset aSuccessInterne ; "Success: Internet Connection\n"
*.text:0040101C      call   sub_40117F
*.text:00401021      add     esp, 4
*.text:00401024      mov     eax, 1
*.text:00401029      jmp     short loc_40103A
*.text:0040102B loc_40102B: ; "Error 1.1: No Internet\n"
*.text:0040102B      push    offset aError1_1NoInte
*.text:00401024      call   sub_40117F
*.text:00401029      add     esp, 4
*.text:0040102B      xor     eax, eax
*.text:0040103A loc_40103A:
*.text:00401024      mov     esp, ebp
*.text:00401029      pop     ebp
*.text:0040102B      retn
*.text:0040102B      sub_401000 endp
```

a) Divisione in costrutti

Creazione dello Stack

Chiamata di funzione

Ciclo IF

Opzione 1 ciclo IF: ZF = 0

Opzione 2 ciclo IF: ZF = 1

Pulizia dello Stack

Termine funzione

*.text:00401000 *.text:00401001	push mov	ebp ebp, esp
*.text:00401003 *.text:00401004 *.text:00401006 *.text:00401008	push push push call	ecx 0 ; dwReserved 0 ; lpdwFlags ds:InternetGetConnectedState
*.text:0040100E	mov	[ebp+var_4], eax
*.text:00401011 *.text:00401015	cmp jz	[ebp+var_4], 0 short loc_40102B
*.text:00401017 *.text:0040101C *.text:00401021 *.text:00401024 *.text:00401029	push call add mov jmp	offset aSuccessInterne ; "Success: Internet Connection\n" sub_40117F esp, 4 eax, 1 short loc_40103A
*.text:0040102B loc_40102B: *.text:0040102B *.text:00401024 *.text:00401029 *.text:0040102B	push call add xor	offset aError1_1NoInte sub_40117F esp, 4 eax, eax
*.text:0040103A loc_40103A: *.text:00401024 *.text:00401029	mov pop	esp, ebp ebp
*.text:0040102B *.text:0040102B	retn sub_401000 endp	



b) Descrizione di una possibile funzione del codice

Questo codice serve ad **effettuare un controllo sullo stato della connessione** ad internet della macchina tramite **l'utilizzo della funzione "InternetGetConnectedState"** che restituisce TRUE se è presente una connessione Internet attiva tramite modem o LAN, o FALSE se non c'è una connessione Internet, o se tutte le possibili connessioni Internet non sono attualmente attive. Avendo al suo interno un **costrutto IF** nel caso dovesse essere **presente una connessione** provvederà a stampare a schermo la scritta: *Success: Internet Connection*, altrimenti provvederà a stampare a schermo la scritta: *Error 1.1: No Internet*.

c) Righe di codice rilevanti

Riga 6

.text: 00401008 call ds: InternetGetConnectedState

effettua una chiamata alla funzione "InternetGetConnectedState" contenuta nella libreria "wininet.h" la quale fornisce accesso alle funzioni per la gestione delle connessioni di rete; nello specifico la funzione di cui sopra permette di controllare se una macchina ha accesso ad internet; il ds (data segment) indica che la seguente funzione è una funzione globale che risiede appunto nel segmento di memoria .data

Riga 9

.text: 00401015 jz short loc 40102B

operazione di jump condizionale, jump zero. In questo caso il salto verso l'indirizzo loc 40102B verrà eseguito se il risultato del compare modificherà la ZF = 1 il che dipende dagli operandi del compare: se destinazione = sorgente quindi in questo caso se la variabile [ebp+var_4] = 0 allora il salto verrà effettuato.

Riga 19

.text: 00401030 xor eax, eax

esegue un'operazione con l'operatore logico XOR tra il registro EAX e se stesso, che serve ad azzerare il valore contenuto fino a quel momento in EAX ed inizializzarlo a 0, in quanto un operatore logico tra due bit identici restituisce sempre 0.

Riga 23

.text: 0040103G retn

effettua un'operazione di ritorno, che restituisce il controllo alla funzione chiamante dopo aver ripulito lo stack

Riga 22

.text: 0040103I sub_401000 endp

dichiara la fine della subroutine iniziata all'indirizzo di memoria 401000, ovvero tutto il codice sopra analizzato che si occupa di effettuare il check della connessione; ciò significa che questo frammento di codice potrebbe far parte di un codice più complesso che per funzionare o effettuare operazioni malevoli necessita di una connessione.

TASK n.3 - BONUS

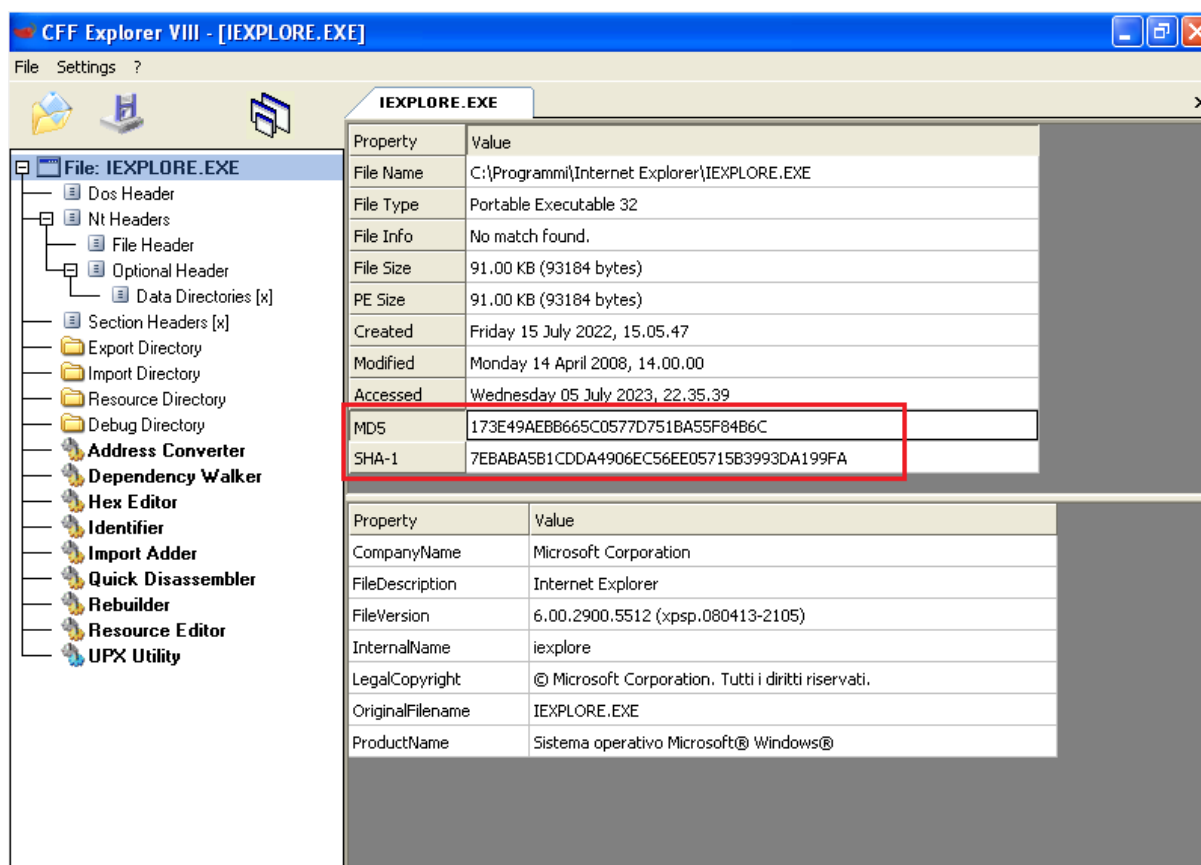
Come convincere un nuovo dipendente che il software trovato sulla sua macchina non è un malware?

Semplice basta effettuare un processo di malware analysis completo.

Un processo di malware analysis ben strutturato prevede:

- analisi hash
- analisi delle librerie importate
- analisi della composizione dell'eseguibile
- esecuzione in ambiente protetto
- monitoraggio dei processi
- monitoraggio delle operazioni effettuate
- verifica di eventuali modifiche alle chiavi di registro (lo può fare solo se ha la libreria adavapi)
- cattura di eventuali richieste verso domini sospetti (lo può fare solo se ha librerie internet)

Quindi iniziamo il nostro processo di analisi statica basica utilizzando CCF Explorer come fatto in precedenza per ottenere delle informazioni generali sul programma, tra cui l'hash, le librerie importate e le sezioni di cui è composto.



Dopo aver trovato l'hash ci spostiamo su Virus Total ed effettuiamo una ricerca per vedere il suo rating.

Come si può notare dopo aver inserito l'hash su VT vediamo subito che si tratta di un file distribuito da Microsoft, con un grado di rischio 0, il cui nome e size corrispondono esattamente a quello trovato sulla macchina del dipendente.

0

/ 64

File distributed by Microsoft

f6ef777eef4f15a9b3bba4295e0bbe3acb5886a9e4d2b5aa72ca25e8f396b9cd

IEXPLORE.EXE

peexe trusted known-distributor

Size

91.00 KB

Last Analysis Date

5 years ago

Reanalyze

Similar

More

Community Score

DETECTION

DETAILS

BEHAVIOR

COMMUNITY 2

Security vendors' analysis

Do you want to automate checks?

Ad-Aware	Undetected	AegisLab	Undetected
AhnLab-V3	Undetected	ALYac	Undetected
Antiy-AVL	Undetected	Arcabit	Undetected
Avast	Undetected	Avast-Mobile	Undetected

Poi effettuiamo un check delle varie sezioni.

CFF Explorer VIII - [IEXPLORE.EXE]

File Settings ?

IEXPLORE.EXE

File: IEXPLORE.EXE

Dos Header

Nt Headers

File Header

Optional Header

Data Directories [x]

Section Headers [x]

Export Directory

Import Directory

Resource Directory

Debug Directory

Address Converter

Dependency Walker

Hex Editor

Identifier

Import Adder

Quick Disassembler

Rebuilder

Resource Editor

UPX Utility

Name	Virtual Size	Virtual Address	Raw Size	Raw Address	Reloc Address	Linenumbers	Re
Byte[8]	Dword	Dword	Dword	Dword	Dword	Dword	W
.text	00001D9B	00001000	00001E00	00000400	00000000	00000000	00
.data	0000009C	00003000	00000200	00002200	00000000	00000000	00
.rsrc	00014730	00004000	00014800	00002400	00000000	00000000	00

Offset

0 1 2 3 4 5 6 7 8 9 A B C D E F

00000000

4D 5A 90 00 03 00 00 00 00 04 00 00 00 FF FF 00 00

00000010

B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00

00000020

00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Ascii

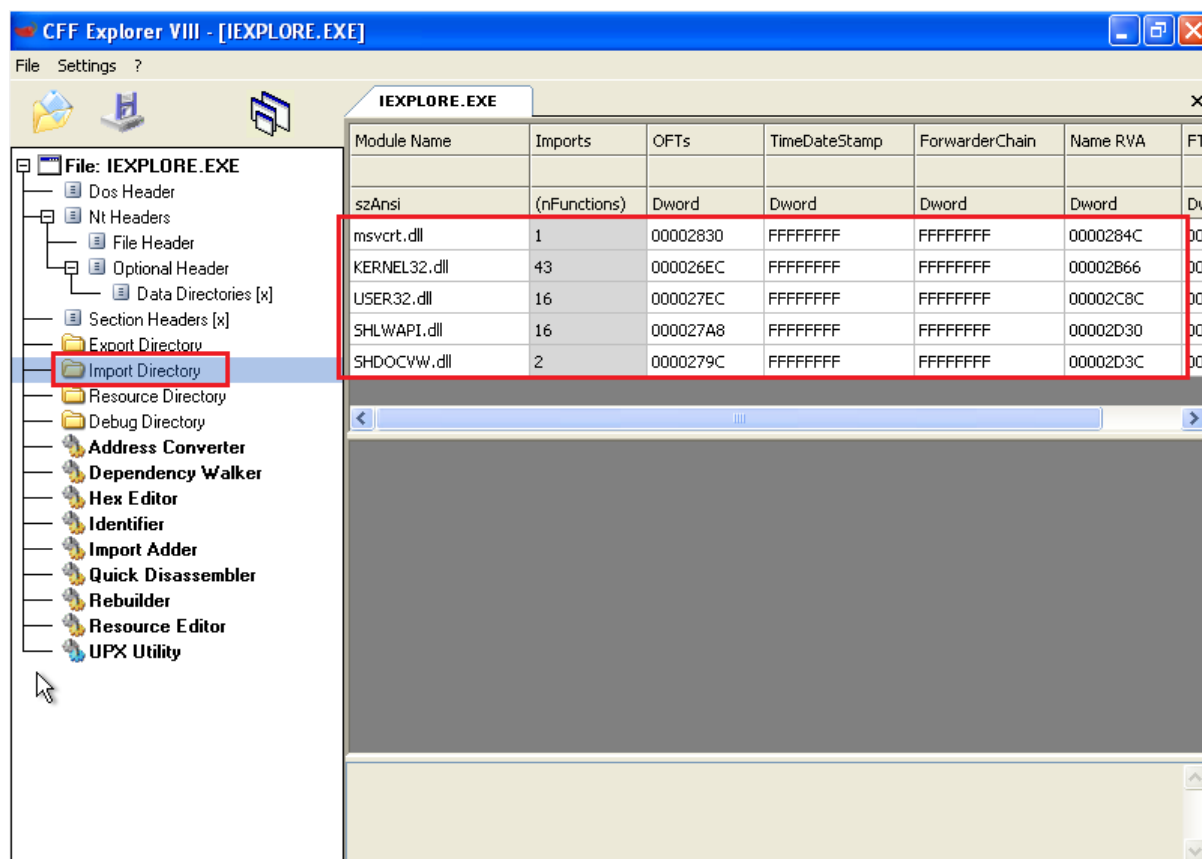
MZ

. @

.

Le sezioni text e data le conosciamo già, approfondiamo solo la sezione:

- c) .rsrc che include le risorse utilizzate dall'eseguibile come icone, immagini, menu e stringhe che non sono parte dell'eseguibile stesso



In questo caso invece sono diverse le librerie che non conosciamo, andiamo ad approfondirle:

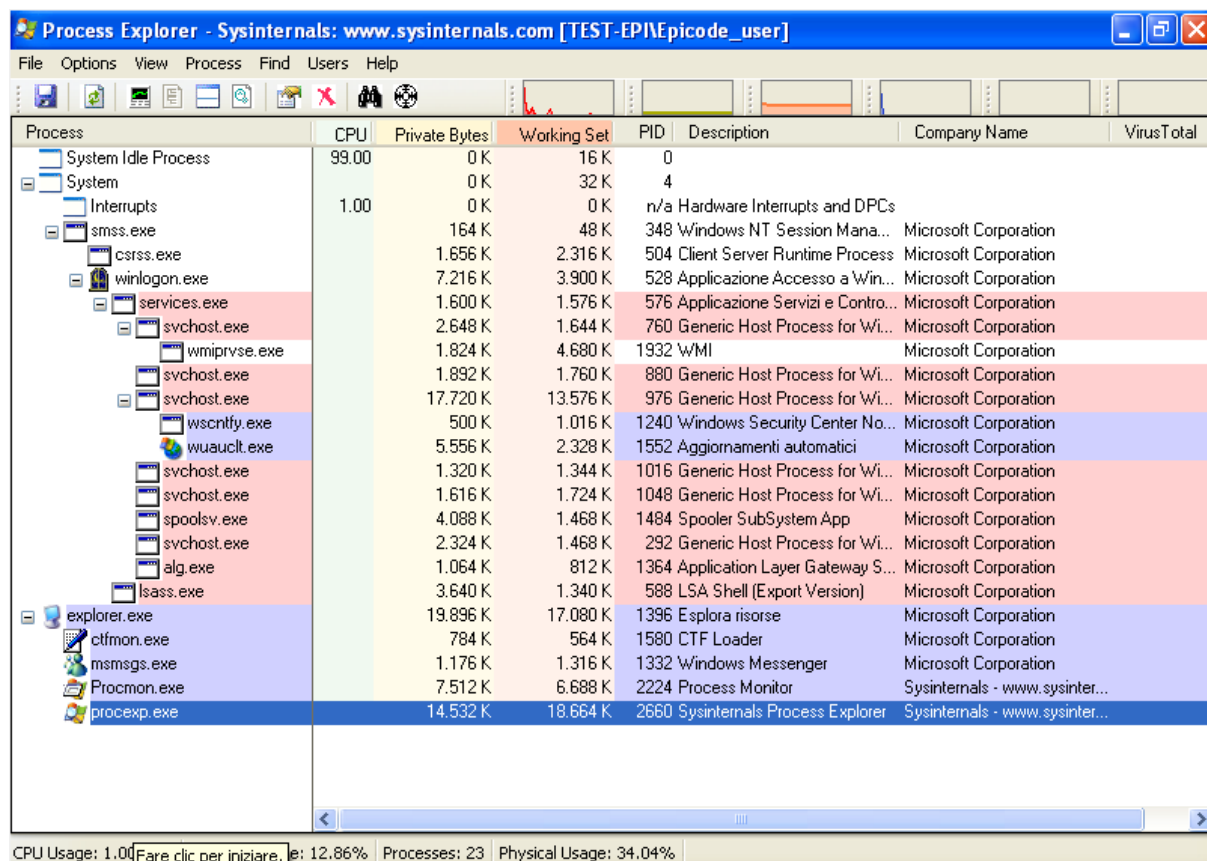
- USER32.dll** che include una serie di funzioni che consentono l'interazione dell'utente con l'interfaccia grafica del sistema operativo, come la gestione delle finestre; funzioni per l'input e output e altre funzioni per il disegno di elementi grafici
- MSVCRT.dll** che include funzioni per la manipolazione stringhe, allocazione memoria e altro come chiamate per input/output in stile linguaggio C
- SHLWAPI.dll** è l'acronimo di "Shell Light-Weight API" e include una serie di funzioni utili per la manipolazione di stringhe e operazioni su percorsi di file.
- SHDOCVW.dll** che include funzioni per la gestione delle finestre del browser Web e l'interazione con i componenti di navigazione e visualizzazione dei contenuti Web.

Anche se dalla fase di analisi statica basica appena effettuata il file non risulta essere un malware per poter esserne certi procediamo con ulteriori controlli utilizzando dei tool appositi per l'analisi dinamica (basica) la quale prevede l'esecuzione del malware così da poterne verificare il comportamento e di avere così più informazioni circa il funzionamento dell'eseguibile.

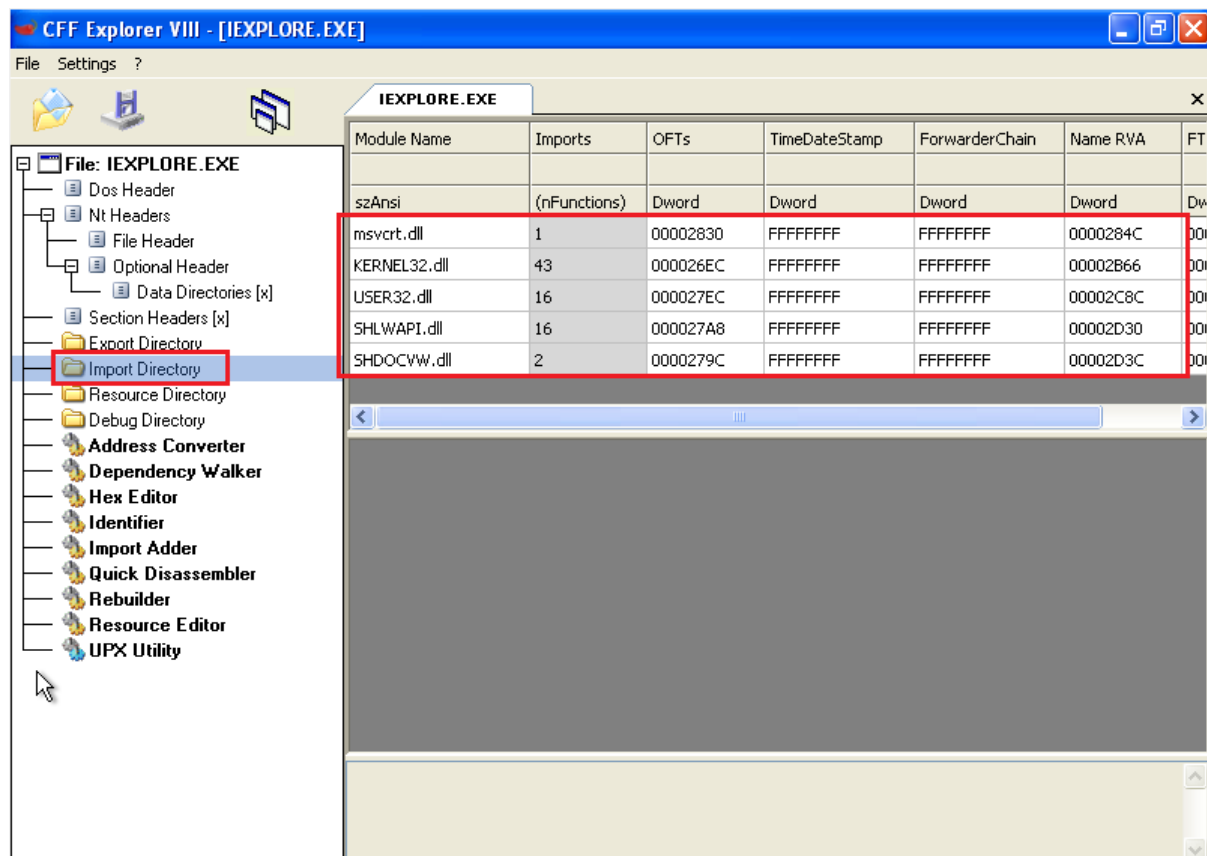
I tool che andremo ad utilizzare sono:

- a) **Process Explorer** che è un tool che permette di effettuare un'analisi dettagliata di tutti i processi in esecuzione

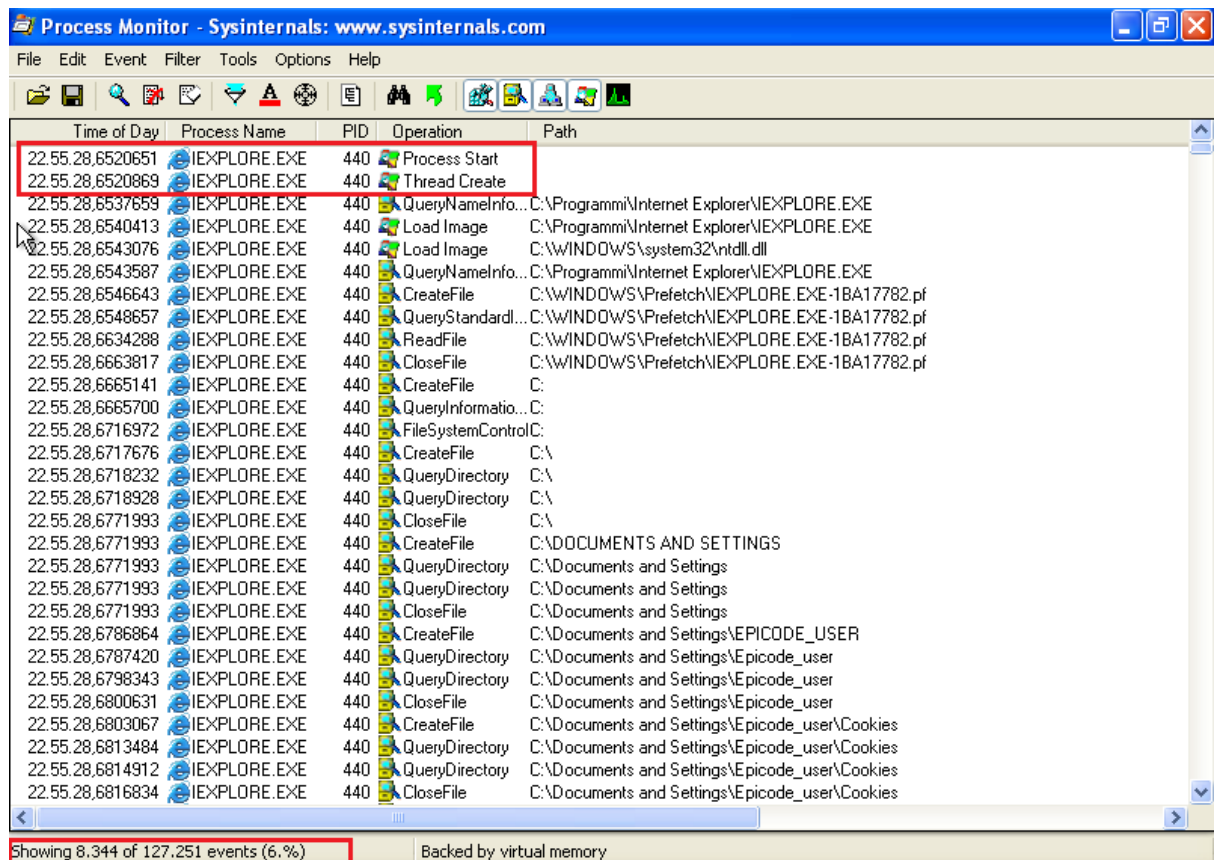
Prima



Dopo

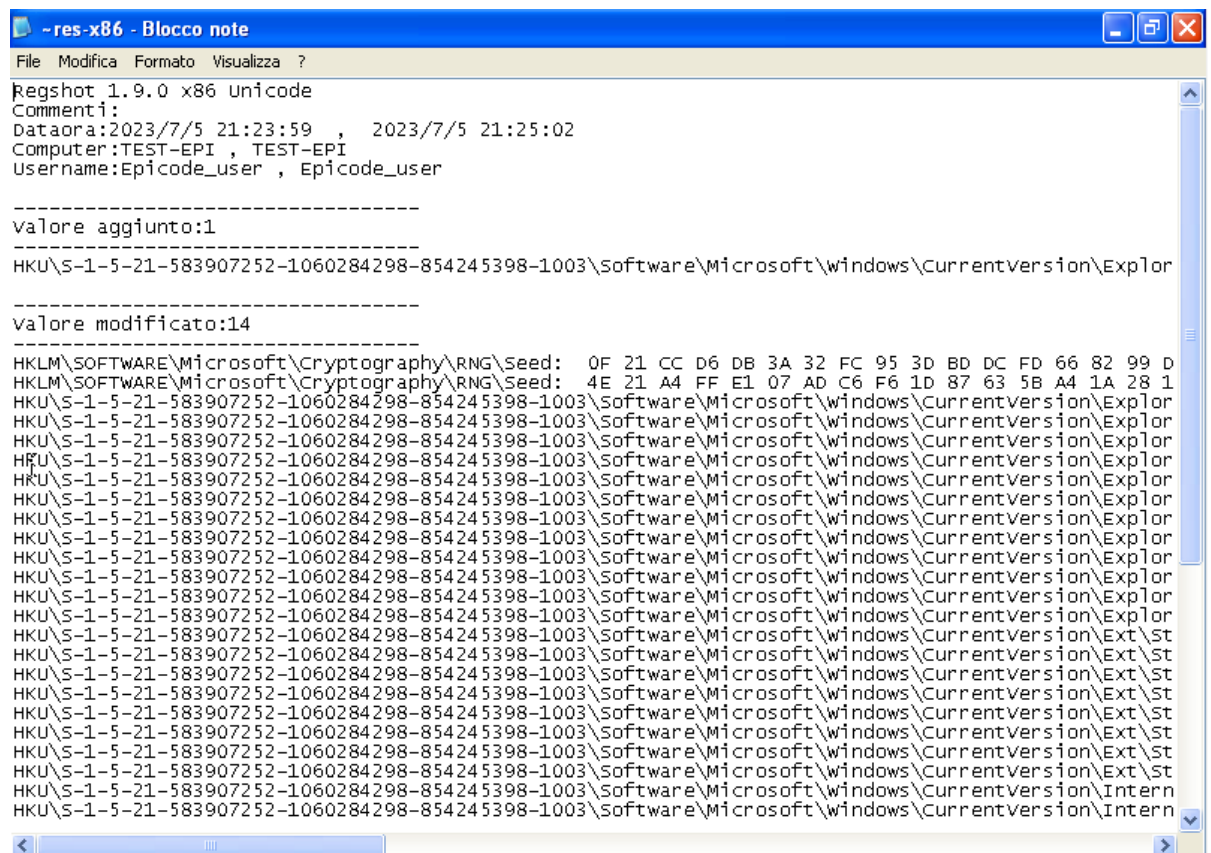


- a) **Process Monitor** che è un tool avanzato di Windows che permette di monitorare i thread e i processi attivi, l'accesso ai file e le chiamate di sistema effettuate.



Dove applicando un filtro con il PID del processo in analisi riusciamo ad avere una situazione più chiara di tutte le operazioni effettuate una volta eseguito il file.

- b) **Regshot** è un tool che permette di paragonare in automatico due istante delle chiavi di registro effettuate in momenti diversi solitamente prima di avviare il malware e successivamente alla sua esecuzione così da poter confrontare i risultati ed evidenziare tutte le modifiche apportate dal malware.



```
-res-x86 - Blocco note
File Modifica Formato Visualizza ?
Regshot 1.9.0 x86 Unicode
Commenti:
Dataora:2023/7/5 21:23:59 , 2023/7/5 21:25:02
Computer:TEST-EPI , TEST-EPI
Username:Epicode_user , Epicode_user

-----
valore aggiunto:1
-----
HKU\S-1-5-21-583907252-1060284298-854245398-1003\Software\Microsoft\windows\CurrentVersion\Explorer

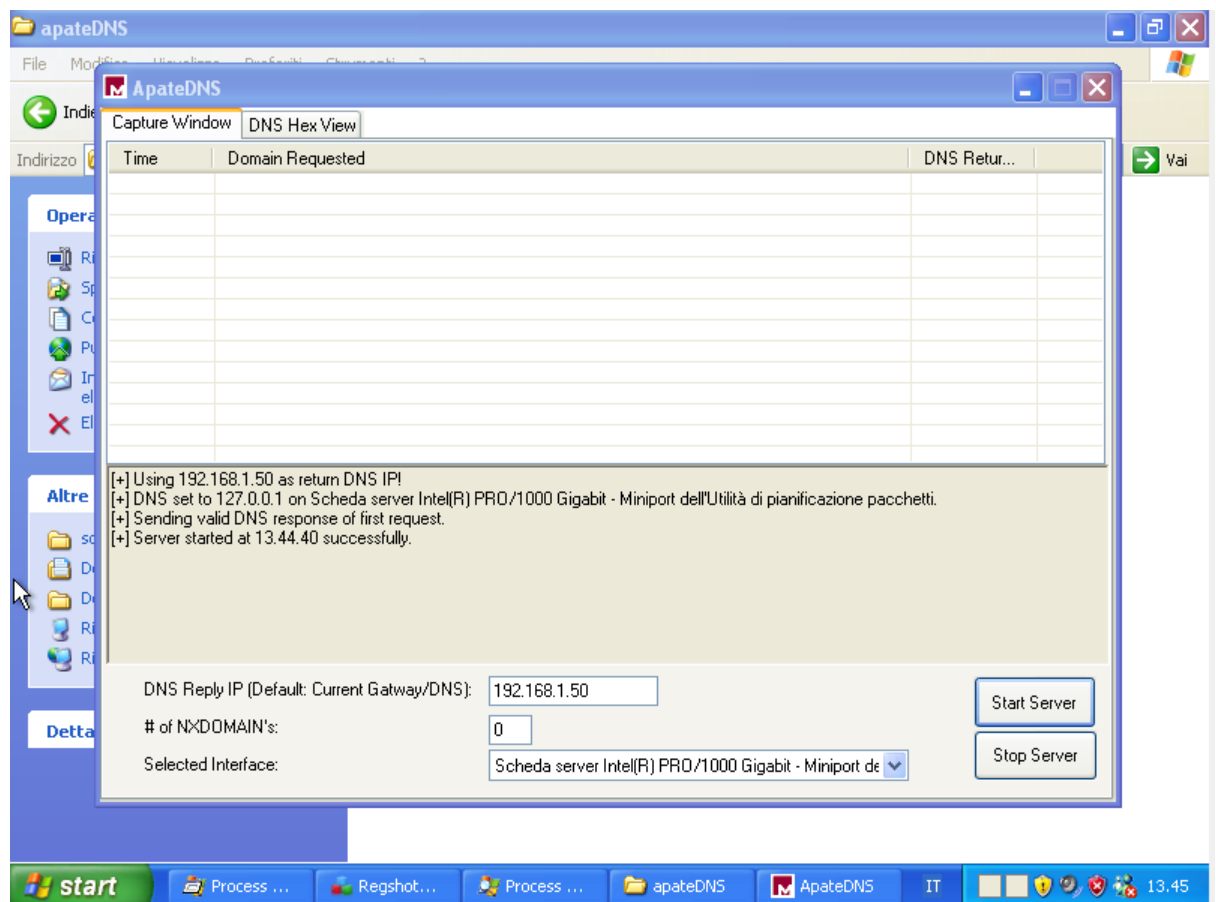
-----
valore modificato:14
-----
HKLM\SOFTWARE\Microsoft\Cryptography\RNG\Seed: 0F 21 CC D6 DB 3A 32 FC 95 3D BD DC FD 66 82 99 D
HKLM\SOFTWARE\Microsoft\Cryptography\RNG\Seed: 4E 21 A4 FF E1 07 AD C6 F6 1D 87 63 5B A4 1A 28 1
HKU\S-1-5-21-583907252-1060284298-854245398-1003\Software\Microsoft\windows\CurrentVersion\Explorer
HKU\S-1-5-21-583907252-1060284298-854245398-1003\Software\Microsoft\windows\CurrentVersion\Explorer
HKU\S-1-5-21-583907252-1060284298-854245398-1003\Software\Microsoft\windows\CurrentVersion\Explorer
HKU\S-1-5-21-583907252-1060284298-854245398-1003\Software\Microsoft\windows\CurrentVersion\Explorer
HKU\S-1-5-21-583907252-1060284298-854245398-1003\Software\Microsoft\windows\CurrentVersion\Explorer
HKU\S-1-5-21-583907252-1060284298-854245398-1003\Software\Microsoft\windows\CurrentVersion\Explorer
HKU\S-1-5-21-583907252-1060284298-854245398-1003\Software\Microsoft\windows\CurrentVersion\Explorer
HKU\S-1-5-21-583907252-1060284298-854245398-1003\Software\Microsoft\windows\CurrentVersion\Explorer
HKU\S-1-5-21-583907252-1060284298-854245398-1003\Software\Microsoft\windows\CurrentVersion\Ext\St
HKU\S-1-5-21-583907252-1060284298-854245398-1003\Software\Microsoft\windows\CurrentVersion\Ext\St
HKU\S-1-5-21-583907252-1060284298-854245398-1003\Software\Microsoft\windows\CurrentVersion\Ext\St
HKU\S-1-5-21-583907252-1060284298-854245398-1003\Software\Microsoft\windows\CurrentVersion\Ext\St
HKU\S-1-5-21-583907252-1060284298-854245398-1003\Software\Microsoft\windows\CurrentVersion\Ext\St
HKU\S-1-5-21-583907252-1060284298-854245398-1003\Software\Microsoft\windows\CurrentVersion\Ext\St
HKU\S-1-5-21-583907252-1060284298-854245398-1003\Software\Microsoft\windows\CurrentVersion\Ext\St
HKU\S-1-5-21-583907252-1060284298-854245398-1003\Software\Microsoft\windows\CurrentVersion\Ext\St
HKU\S-1-5-21-583907252-1060284298-854245398-1003\Software\Microsoft\windows\CurrentVersion\Intern
```

Qui vediamo come il confronto delle scansioni sulle chiavi di registro prima e dopo l'esecuzione del file non abbia apportato grandi cambiamenti.

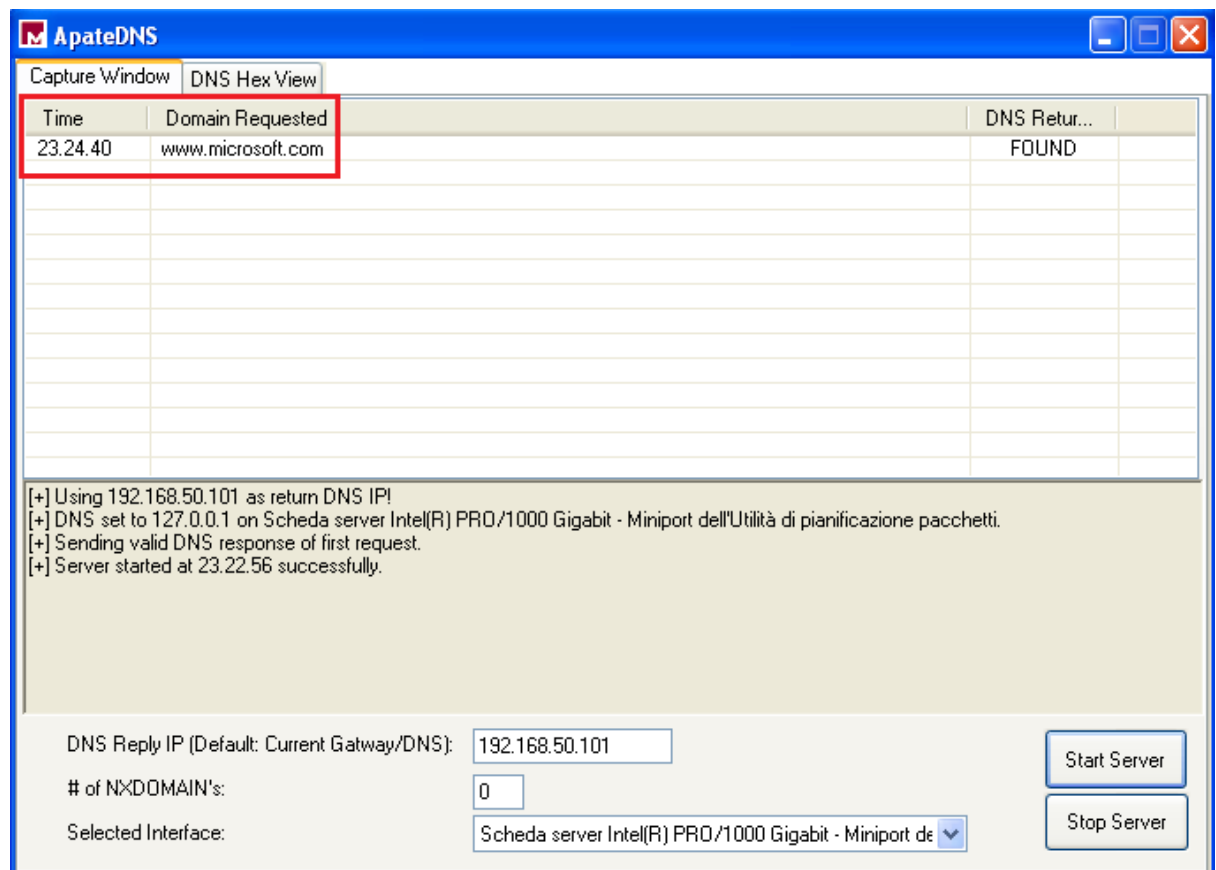
(sarà perchè non c'è la libreria adavapi.dll?)

- c) **ApateDNS** è un altro tool utile per il monitoraggio del traffico di rete e può essere utilizzato per simulare un server Dns

Prima



Dopo



Ovviamente andiamo ad effettuare uno scan anche il dominio individuato e anche in questo caso risulta assolutamente pulito.

Possiamo quindi concludere dicendo al nuovo impiegato di poter stare tranquillo in quanto il software segnalato è legittimo al 100%.