

GIULIA FIACCHI

S10/LS

PROGETTO





# TRACCIA

Con riferimento al file Malware\_U3\_W2\_L5 presente all'interno della cartella «**Esercizio\_Pratico\_U3\_W2\_L5**» sul desktop della macchina virtuale dedicata per l'analisi dei malware, rispondere ai seguenti quesiti:

1. Quali **librerie** vengono importate dal file eseguibile ? Fare anche una descrizione
2. Quali sono le **sezioni** di cui si compone il file eseguibile del malware? Fare anche una descrizione

Con riferimento alla figura in slide 3, risponde ai seguenti quesiti:

3. Identificare i **costrutti noti** (creazione dello stack, eventuali cicli, altri costrutti )
4. Ipotizzare il **comportamento della funzionalità implementata**
5. Fare una **tabella** per spiegare il significato delle singole righe di codice

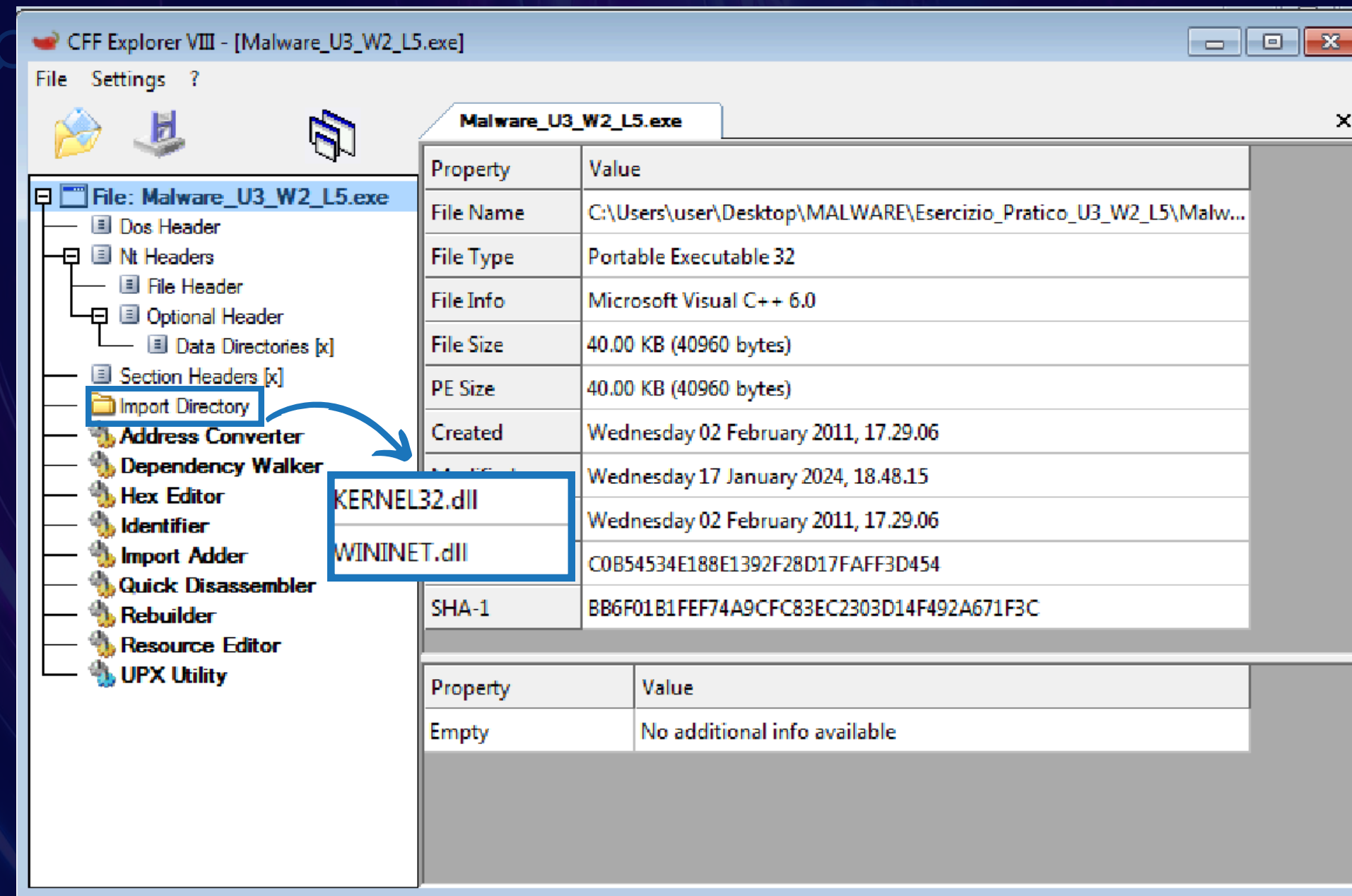


# PARTE 1

## LIBRERIE IMPORTATE

Come prima cosa avviamo la nostra macchina "Window 7 - malware analysis", poi clicchiamo sulla cartella che si trova nel Desktop con il nome di "**Software Malware Analysis**" e da qui apriamo il software di nostro interesse che è **CFF Explorer**.

Una volta aperto è possibile da qui caricare un file per la seguente scansione, quindi clicchiamo sulla cartellina in alto a destra e cerchiamo il file di nostro interesse; che troviamo nella cartella "**MALWARE**" con il nome di «**Esercizio\_Pratico\_U3\_W2\_L5**».





# PARTE 1

## LIBRERIE IMPORTATE

Cliccando nella sezione “**Import directory**” possiamo trovare le librerie importate dal malware:

- **KERNEL32.DLL**: È essenziale per molte operazioni di base del **sistema operativo**. Necessaria per la gestione delle risorse del sistema, come la memoria, i processi e i thread, l'accesso ai file e la gestione degli input/output. È una delle librerie più fondamentali di Windows e viene utilizzata da quasi tutte le applicazioni per eseguire operazioni comuni come la **gestione della memoria**, **l'accesso ai file e la comunicazione tra processi** (ES: CreateFile, ReadFile, WriteFile, CreateProcess, Sleep, GetTickCount). La gestione efficiente della memoria è cruciale per la stabilità e le prestazioni del sistema operativo e delle applicazioni.
- **WININET.dll**: fornisce le funzioni necessarie per **l'accesso a Internet e per la gestione delle comunicazioni di rete**, fondamentale per applicazioni che richiedono l'accesso a risorse online, come browser web, client di posta elettronica, software di aggiornamento. Comprende set di API che permettono alle applicazioni di interagire con i protocolli di Internet, come **HTTP e FTP**. Si occupa anche della **gestione dei cookie**, cruciali per mantenere le sessioni web e per la personalizzazione dell'esperienza utente nelle applicazioni web. (ES: InternetOpen, InternetConnect, HttpOpenRequest, InternetReadFile, FtpGetFile.)

Poiché kernel32.dll gestisce le risorse di sistema fondamentali, la sua integrità è cruciale per la stabilità e il corretto funzionamento di Windows. Problemi con questa libreria possono causare crash di sistema, blocchi e comportamenti imprevedibili. wininet.dll è essenziale per tutte le applicazioni che richiedono l'accesso a Internet. Senza questa libreria, i browser web, i client di posta elettronica e molte altre applicazioni non potrebbero funzionare correttamente. Entrambe le librerie includono funzioni critiche per la sicurezza e per questo sono frequentemente aggiornate da Microsoft per risolvere bug e migliorare la sicurezza.



# PARTE 2

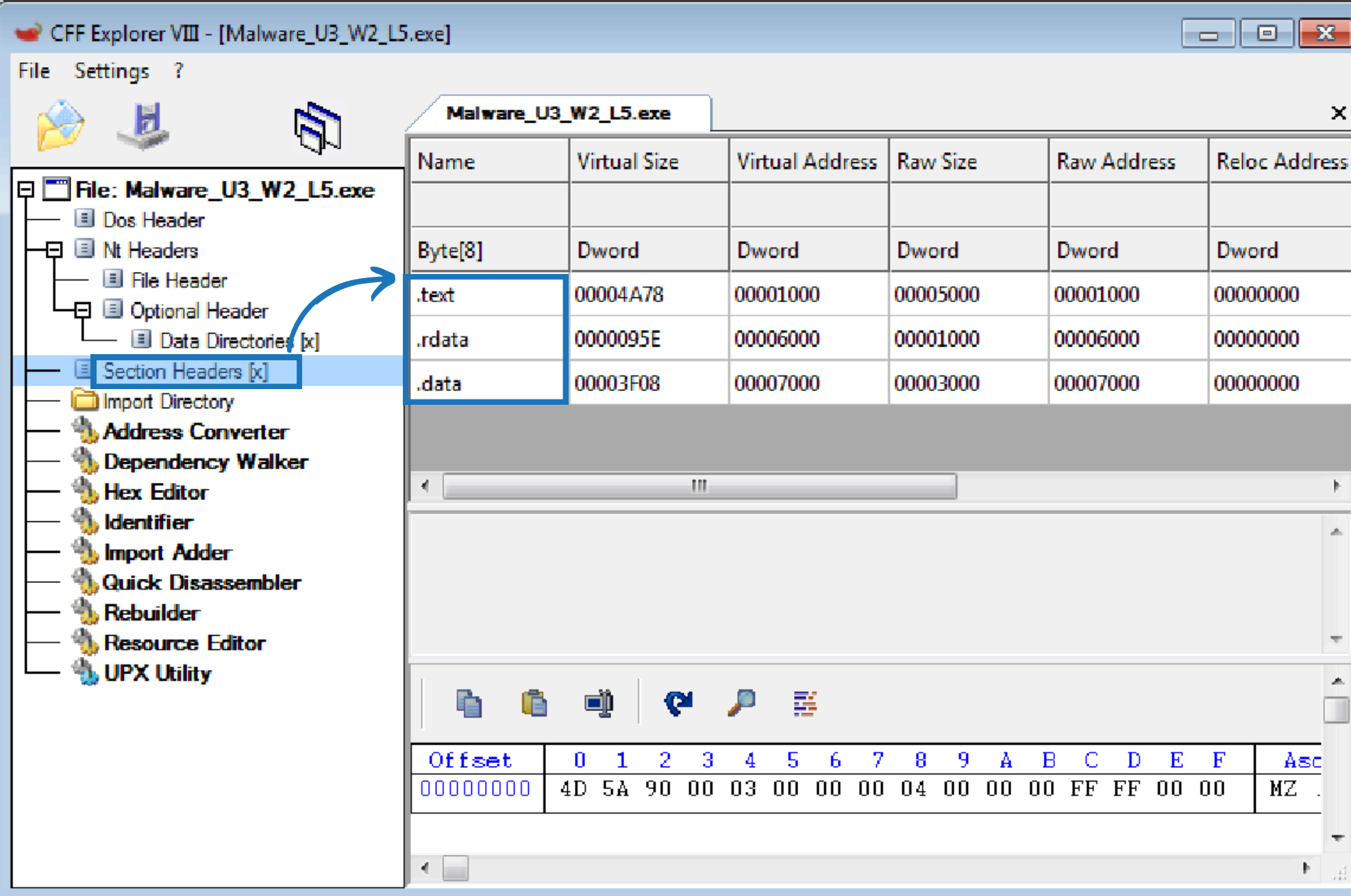
## SEZIONI

Ora ci spostiamo nella sezione “**Section Headers [x]**” e qui possiamo trovare le sezioni (blocchi di dati distinti che contengono vari tipi di informazioni necessarie per l'esecuzione del programma).

Sono state trovate 3 tipologie diverse:

- **.txt** : contiene il codice eseguibile del programma. Questa è la parte del file che contiene le istruzioni macchina che il processore esegue.
- **.rdata** : (read-only data) contiene dati costanti e altre informazioni di sola lettura che non devono essere modificate durante l'esecuzione del programma.
- **.data** : contiene dati globali e statici che il programma può leggere e scrivere durante l'esecuzione. Include variabili inizializzate

Nelle eseguibili e nelle librerie di collegamento dinamico (DLL) in Windows, queste sezioni sono parti importanti del formato del file, solitamente PE (Portable Executable), che organizzano il codice e i dati in modo strutturato.



CFF Explorer VIII - [Malware\_U3\_W2\_L5.exe]

File Settings ?

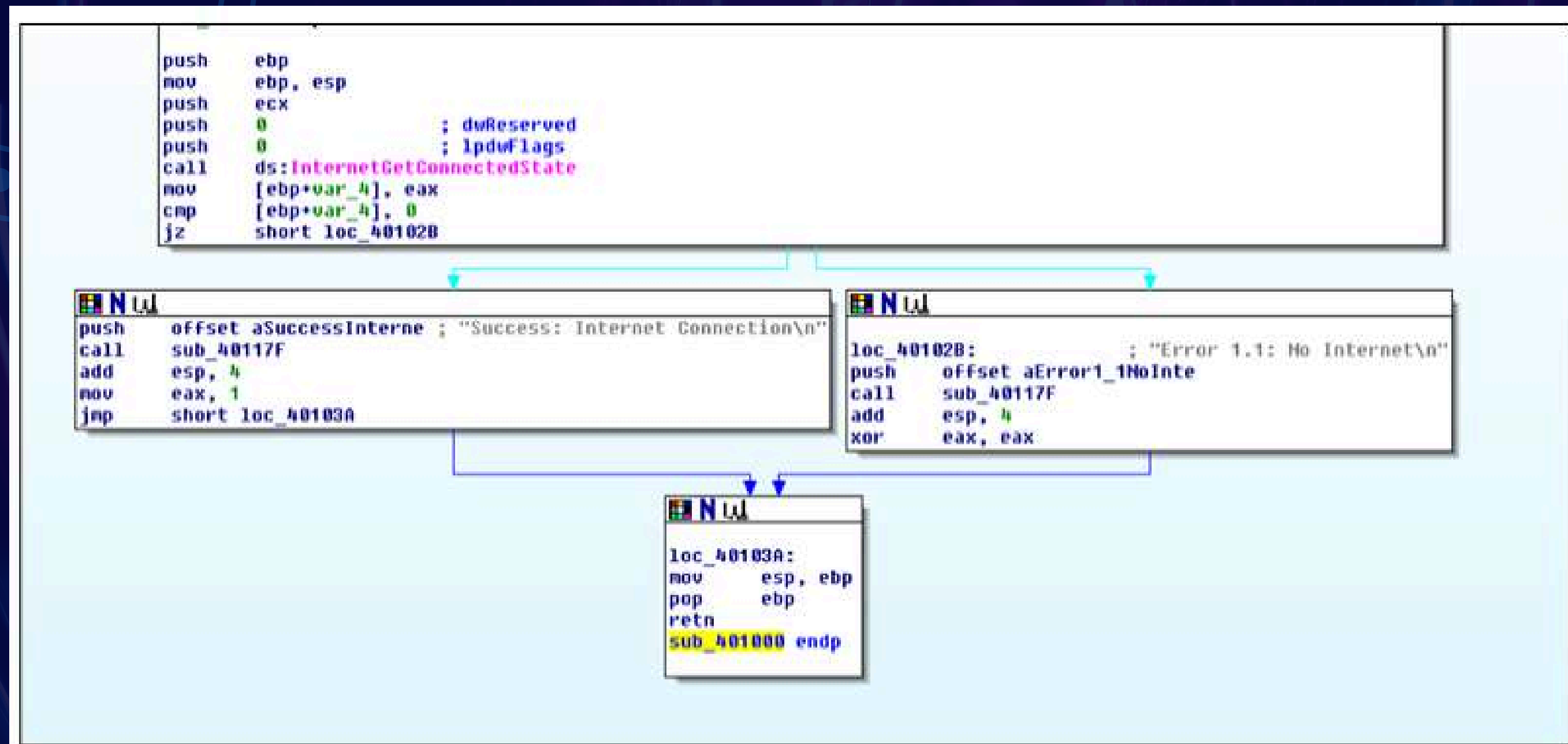
Malware\_U3\_W2\_L5.exe

Name	Virtual Size	Virtual Address	Raw Size	Raw Address	Reloc Address
Byte[8]	Dword	Dword	Dword	Dword	Dword
.text	00004A78	00001000	00005000	00001000	00000000
.rdata	0000095E	00006000	00001000	00006000	00000000
.data	00003F08	00007000	00003000	00007000	00000000

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

00000000 4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00 MZ

In riferimento alla figura sottostante sono state eseguite le successive analisi, consultabili nelle parti 3, 4, 5





# PARTE 3

COSTRUTTI NOTI



## 1. Creazione dello stack

```
push    ebp
mov     ebp, esp
```

## 2. Chiamata di funzione: I parametri sono passati sullo stack tramite le istruzioni push

La funzione viene chiamata "InternetGetConnectedState" per verificare lo stato della connessione internet.

```
push    0                ; dwReserved
push    0                ; lpdwFlags
call     ds:InternetGetConnectedState
```

## 3. Ciclo IF

(dove mov inizializza la funzione)

```
mov     [ebp+var_4], eax
cmp     [ebp+var_4], 0
jz      short loc_40102B
```



# PARTE 3

## COSTRUTTI NOTI



**4. Poi ci sono due percorsi che possono verificarsi** (è una chiamata di funzione nel momento in cui si verificano le condizioni di if, i parametri sono passati sullo stack tramite istruzioni push) :

```
1 push    offset aSuccessInterne ; "Success: Internet Connection\n"
  call    sub_40117F
  add     esp, 4
  mov     eax, 1
  jmp     short loc_40103A
```

Qui afferma che se c'è una connessione internet, viene eseguita la parte di codice che stampa "Success: Internet Connection"

```
2 loc_40102B:                ; "Error 1.1: No Internet\n"
  push    offset aError1_1NoInte
  call    sub_40117F
  add     esp, 4
  xor     eax, eax
```

Di contro se non c'è una connessione, viene eseguita la parte di codice che stampa "Error 1.1: No Internet"

## 5. Pulizia dello stack e restituzione della funzione

termina ripristinando il puntatore di base e restituendo al chiamante



```
loc_40103A:
mov     esp, ebp
pop     ebp
retn
sub_401000 endp
```



# PARTE 4

## FUNZIONALITA' IMPLEMENTATA

Il comportamento della funzionalità implementata è quello di verificare la connessione internet e di stampare un messaggio appropriato in base al risultato della verifica, ciò avviene attraverso la chiamata della funzione `internetgetconnectedstate` e ne controlla il valore di ritorno con «if». Se c'è una connessione internet, stampa "Success: Internet Connection" e ne darà valore uguale a 1. Se non c'è una connessione, stampa "Error 1.1: No Internet" con valore uguale a 0.

Infine il codice ripristina lo stack frame originale e restituisce il controllo al chiamante, con il valore di ritorno impostato a 1 o 0 in base allo stato della connessione.





# PARTE 5

SPIEGAZIONE RIGHE CODICE

**push ebp:** Salva il valore corrente del puntatore di base (EBP) sullo stack.

**mov ebp, esp:** Imposta il puntatore di base (EBP) al valore del puntatore dello stack (ESP)

**push ecx :** Salva il valore del registro ecx sullo stack

**push 0:** Passa il valore 0 come argomento (dwReserved) alla funzione InternetGetConnectedState

**push 0:** Passa il valore 0 come argomento (lpdwFlags) alla funzione InternetGetConnectedState

**call ds InternetGet ConnectedState:** Chiama la funzione InternetGetConnectedState per verificare lo stato della connessione Internet

**mov [ebp+var 4], eax:** Salva il valore di ritorno della funzione (stato della connessione) in una variabile locale

**cmp [ebp+var 4] 0:** Confronta il valore di ritorno con 0 per verificare se c'è connessione

**jz short loc\_40102B:** Se non c'è connessione (valore di ritorno 0), salta all'etichetta loc\_40102B



# PARTE 5

## SPIEGAZIONE RIGHE CODICE

**push offset aSuccesInterne ; "Succes: Internet Connection\n":** Carica l'indirizzo della stringa "Success: Internet Connection" nello stack

**call sub\_40117F:** Chiama una funzione per stampare il messaggio

**add esp, 4:** Ripristina lo stack pointer dopo la chiamata di funzione

**mov eax, 1:** Imposta EAX a 1, probabilmente per indicare successo

**jmp short loc\_40103A:** Salta alla fine della funzione

**loc\_40102B ; "Error 1.1: No Internet\n":** Etichetta in caso di mancata connessione

**push offset aError1\_1NoInte:** Carica l'indirizzo della stringa "Error 1.1: No Internet" nello stack

**call sub\_40117F:** Chiama una funzione per stampare il messaggio

**add esp, 4:** Ripristina lo stack pointer dopo la chiamata di funzione

**xor eax, eax:** Imposta EAX a 0, probabilmente per indicare errore



# PARTE 5

SPIEGAZIONE RIGHE CODICE

**loc\_40103A:** Etichetta per la fine della funzione

**mov esp, ebp:** Ripristina il puntatore dello stack al valore del puntatore di base

**pop ebp:** Ripristina il valore del puntatore di base (EBP)

**retn:** Restituisce il controllo al chiamante

**sub\_401000 endp:** fine della subroutine





# TRACCIA **BONUS**

Un giovane dipendente neo assunto segnala al reparto tecnico la presenza di un programma sospetto.

Il suo superiore gli dice di stare tranquillo ma lui non è soddisfatto e chiede supporto al SOC.

Il file "sospetto" è **iexplore.exe** contenuto nella cartella C:\Programmi\Internet Explorer (no, non ridete ragazzi)

Come membro senior del SOC ti è **richiesto di convincere il dipendente che il file non è maligno.**

Possono essere usati gli strumenti di analisi statica basica e/o analisi dinamica basica visti a lezione.

No disassembly no debug o similari

VirusTotal non basta, ovviamente non basta dire iexplorer è Microsoft quindi è buono, punto.



# BONUS

Per prima cosa ci siamo accertati che il percorso del file fosse: “ **C:\Programmi\Internet Explorer\iexplore.exe**” e abbiamo confermato che è questo.

Dopodiché abbiamo deciso di partite dall'*analisi statica basica* e abbiamo utilizzato per primo *Virus Total* per verificare se si tratta di un malware o meno controllando la sua reputazione in base ad un numero variabile di software antivirus, può essere usato per lo stesso scopo anche *Metadefender*. Abbiamo verificato la sua integrità caricando il file e non abbiamo notato nulla che ci possa far pensare ad un malware.

ⓘ File distributed by Microsoft

Reanalyze Similar More

cfa888e71c65a8807cd719a19c211d1a5dcc04b36d2ebe2d94bf17971ec22690


IEXPLORE.EXE

Size

678.77 KB

Last Analysis Date

2 months ago



EXE

peexe

direct-cpu-clock-access

assembly

trusted

overlay

via-tor

known-distributor

runtime-modules

64bits

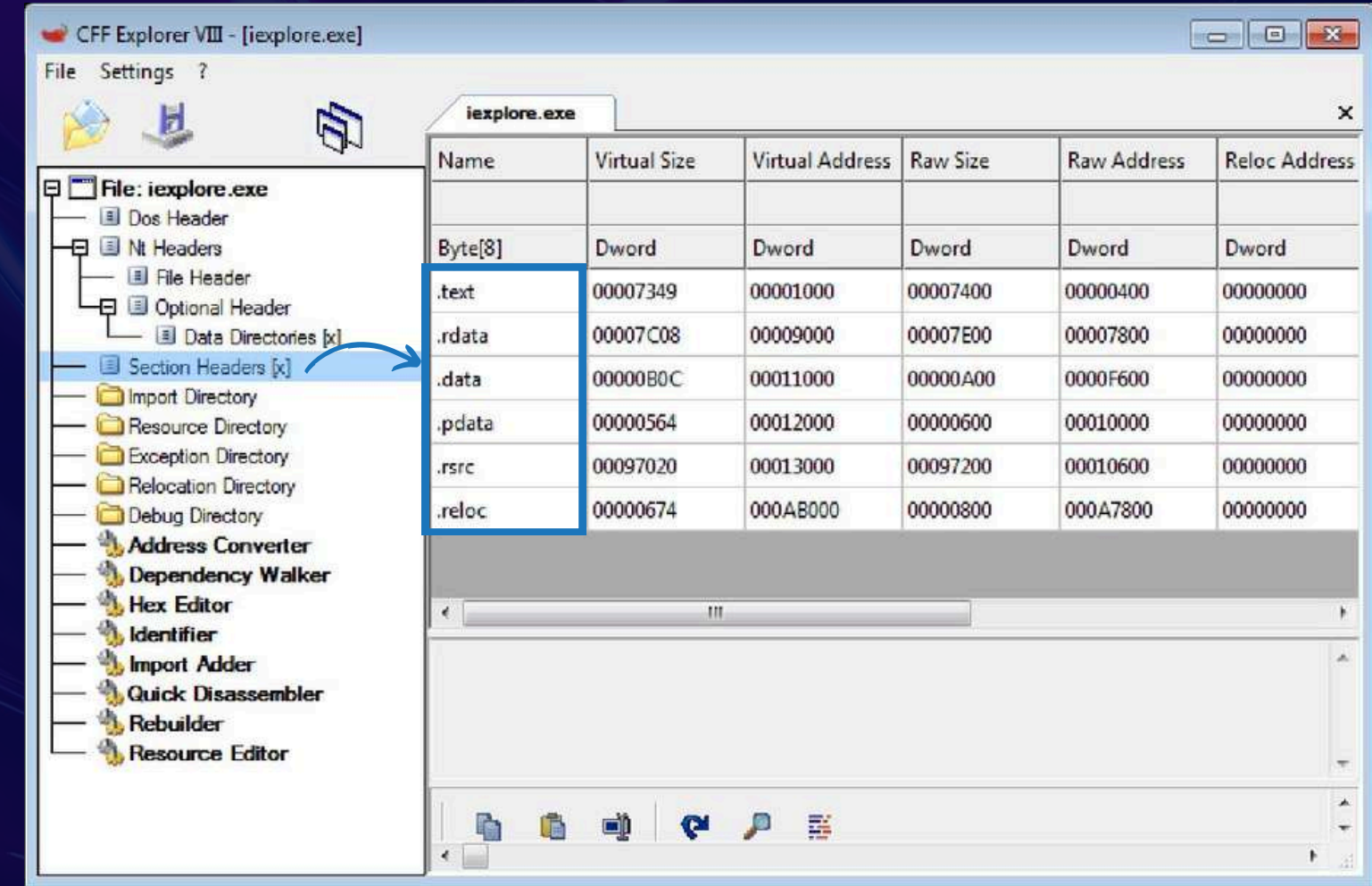
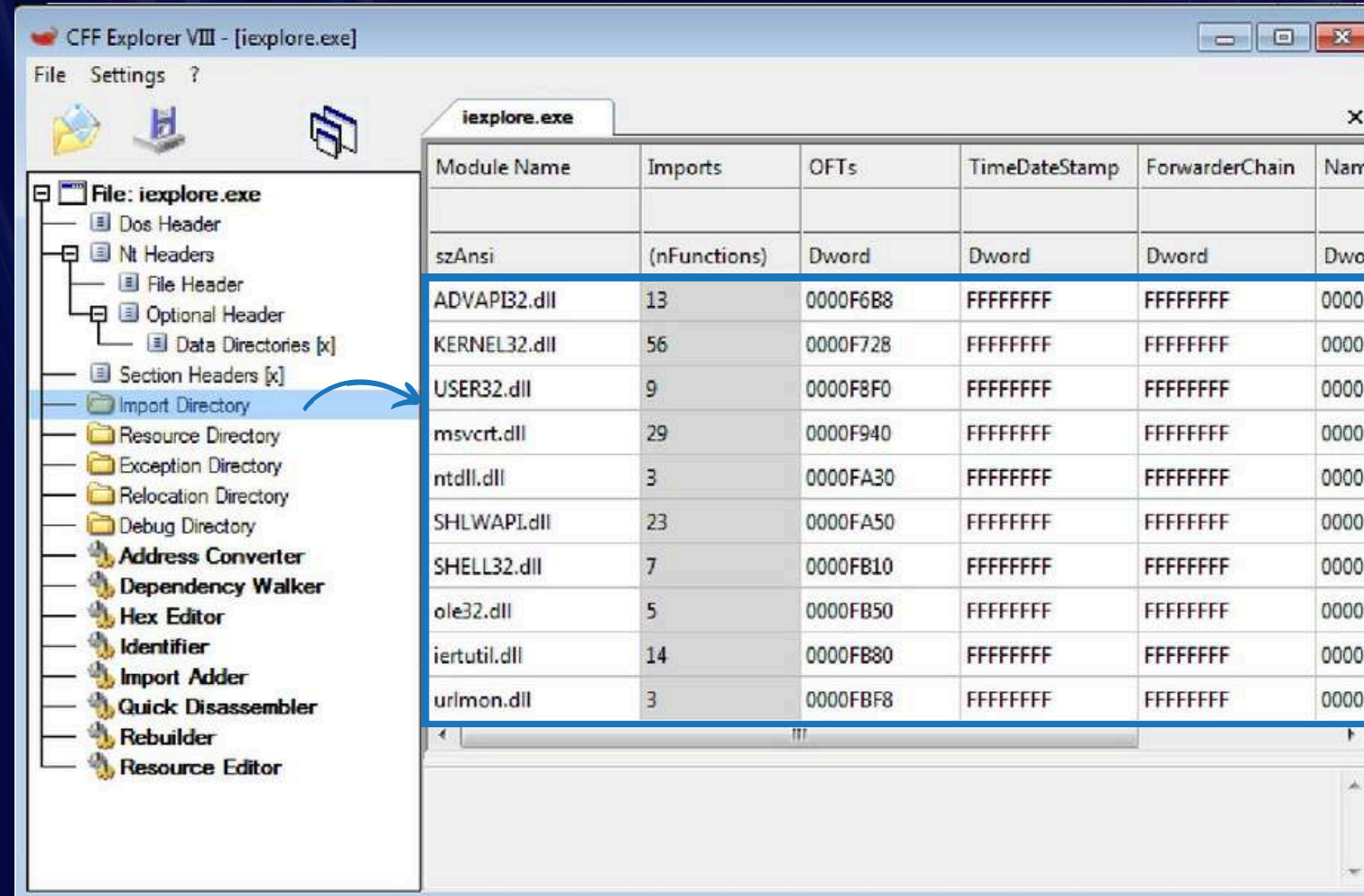
signed

AhnLab-V3	✔ Undetected	Alibaba	✔ Undetected
AliCloud	✔ Undetected	ALYac	✔ Undetected
Antiy-AVL	✔ Undetected	Arcabit	✔ Undetected
Avast	✔ Undetected	AVG	✔ Undetected



# BONUS

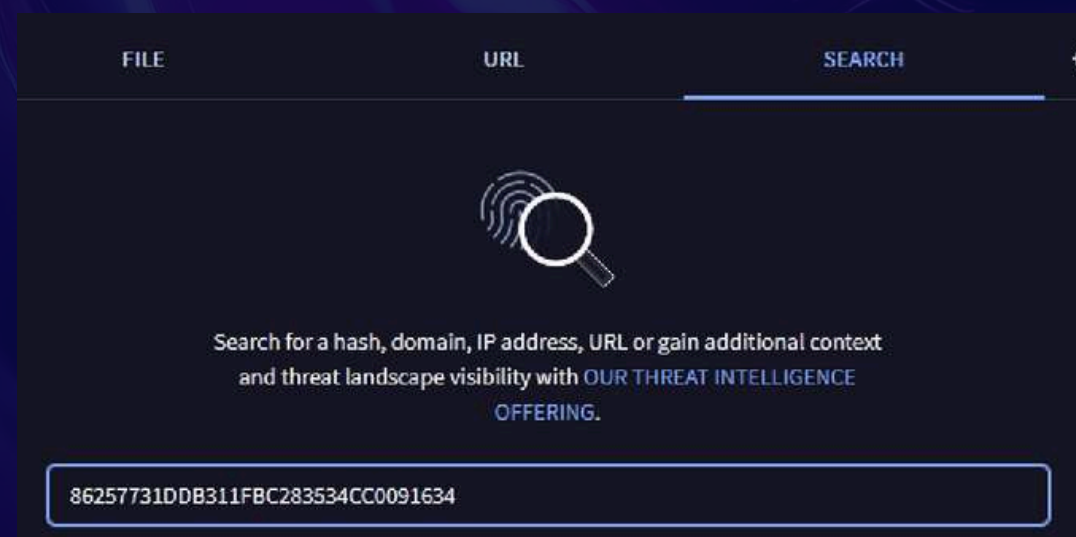
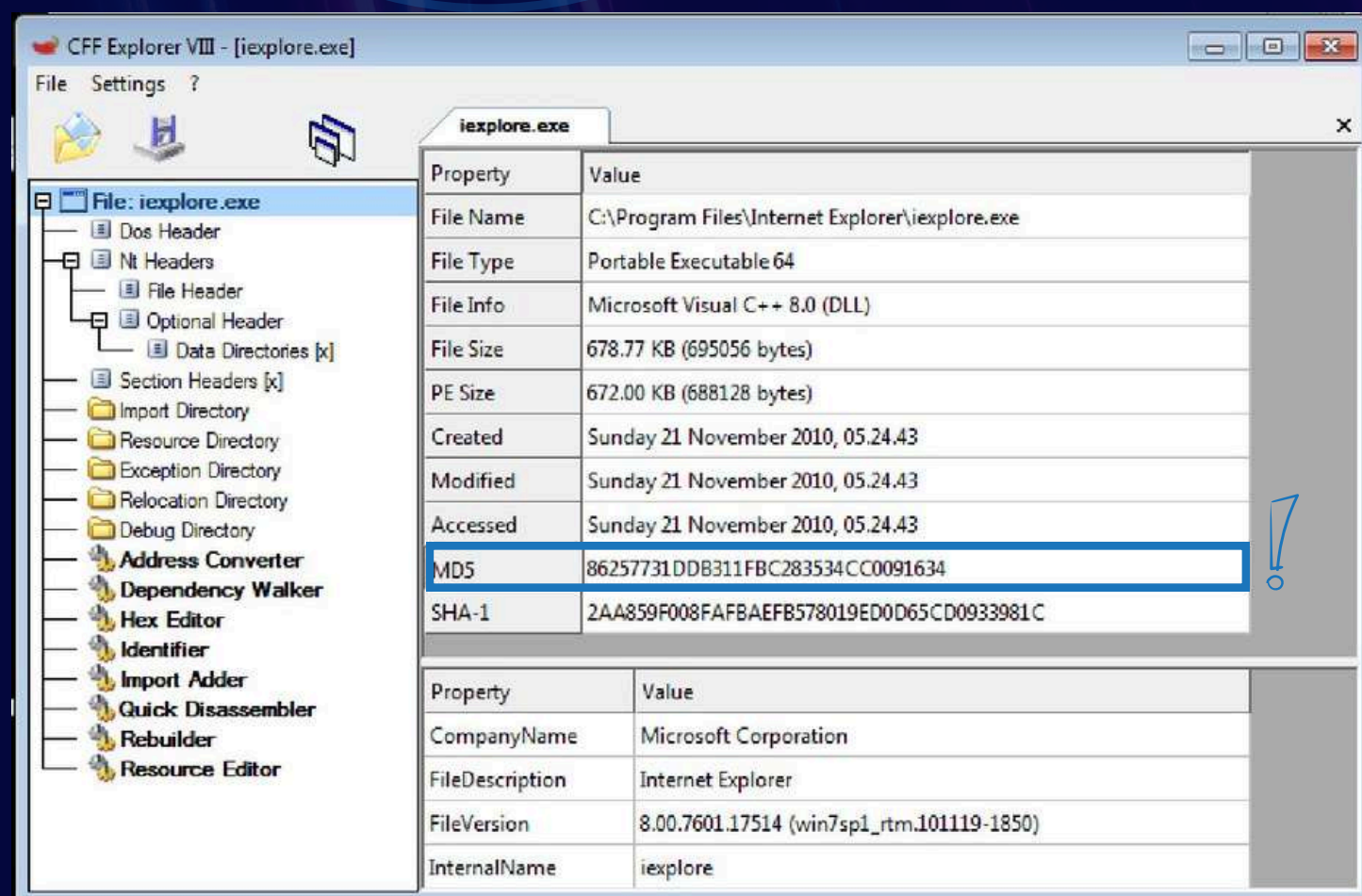
Inoltre per effettuare una scansione più approfondita ci siamo concentrati su **CFF Explorer**, utile per controllare le funzioni importate ed esportate da un malware. Qui possiamo vedere **le librerie e le sezioni** che il programma utilizza e abbiamo notato che possiamo vedere una quantità massiccia di librerie, ciò indica che possa trattarsi di un file legittimo in quanto solitamente i malware utilizzano tecniche per rendere meno evidente la loro presenza o per ingannare gli strumenti di analisi, nonostante ciò questo non esclude che non si tratti di un malware.





# BONUS

Sempre tramite CFF Explorer abbiamo ottenuto il codice **hash** ed lo abbiamo inserito in Virus total per per verificare se i file sono stati alterati e abbiamo visto che risulta essere tutto nella norma anche in questo caso. Possiamo generare hash anche con *Md5deep* per singoli file o per tutti i file all'interno di una directory. Questo è utile per creare un'impronta digitale univoca di ciascun file, che può essere utilizzata per verificare l'integrità del file in futuro.



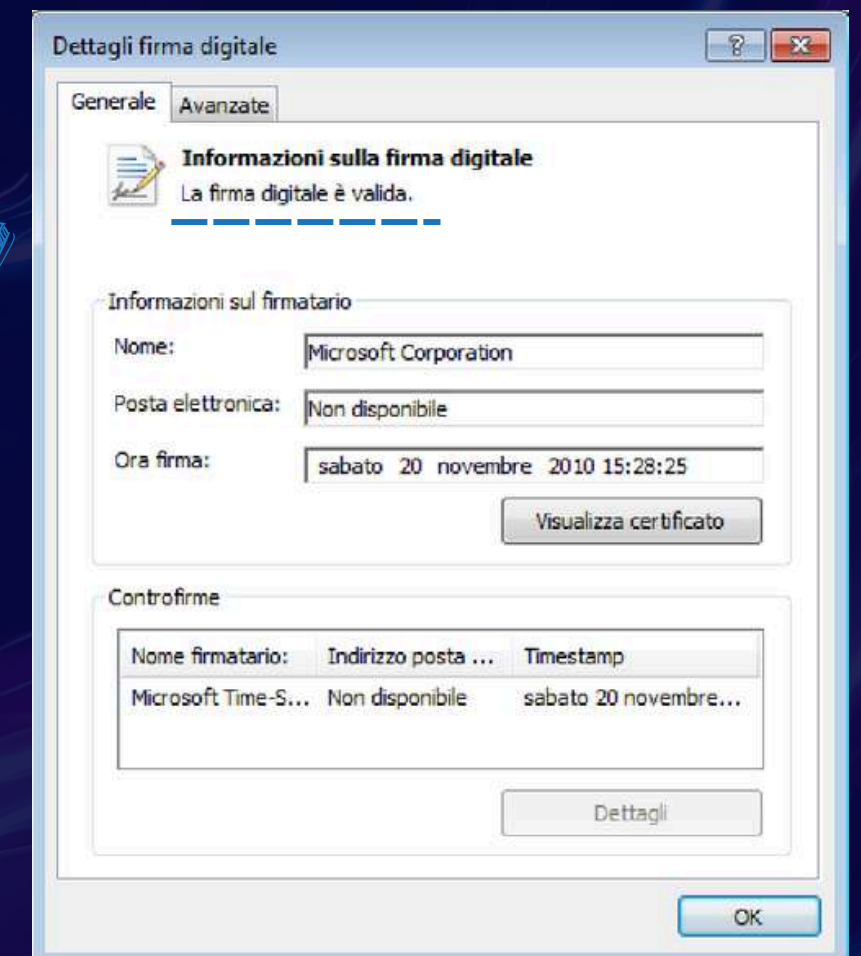
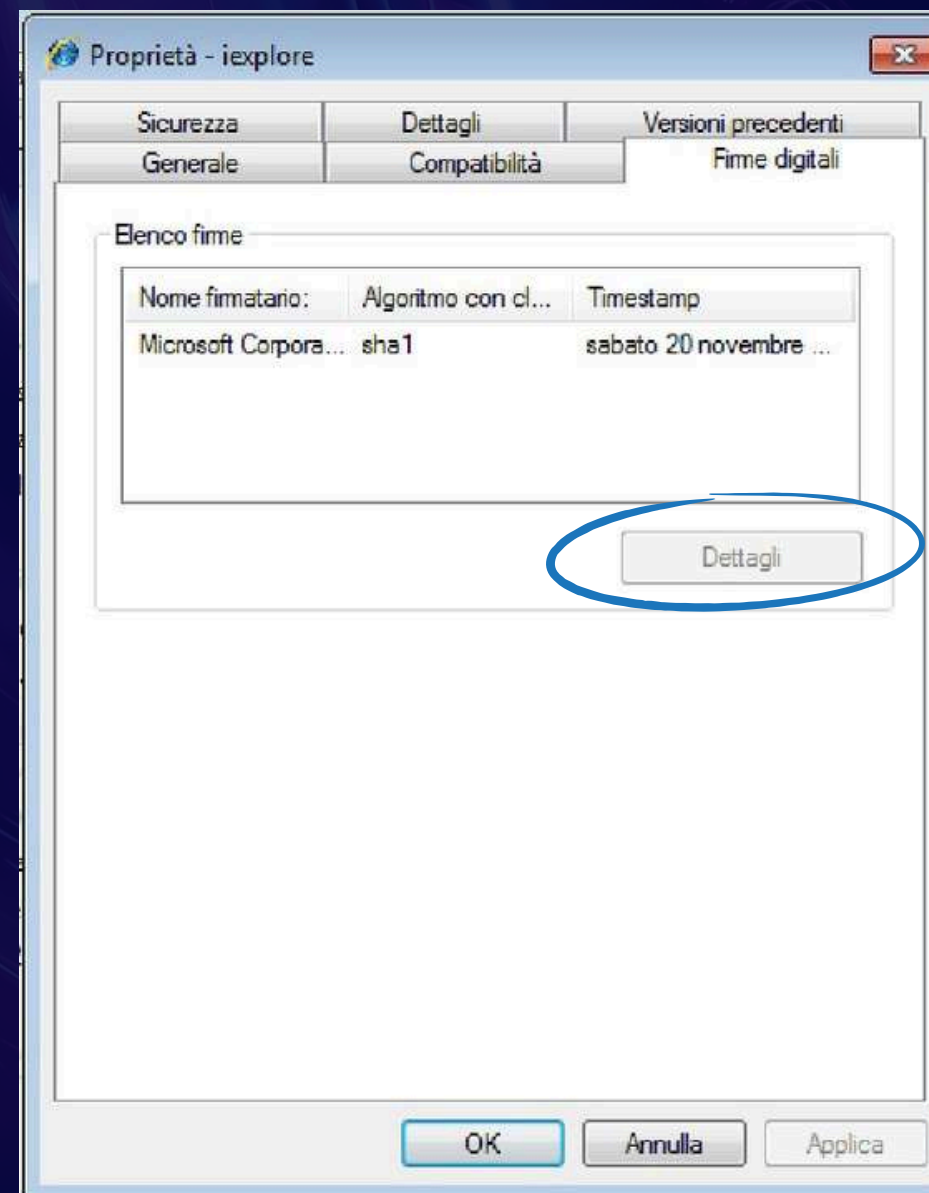
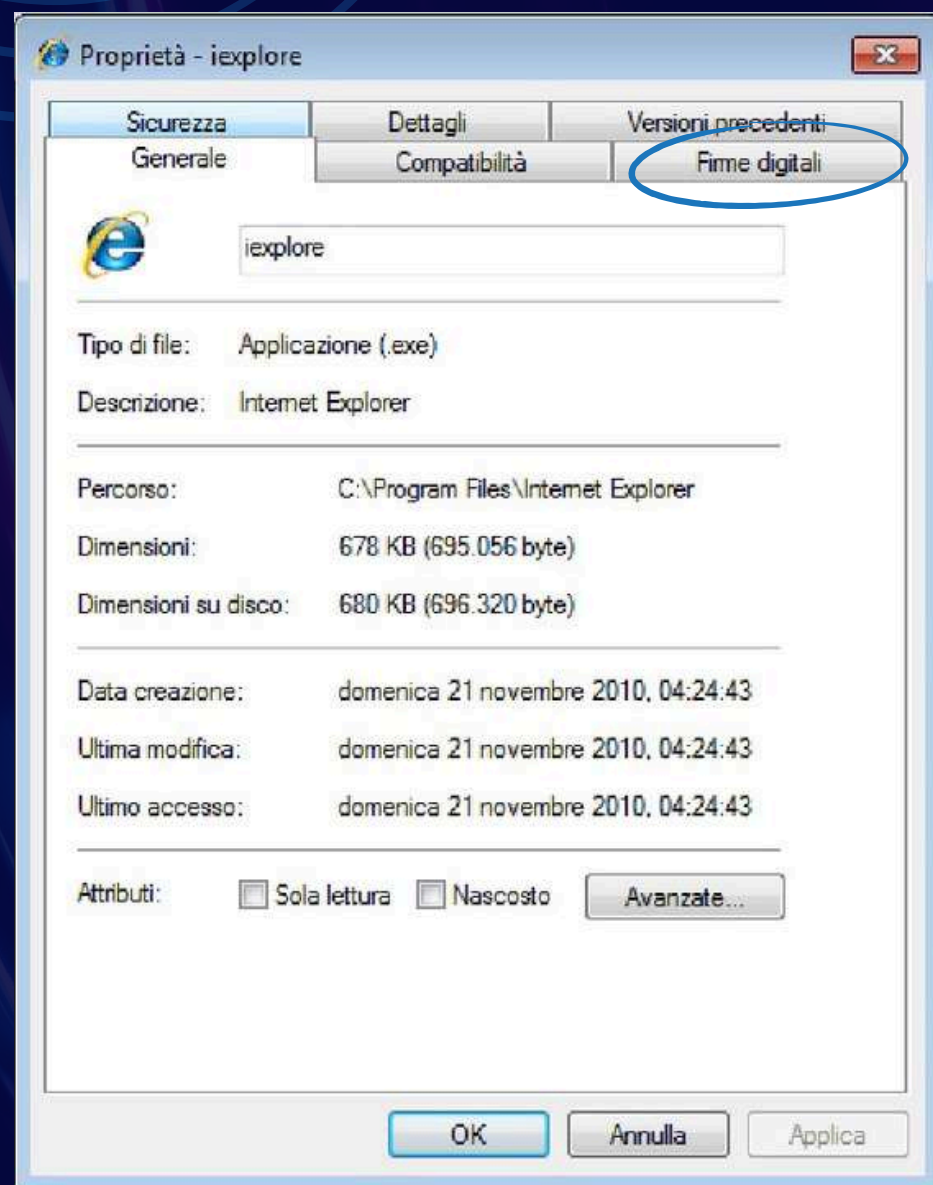
AhnLab-V3	Undetected	Alibaba	Undetected
AliCloud	Undetected	ALYac	Undetected
Antiy-AVL	Undetected	Arcabit	Undetected
Avast	Undetected	AVG	Undetected
Avira (no cloud)	Undetected	Baidu	Undetected



# BONUS

Altro metodo per dimostrare che si tratta di un file legittimo e non di un malware è verificare la **firma digitale** e garantire la sua autenticità e sicurezza.

Quindi seguendo lo stesso percorso nella cartella programmi cliccare tasto dx sul file iexplore e andare su proprietà, qui possiamo vedere una scheda chiamata "Firma digitale", poi cliccare sulla firma che troviamo e andare su dettagli qui troveremo "la firma digitale è valida" e perciò possiamo affermare che il file è firmato digitalmente.

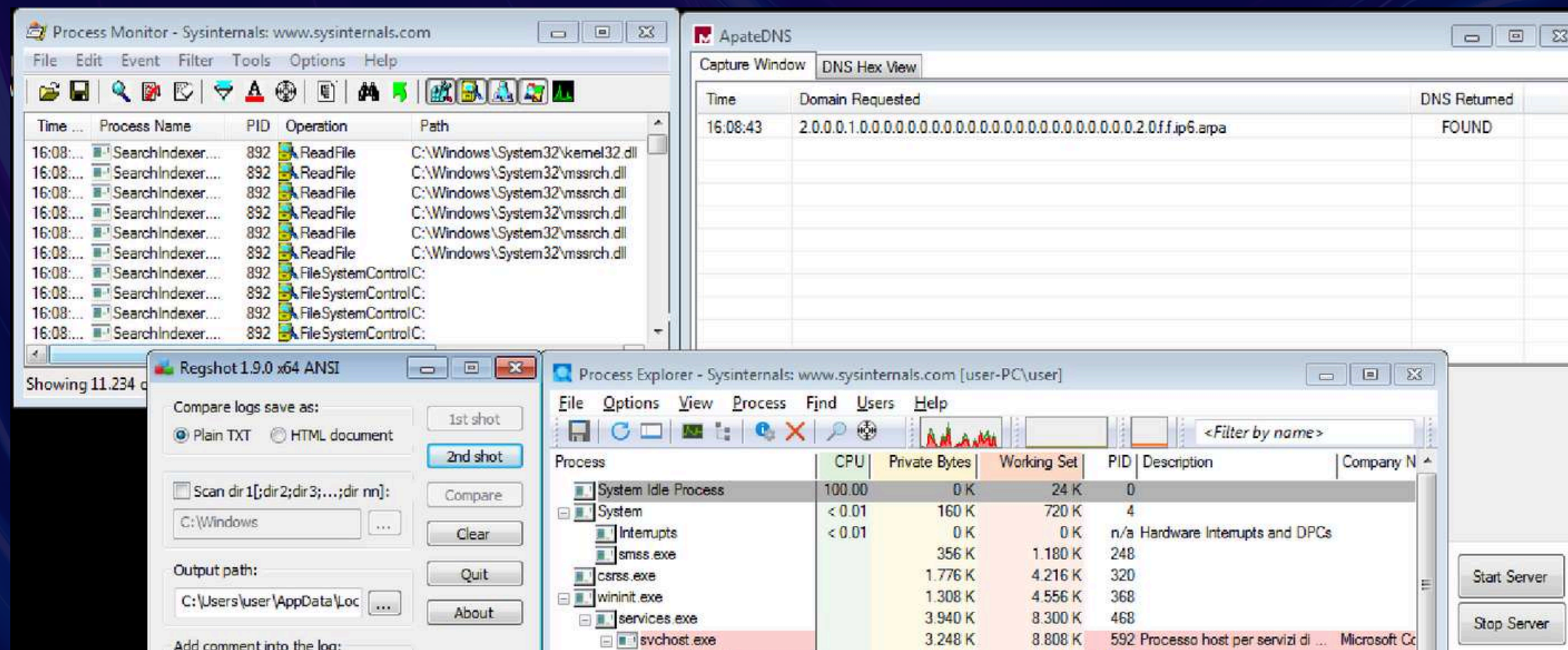




# BONUS

Dopo aver effettuato l'analisi statica siamo passati all'**analisi dinamica basica** per capire cosa fa realmente il file quando viene eseguito, per prima cosa però ci assicuriamo che la rete sia impostata su **Rete interna**, i passaggi che abbiamo seguito per la preparazione dell'ambiente sono i seguenti:

1. E' stato avviato **Process Explorer** (Fornisce una panoramica dettagliata di tutti i processi attualmente in esecuzione nel sistema)
2. Poi il server **ApateDNS** (Permette di alterare le risposte DNS per reindirizzare il traffico di rete. Questo è utile per testare e analizzare il comportamento del software)
3. Aperto **Regshot** e salvata una prima istantanea (permette di creare snapshot dello stato attuale del registro di sistema in due momenti diversi)
4. Aperto **Procomon** (consente di monitorare in tempo reale le attività del file system, del registro di sistema e dei processi)

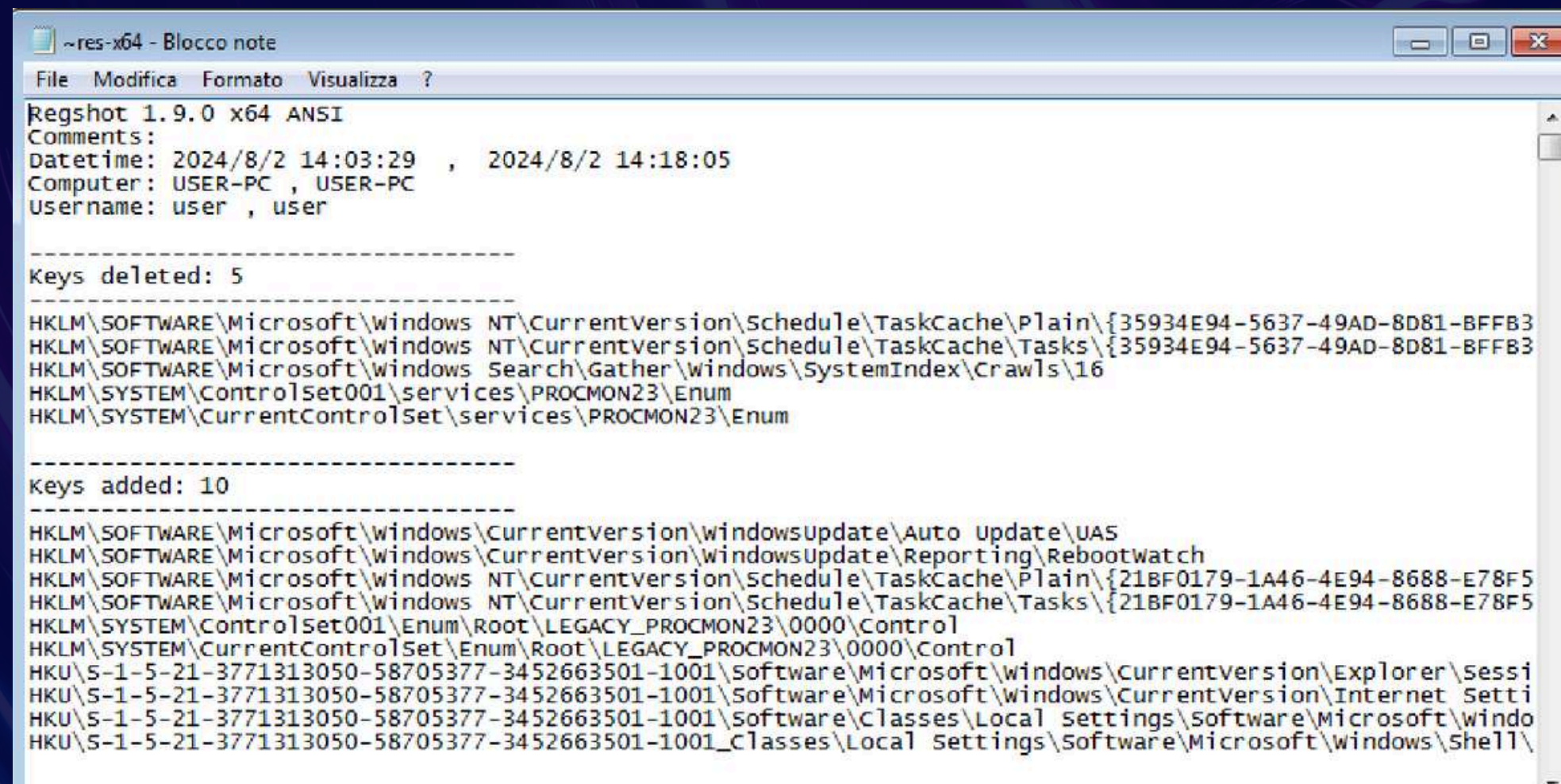




# BONUS

Poi abbiamo *avviato il file iexplore.exe* e osservato nei vari tool che abbiamo attivato in precedenza quali sono stati i suoi comportamenti:

- abbiamo salvato una seconda istantanea di Regshot e poi comparato il primo con il secondo anche se sono state aggiunte, modificate ed eliminate alcune chiavi, queste non sono riconducibili ad un malware ma rappresentano il normale andamento del file.



```
regshot 1.9.0 x64 ANSI
Comments:
Datetime: 2024/8/2 14:03:29 , 2024/8/2 14:18:05
Computer: USER-PC , USER-PC
Username: user , user

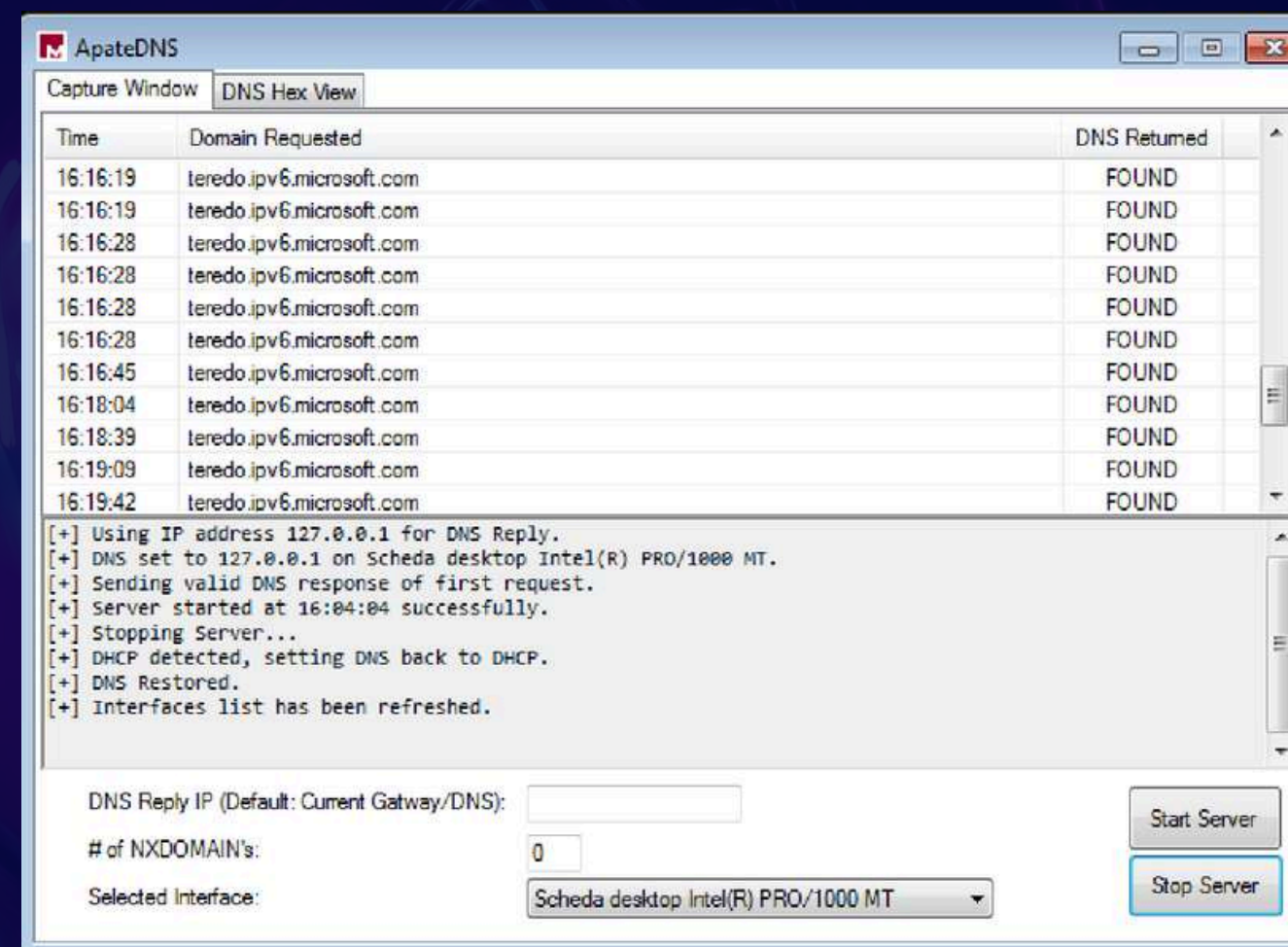
-----
Keys deleted: 5
-----
HKLM\SOFTWARE\Microsoft\windows NT\CurrentVersion\Schedule\TaskCache\Plain\{35934E94-5637-49AD-8D81-BFFB3
HKLM\SOFTWARE\Microsoft\windows NT\CurrentVersion\Schedule\TaskCache\Tasks\{35934E94-5637-49AD-8D81-BFFB3
HKLM\SOFTWARE\Microsoft\windows Search\Gather\windows\SystemIndex\Crawls\16
HKLM\SYSTEM\ControlSet001\services\PROCMON23\Enum
HKLM\SYSTEM\CurrentControlSet\services\PROCMON23\Enum

-----
Keys added: 10
-----
HKLM\SOFTWARE\Microsoft\windows\CurrentVersion\windowsupdate\Auto update\UAS
HKLM\SOFTWARE\Microsoft\windows\CurrentVersion\windowsupdate\Reporting\Rebootwatch
HKLM\SOFTWARE\Microsoft\windows NT\CurrentVersion\Schedule\TaskCache\Plain\{21BF0179-1A46-4E94-8688-E78F5
HKLM\SOFTWARE\Microsoft\windows NT\CurrentVersion\Schedule\TaskCache\Tasks\{21BF0179-1A46-4E94-8688-E78F5
HKLM\SYSTEM\ControlSet001\Enum\Root\LEGACY_PROCMON23\0000\Control
HKLM\SYSTEM\CurrentControlSet\Enum\Root\LEGACY_PROCMON23\0000\Control
HKU\S-1-5-21-3771313050-58705377-3452663501-1001\Software\Microsoft\windows\CurrentVersion\Explorer\sessi
HKU\S-1-5-21-3771313050-58705377-3452663501-1001\Software\Microsoft\windows\CurrentVersion\Internet Setti
HKU\S-1-5-21-3771313050-58705377-3452663501-1001\Software\Classes\Local Settings\Software\Microsoft\windo
HKU\S-1-5-21-3771313050-58705377-3452663501-1001\Classes\Local Settings\Software\Microsoft\windows\Shell\
```



# BONUS

- fermato il Server DNS e osservato che non ci sono attività sospette
- fermato Process Explorer e osservato che il file non ha tentato di creare altri file
- filtrato su Procmon iexplore per vedere il suo funzionamento e vista la quantità di file presenti sono stati effettuati altri test ma hanno confermato che **non si tratta di un malware**.



Procmon64.exe	< 0.01	35.304 K	42.368 K	2796	System Idle
iexplore.exe	< 0.01	7.776 K	25.140 K	1916	Internet Explorer
iexplore.exe	< 0.01	9.364 K	28.652 K	600	Internet Explorer

16:12:...	iexplore.exe	1916	Process Start	SUCCESS	Parent PID: 1524, ...
16:12:...	iexplore.exe	1916	Thread Create	SUCCESS	Thread ID: 1640
16:12:...	iexplore.exe	1916	Load Image	SUCCESS	Image Base: 0x12b...
16:12:...	iexplore.exe	1916	Load Image	SUCCESS	Image Base: 0x777...
16:12:...	iexplore.exe	1916	CreateFile	NAME NOT FOUND	Desired Access: G...



# BONUS

## CONCLUSIONE

Dopo aver completato un'analisi statica e dinamica di base sul file `iexplore.exe`, possiamo concludere che il file in questione non presenta caratteristiche tipiche di un malware. L'analisi statica ha confermato che il file non contiene codici o comportamenti sospetti all'interno della sua struttura, mentre l'analisi dinamica ha mostrato che il file si comporta in modo conforme alle aspettative di un'applicazione legittima di Internet Explorer durante l'esecuzione. Entrambi i metodi di analisi hanno contribuito a confermare l'integrità e la sicurezza del file

