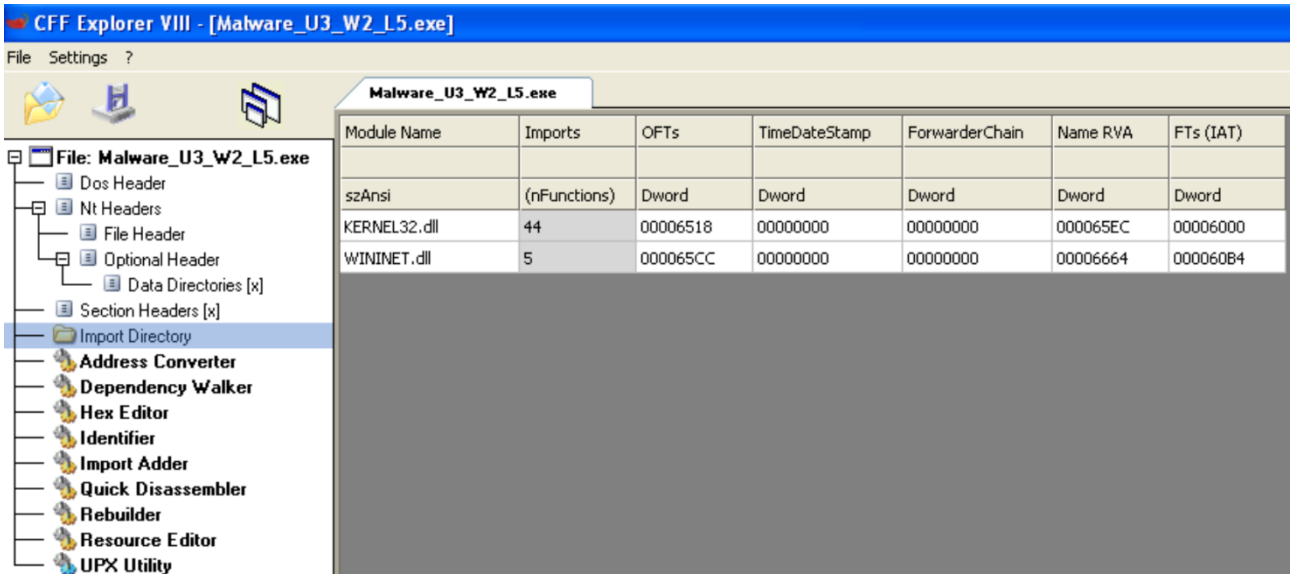
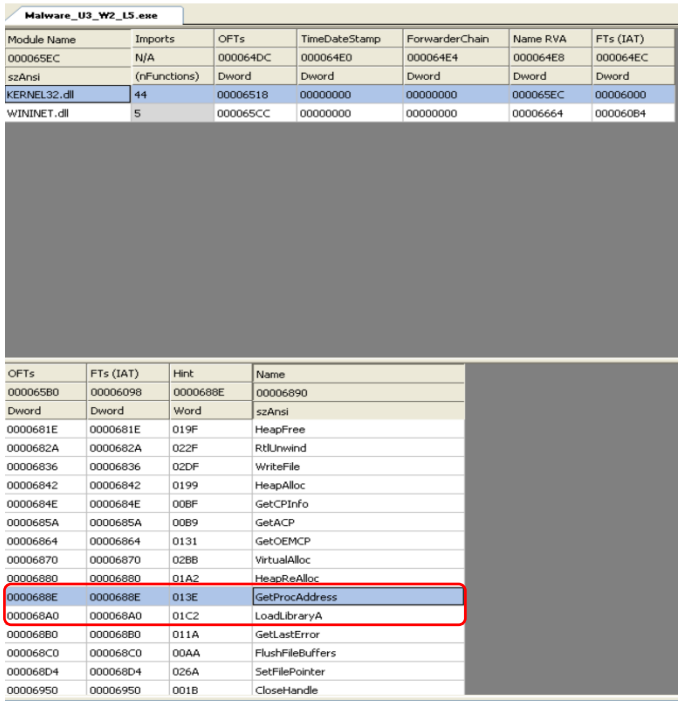
ANALISI STATICA E DINAMICA DEL MALWARE

Nel seguente progetto si procederà con l'analisi di un malware sulla macchina virtuale Windows XP. Essa può essere effettuata in due modi: staticamente e dinamicamente dove la seconda è una complementazione della prima. Inoltre entrambe vengono suddivise a loro volta in forma basica e avanzata.

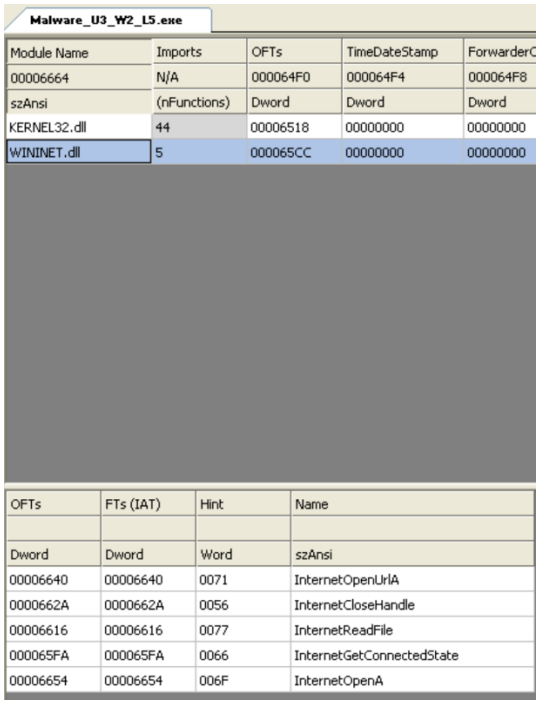
Viene richiesto ora di eseguire la fase iniziale dell'analisi attraverso una statica basica che permette di ricavare alcune importanti informazioni senza eseguire il codice.



A tale scopo si può utilizzare il tool CFF Explorer che consente di visualizzare le librerie importate o esportate da un file eseguibile. Come si vede nella figura sopra il malware importa due tipi di librerie ovvero *kernel32.dll* e *wininet.dll*. La prima permette all'eseguibile di interagire con il sistema operativo e manipolare i file e presenta nel caso specifico 44 funzioni, la seconda implementa protocolli di rete (http, ftp, ntp…) e presenta 5 funzioni.



Esse vengono elencate selezionando la libreria: si può notare cosi che per *kernel32.dll* alcune agiscono sulla gestione della memoria o scrivendo file altre richiamano la libreria run-time (come evidenziato nel rettangolo rosso) ovvero durante l'esecuzione del programma e quindi solo quando la funzione specifica deve essere utilizzata. Ciò è molto importante per l'analisi perché fa capire che il malware cerca di rendersi invisibile e meno invasivo.

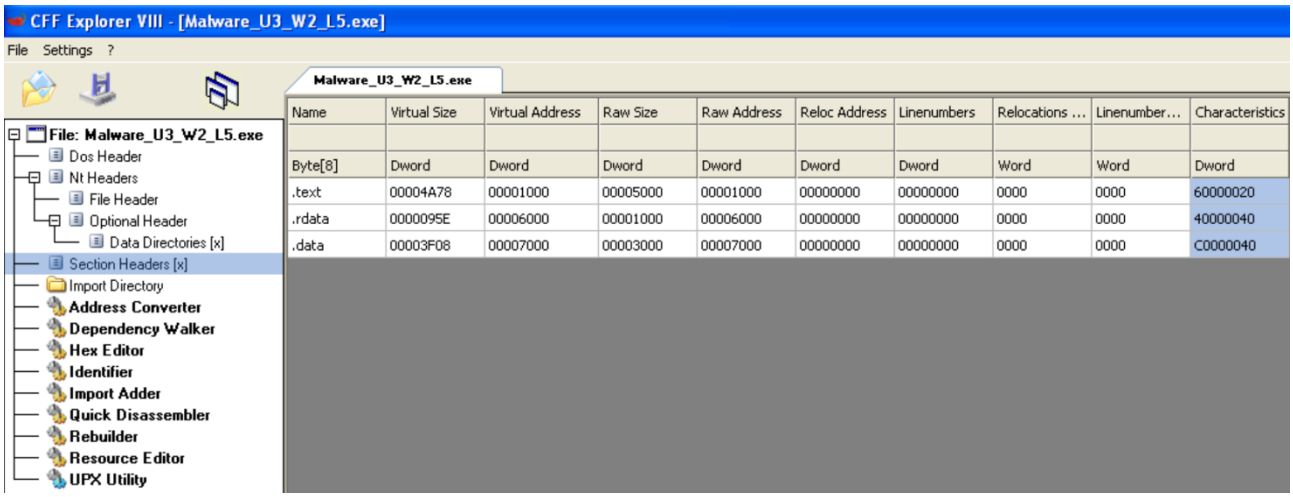


Malware_U3_W2_L5.exe

Module Name	Imports	OFTs	TimeDateStamp	ForwarderC
00006664	N/A	000064F0	000064F4	000064F8
szAnsi	(nFunctions)	Dword	Dword	Dword
KERNEL32.dll	44	00006518	00000000	00000000
WININET.dll	5	000065CC	00000000	00000000

OFTs	FTs (IAT)	Hint	Name
Dword	Dword	Word	szAnsi
00006640	00006640	0071	InternetOpenUrlA
0000662A	0000662A	0056	InternetCloseHandle
00006616	00006616	0077	InternetReadFile
000065FA	000065FA	0066	InternetGetConnectedState
00006654	00006654	006F	InternetOpenA

Con le funzioni di *wininet.dll* il malware cerca di aprire una connessione e accedere ad un URL e tra le varie ipotesi il virus potrebbe in questo modo scaricare ulteriori eseguibili malevoli (downloader).



CFF Explorer VIII - [Malware_U3_W2_L5.exe]

File Settings ?

Malware_U3_W2_L5.exe

File: Malware_U3_W2_L5.exe
- Dos Header
- Nt Headers
  - File Header
  - Optional Header
    - Data Directories [x]
- Section Headers [x]
- Import 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	Relocations ...	Linenumber...	Characteristics
Byte[8]	Dword	Dword	Dword	Dword	Dword	Dword	Word	Word	Dword
.text	00004A78	00001000	00005000	00001000	00000000	00000000	0000	0000	60000020
.rdata	0000095E	00006000	00001000	00006000	00000000	00000000	0000	0000	40000040
.data	00003F08	00007000	00003000	00007000	00000000	00000000	0000	0000	C0000040

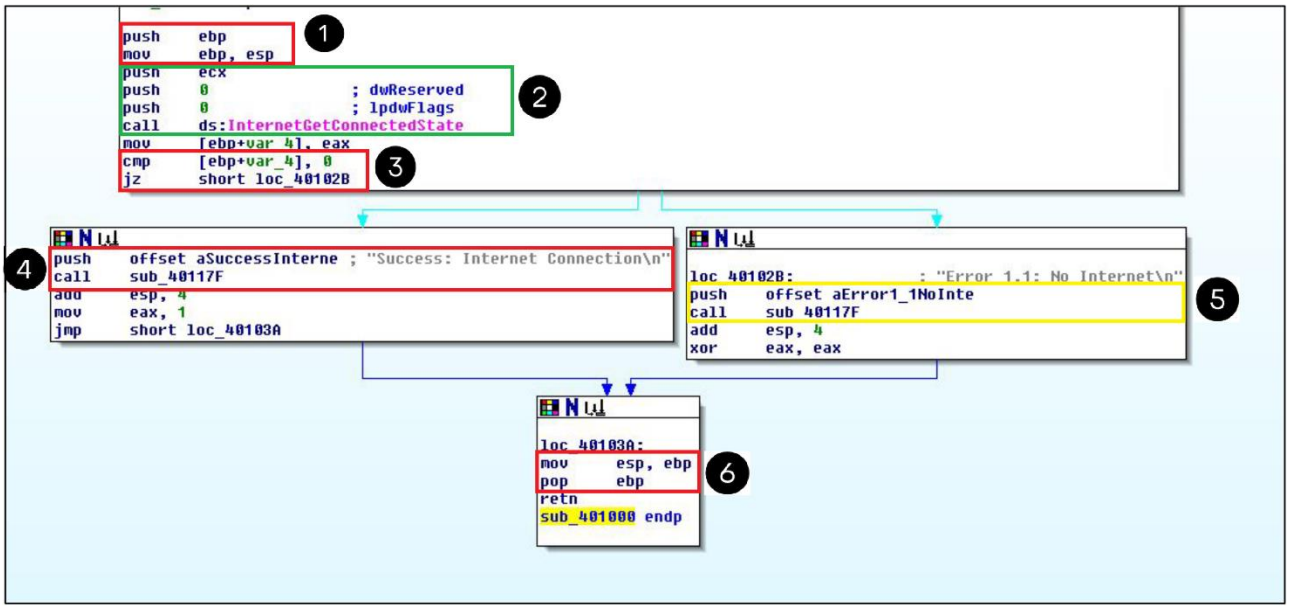
Nello screen riportato sopra invece possiamo identificare le sezioni dell'headers cliccando sull'apposita opzione a sinistra:

- *.txt* fa riferimento alle righe di codice che vengono eseguite dalla CPU
- *.rdata* corrisponde alle informazioni delle librerie e di conseguenza dell'import/export delle funzioni

- *.data* identifica la parte di memoria in cui vengono salvate le variabili globali (ovvero quelle dichiarate globalmente e che quindi non hanno bisogno di essere riportate nel contesto di una funzione specifica)

Si possono visualizzare ulteriori info sullo spazio occupato dalla sezione, nel dettaglio *Virtual Size* durante il caricamento dell'eseguibile (RAM virtuale) e *RAW Size* per lo spazio su disco.

Fatto ciò ci viene richiesto di riconoscere i costrutti di un codice malevolo tradotto in assembly riportato di seguito. Si ricorda che quest'ultimo è un linguaggio di basso livello che astrae il linguaggio macchina in cui sono riportate le istruzioni in binario che esegue la CPU attraverso dei tool (disassembler) per renderlo più comprensibile all'uomo. Attraverso la *reverse engineering* invece si astrae l'assembly ad un livello superiore.



1. Il costrutto evidenzia la creazione dello stack (sezione della RAM) dove vengono salvate le variabili locali e i parametri delle funzioni. Ciò avviene mediante i registri *ebp, esp* che puntano rispettivamente alla base e alla cima. Lo stack puo essere immaginato come una pila di piatti che funziona secondo il modello LIFO ovvero l'ultimo piatto inserito (*push*) è il primo ad essere rimosso (*pop*)
2. Rappresenta la chiamata di funzione attraverso l'istruzione *call* in cui il malware tenta una connessione a internet.
3. Indica un ciclo for attraverso l'istruzione condizionale *cmp* e il salto condizionale *jz* che appunto salta alla locazione specificata se lo ZF è 0
4. Viene effettuato un "printf" se la connessione internet viene stabilita con successo
5. Viene indicato al contrario un "else" se paragonato al linguaggio C in cui non viene stabilita alcuna connessione.
6. Pulizia dello stack: una volta terminato il proprio compito, lo stack della funzione chiamata e le sue variabili vengono rimosse. Precisamente se questa operazione avviene per opera dello stack della funzione chiamata allora si parla di *stdcall* se al contrario è la funzione chiamante che elimina lo stack della funzione chiamata si parla di *cdecl.* In entrambi i casi i

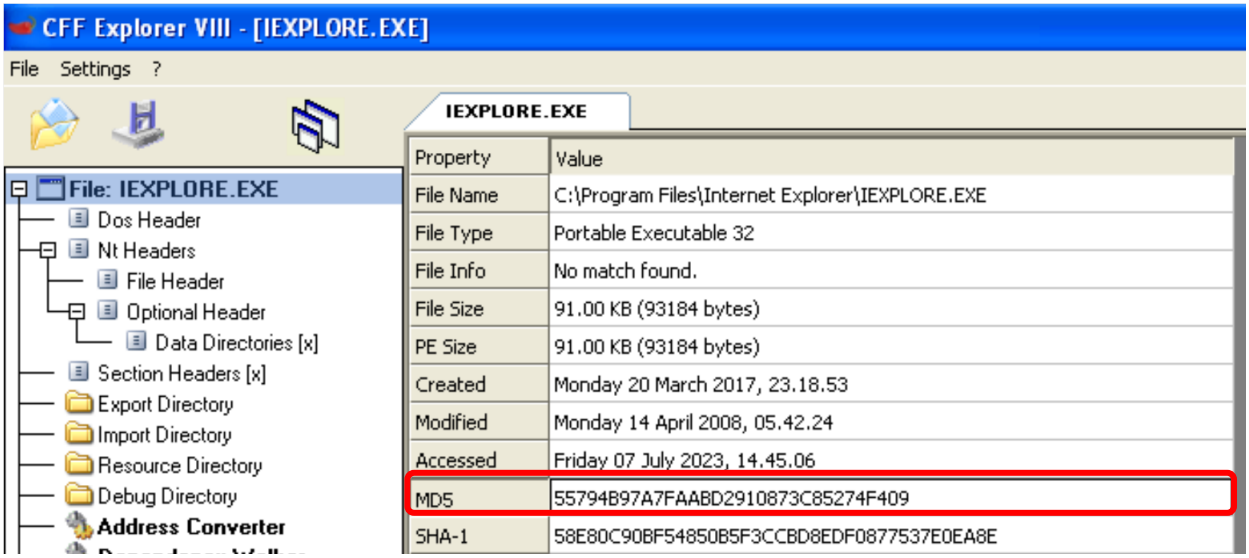
parametri vengono passati alla funzione chiamata sullo stack a differenza del metodo *fastcall* in cui ciò avviene sui registri.

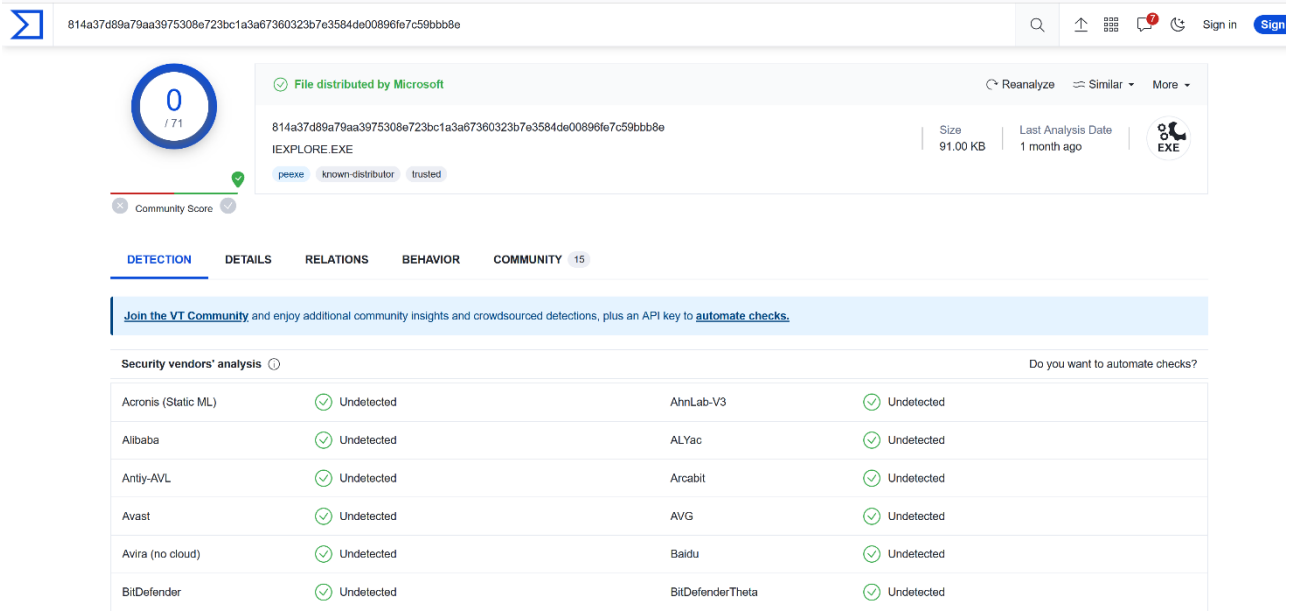
BONUS

Viene richiesto in questo caso l'analisi di un file eseguibile iexplore.exe ritenuto sospetto da un dipendente assunto da poco.

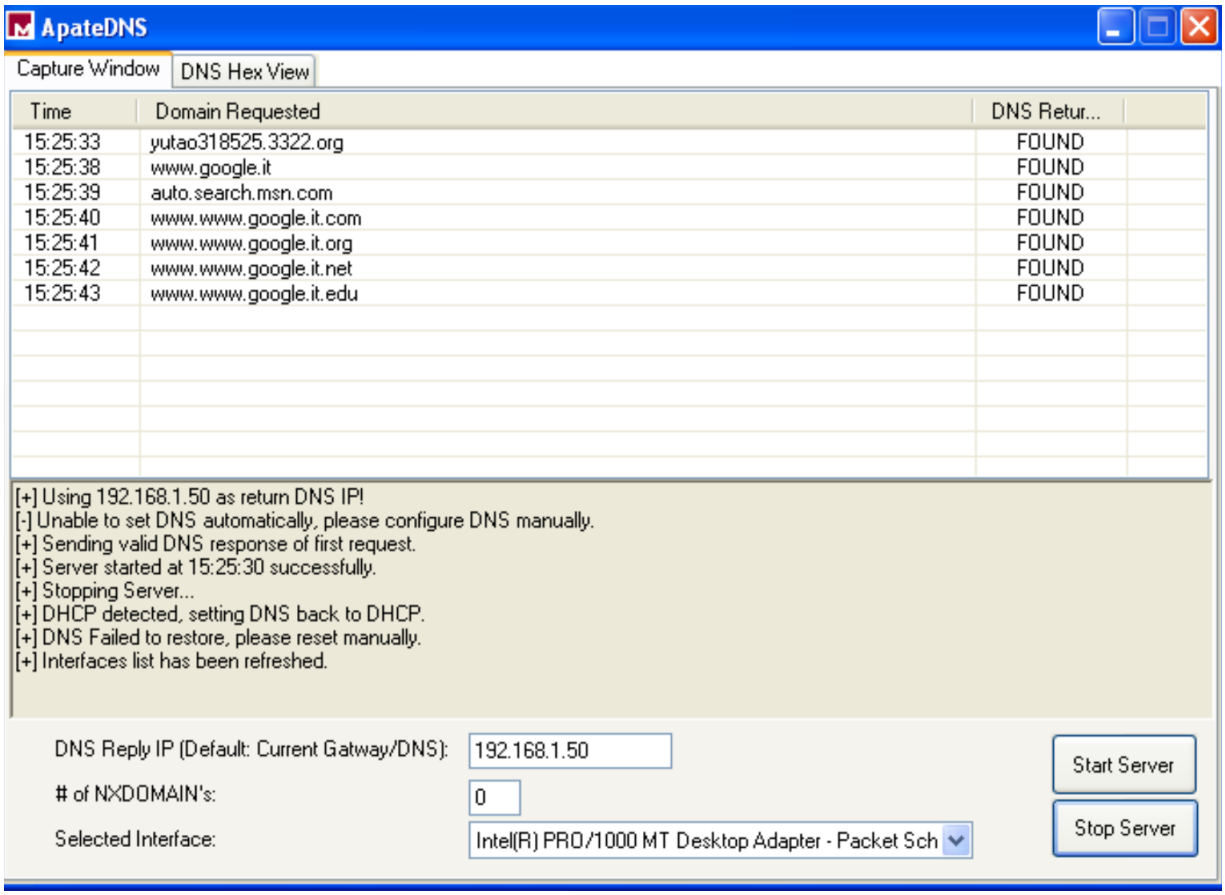
Eseguiamo un'analisi statica basica cercando di ottenere info generali sul file e verificare la sua reputazione sul web partendo ad esempio con l'utilizzo di VirusTotal che consente di verificare la firma del file in questione o l'hash tra quelle riportate nei database di vari antivirus.

Si può calcolare l'hash usando *md5deep* dal prompt dei comandi della macchina o più semplicemente usando CFF Explorer.

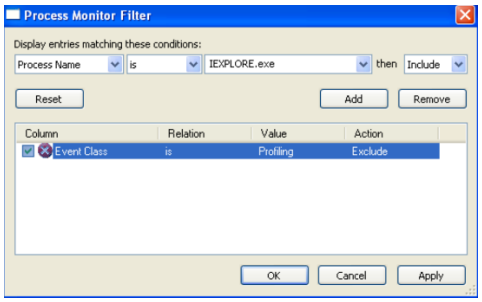


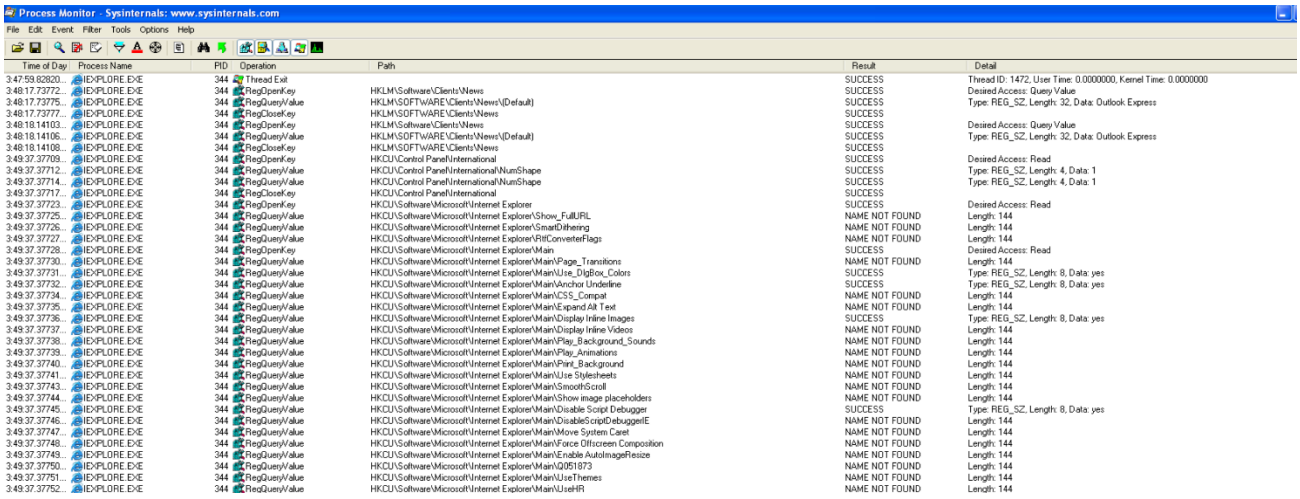


Si evince che il file non è malevole. Proseguiamo però con ulteriori verifiche passando ad un'analisi dinamica basica in cui verrà eseguito il codice del programma:



Con il tool ApateDNS si simula un Server DNS per verificare se il file eseguibile possa mandare richieste verso domini anomali o infetti. Dopo aver inserito in questo caso l'IP di XP, ovvero la macchina su cui si trova l'eseguibile sospetto, si avvia l'analisi; scrivendo www.google.it vediamo che cattura il traffico su di esso ma alla fine non ci sono altre richieste o eventuali anomali.





Utilizzando successivamente Procmon applicando il filtro per nome del file per monitorare i processi, chiavi di registro, file system e attività di rete non si trovano anomalie e modifiche particolari, ne segue che il file risulta essere legittimo e non dannoso.