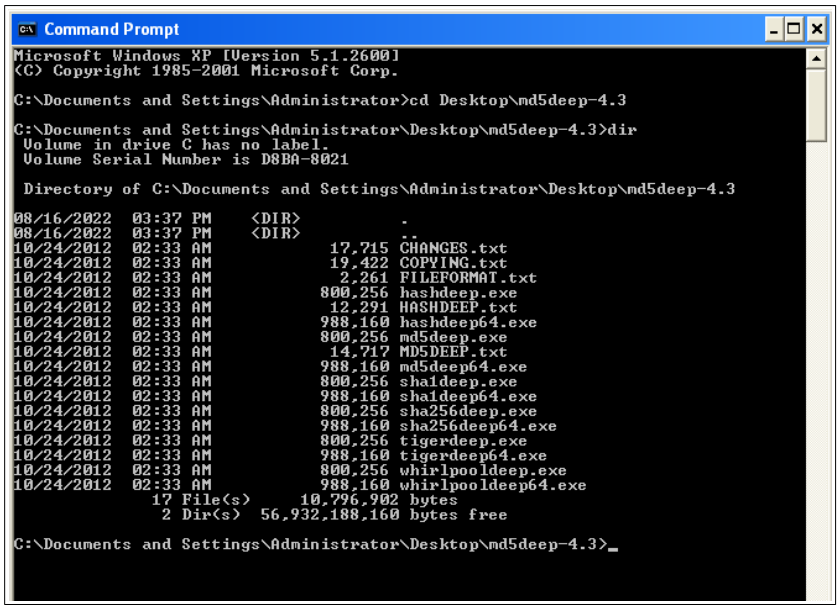
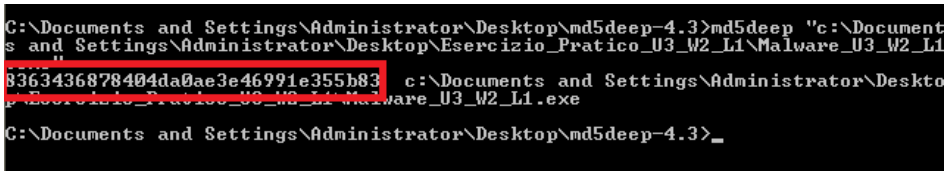
Analisi Statica Basica

Obiettivo: Eseguire un'analisi statica basica di un Malware su macchina Windows XP 32 bit.

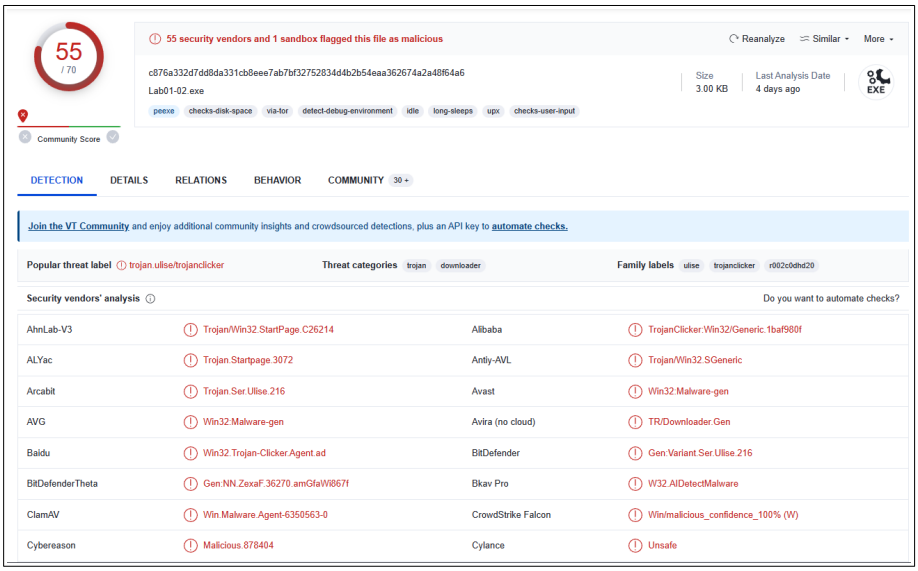
1) Avvio CMD e Analisi Superficiale Virus



Per cominciare ho avviato il **Command Prompt** su **Windows 7** per iniziare ad analizzare il **Malware** richiesto. Ho analizzato per prima cosa l'hash con ***md5deep***.



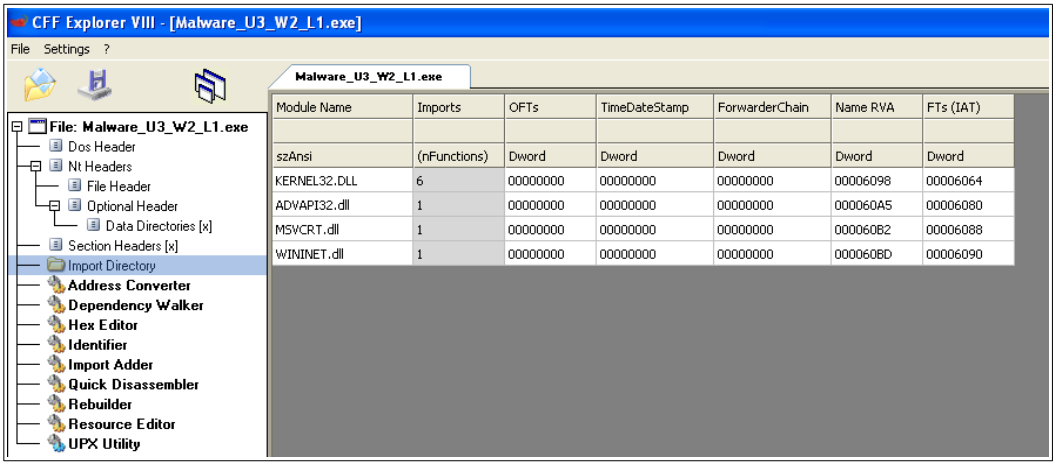
Per ottenere ulteriori informazioni ho caricato l'hash sul sito **VirusTotal**



dove 55/70 Antivirus lo segnalano come Malevolo, principalmente come ***trojan***.

2) CFF Explore

Utilizzando questo tool possiamo controllare le funzioni esportate ed importate dal Malware. Aprendo semplicemente il file e andando nella sezione *Import Directory* vengono mostrate le *Funzioni Importate*.



Module Name	Imports	OFTs	TimeDateStamp	ForwarderChain	Name RVA	FTs (IAT)
szAnsi	(nFunctions)	Dword	Dword	Dword	Dword	Dword
KERNEL32.DLL	6	00000000	00000000	00000000	00006098	00006064
ADVAPI32.dll	1	00000000	00000000	00000000	000060A5	00006080
MSVCRT.dll	1	00000000	00000000	00000000	000060B2	00006088
WININET.dll	1	00000000	00000000	00000000	000060BD	00006090

Le **Librerie** trovate con rispettive funzioni richieste sono:

- **KERNEL32**: una libreria comune che contiene le funzioni principali per interagire con il sistema operativo.

OFTs	FTs (IAT)	Hint	Name
Dword	Dword	Word	szAnsi
N/A	0000219E	0000	SystemTimeToFileTime
N/A	000021B4	0000	GetModuleFileNameA
N/A	000021C8	0000	CreateWaitableTimerA
N/A	000021DE	0000	ExitProcess
N/A	000021EC	0000	OpenMutexA
N/A	000021F8	0000	SetWaitableTimer
N/A	0000220A	0000	WaitForSingleObject
N/A	00002220	0000	CreateMutexA
N/A	0000222E	0000	CreateThread

- **ADVAPI32**: libreria per interagire con i servizi ed i registri del sistema operativo Microsoft.

OFTs	FTs (IAT)	Hint	Name
Dword	Dword	Word	szAnsi
N/A	0000223C	0000	CreateServiceA
N/A	0000224C	0000	StartServiceCtrlDispatcherA
N/A	0000226A	0000	OpenSCManagerA

- **MSVCRT**: libreria che contiene le funzioni per la manipolazione di stringhe, l'allocazione di memoria e chiamateper input/output.

OFTs	FTs (IAT)	Hint	Name
Dword	Dword	Word	szAnsi
N/A	0000227A	0000	_exit
N/A	00002282	0000	_XcptFilter
N/A	00002290	0000	exit
N/A	00002296	0000	__p___initenv
N/A	000022A6	0000	__getmainargs
N/A	000022B6	0000	_initterm
N/A	000022C2	0000	__setusermatherr
N/A	000022D4	0000	_adjust_fdiv
N/A	000022E2	0000	__p__commode
N/A	000022F0	0000	__p__fmode
N/A	000022FC	0000	__set_app_type
N/A	0000230C	0000	_except_handler3
N/A	0000231E	0000	_controlfp

- **WININET**: questa libreria contiene le funzioni per l'implementazione di alcuni protocolli di rete come HTTP, FTP ed NTP.

OFTs	FTs (IAT)	Hint	Name
Dword	Dword	Word	szAnsi
N/A	0000232A	0000	InternetOpenUrlA
N/A	0000233C	0000	InternetOpenA

Alcune funzioni sono risultate visibili solo dopo aver decompresso con ***UPX Utility*** il codice del **Malware**. Questa procedura di crittografazione è utilizzata dai creatori di Malware per renderli meno individuabili agli occhi degli antivirus.

3) Sezioni del Malware

Con **CFF Explore** è possibile analizzare anche le *Sezioni* che compongono il Malware analizzato, più precisamente in **Section Headers**. Se controlliamo questa parte prima di decomprimere con ***UPX Utility*** non verranno mostrati i nomi delle sezioni ma la dicitura ***UPX***.

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
UPX0	00004000	00001000	00000000	00000400	00000000	00000000	0000	0000	E0000080
UPX1	00001000	00005000	00000600	00000400	00000000	00000000	0000	0000	E0000040
UPX2	00001000	00006000	00000200	00000A00	00000000	00000000	0000	0000	C0000040

Perciò utilizziamo l'Utility del nostro programma

Name	Virtual Size	Virtual Address	Raw Size	Raw Address	Reloc Address	Linenumbers	Relocations ...	Linenumber...	Characteristics
00000228	00000230	00000234	00000238	0000023C	00000240	00000244	00000248	0000024A	0000024C
Byte[8]	Dword	Dword	Dword	Dword	Dword	Dword	Word	Word	Dword
.text	000002DC	00001000	00001000	00001000	00000000	00000000	0000	0000	60000020
.rdata	00000372	00002000	00001000	00002000	00000000	00000000	0000	0000	40000040
.data	0000008C	00003000	00001000	00003000	00000000	00000000	0000	0000	C0000040

Troviamo le sezioni:

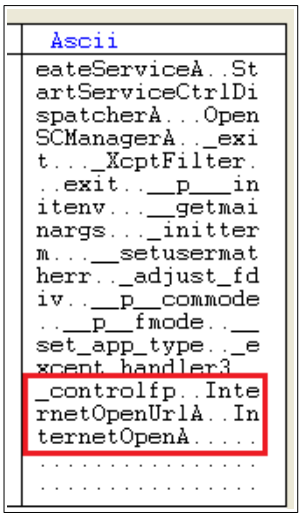
- **text**: contiene istruzioni che la CPU eseguirà una volta che il software sarà avviato.

- **rdata**: include informazioni circa le librerie e le funzioni importate ed esportate dall'eseguibile.

- **data**: questa sezione contiene i dati e variabili globali del programma eseguibile, che devono essere disponibili da qualsiasi parte del programma.

4) Conclusioni Finali

La traccia dell'esercizio chiede di ipotizzare cosa questo **Malware** faccia. Analizzando le sezioni **rdata** e **data** ho notato che nella prima alcune stringhe dove accenna ad un ***mutex***, un processo di sincronizzazione che impedisce che task paralleli accedano contemporaneamente ai dati di memoria.



In caso di **Malware** un ***mutex*** è utilizzato ad esempio per far eseguire task in maniera che non vadano in conflitto tra loro (ad esempio se si parla di un Malware complesso e deve eseguire più processi) o mascherare la propria presenza utilizzando nomi di applicazioni e servizi di sistema.

In secondo luogo ho anche notato la presenza di un ***Internet Open URL*** che collegato con il link trovato nella sezione data (figura in basso) mi fa pensare che probabilmente questo Malware porti ad creare una connessione verso la pagina e a mantenere attiva la sessione.