

Report UNIT 3 WEEK 10.1

ANALISI FILE

Analizzo il file con **Hybrid Analysis**, viene segnalato come malevolo su più versioni di Windows. È conosciuto con molte varianti di nomi.

Submission name: Lab01-02.exe ⓘ

Size: 3KiB

Type: peexe executable ⓘ

Mime: application/x-dosexec

SHA256: c876a332d7dd8da331cb8eee7ab7bf32752834d4b2b54eaa362674a2a48f64a6 ⓘ

Operating System: Windows

Last Anti-Virus Scan: 04/05/2023 17:14:58 (UTC)

Last Sandbox Report: 11/05/2022 20:58:26 (UTC)

malicious

Threat Score: 100/100

AV Detection: 81%

Labeled as: Ser.Ulise.Generic

#tag #adware #autorun #backdoor

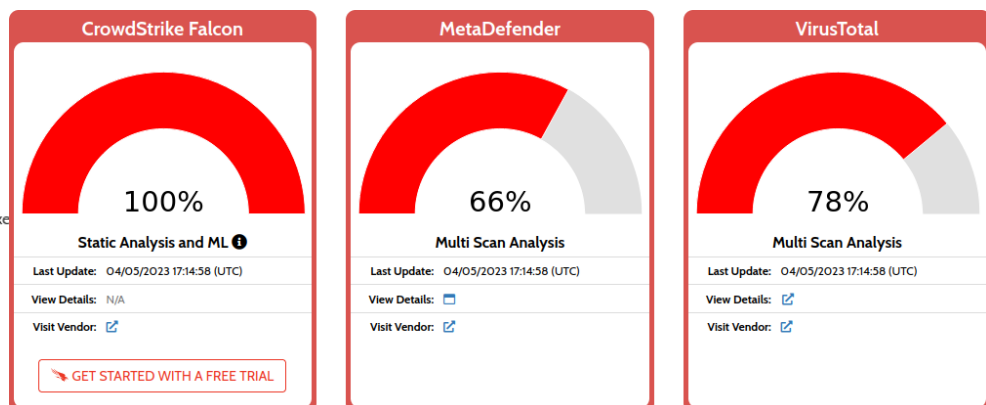
#crypt #downloader #exploit

#injector #keylogger #ransomware

#riskware #rootkit #toolbar

#worm


- 01-02.exe
- 8363436878404da0ae3e46991e355b83
- Copy of dont_run_me.exe
- Dragon01.exe
- Malware02.exe
- Malware_U3_W2_L1.exe
- Sample-02.exe
- WindowsUpdate.bin
- WindowsUpdate.exe
- bounty-17670003943381324
- bounty-70992943971905906
- bounty-87304316951485388
- dont_run_me.exe
- example.exe
- exercise_2_exe
- file
- file2-2.exe
- lab02-01.exe
- literal_strings.exe
- mal01-02_8363436878404da0ae3e46991e355b83.exe
- malware.exe
- sample01.exe
- sample2.exe
- static_analysis_2.exe
- static_analysis_2.mal
- test01-02.exe
- test1.exe
- virus-removal.exe



Dall'incident response ricaviamo:

- Remote Access** [Reads terminal service related keys \(often RDP related\)](#)
- Spyware** [Hooks API calls](#)
- Persistence** [Installs hooks/patches the running process](#)
- Fingerprint** [Queries kernel debugger information](#)
[Reads the active computer name](#)
[Reads the windows installation language](#)
- Evasive** [PE file has a section name known to be used by a packer/protector](#)

- Legge le chiavi di registro relative al servizio terminale, spesso associate al protocollo RDP (Remote Desktop Protocol).
- Intercetta le chiamate alle API di sistema.
- Installa hook o patch nel processo in esecuzione.
- Interroga le informazioni sul debugger del kernel.
- Legge il nome del computer attivo.
- Legge la lingua di installazione di Windows.
- Il file PE ha una sezione con un nome comunemente usato da un programma di compressione o protezione, spesso associato a malware.

Lab01-02.exe	
Filename	Lab01-02.exe
Size	3KiB (3072 bytes)
Type	peexe executable
Description	PE32 executable (console) Intel 80386, for MS Windows, UPX compressed
Architecture	WINDOWS
SHA256	c876a332d7dd8da331cb8eee7ab7bf32752834d4b2b54eaa362674a2a48f64a6 
Compiler/Packer	UPX v1.25 (Delphi) Stub

UPX è uno strumento utilizzato per comprimere eseguibili, riducendo così le dimensioni dei file e facilitandone la distribuzione. Il Delphi stub è la porzione di codice aggiunta all'inizio dell'eseguibile compresso con UPX per consentirne l'estrazione e l'esecuzione corretta.

File Metadata

File Compositions	Imported Objects	File Analysis
-------------------	------------------	---------------

- 2 .CPP Files compiled with CL.EXE 12.00 (Visual Studio 6) (build: 8168)
- 5 .OBJ Files (COFF) linked with LINK.EXE 5.12 (Visual Studio 5 SP2) (build: 8034)

Nella composizione dei file viene indicato che il codice sorgente (.cpp) è stato compilato utilizzando Visual Studio 6 con la versione 12.00 del compilatore CL.EXE, mentre i file oggetto (.obj) sono stati collegati utilizzando Visual Studio 5 SP2 con la versione 5.12 del linker LINK.EXE.

File Metadata

File Compositions	Imported Objects	File Analysis
-------------------	------------------	---------------

- 2 .OBJ Files (COFF) linked with LINK.EXE 6.00 (Visual Studio 6) (build: 8168)
- 1 .ASM Files assembled with MASM 6.13 (Visual Studio 6 SP1) (build: 7299)
- 11 .C Files compiled with CL.EXE 12.00 (Visual Studio 6) (build: 8168)

Negli oggetti importati viene indicato che il file importa 2 file oggetto, 1 file assembly e 11 file di codice sorgente che sono stati collegati o compilati utilizzando gli strumenti di Visual Studio 6, inclusi il linker LINK.EXE, il MASM e il compilatore CL.EXE.

File Metadata

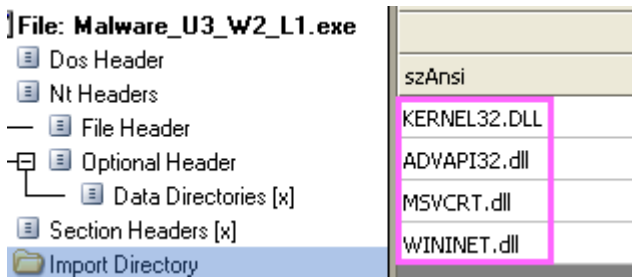
File Compositions	Imported Objects	File Analysis
-------------------	------------------	---------------

- File contains C++ code
- File appears to contain raw COFF/OMF content
- File is the product of a small codebase (2 files)

L'analisi del file suggerisce che il file sia scritto utilizzando il linguaggio di programmazione C++ e fa parte di un progetto più ampio composto da soli due file. Inoltre, suggerisce che il formato del file sia COFF/OMF.

ANALISI LIBRERIE

Aprendo CFF Explorer ed importando il malware per l'analisi possiamo visualizzare le librerie importate.



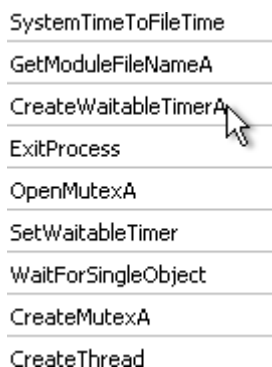
KERNEL32.DLL è una libreria di sistema di Windows che fornisce funzioni di basso livello per interagire con il sistema operativo. Essa gestisce processi, thread, memoria, file, tempo e risorse. La libreria è essenziale per molti programmi Windows, consentendo loro di eseguire operazioni fondamentali come creare processi, leggere/scrivere file, allocare memoria e gestire il tempo di sistema. KERNEL32.DLL svolge un ruolo cruciale nella comunicazione tra un'applicazione e il sistema operativo, fornendo le funzionalità necessarie per l'esecuzione corretta dei programmi su Windows.

ADVAPI32.DLL è una libreria di sistema di Windows che fornisce funzionalità di crittografia, gestione dei certificati, autenticazione, controllo degli accessi e altre operazioni di sicurezza. Contiene una serie di funzioni che consentono di generare chiavi crittografiche, crittografare e decrittografare dati, gestire certificati digitali e molto altro. È ampiamente utilizzata dalle applicazioni Windows per implementare meccanismi di sicurezza avanzati.

MSVCRT.DLL è una libreria di runtime di Microsoft Visual C++ che contiene le funzioni di runtime standard utilizzate dai programmi C e C++ compilati con il compilatore Microsoft Visual C++. Fornisce funzionalità come l'allocazione della memoria, la gestione delle stringhe, l'input/output dei file, le operazioni matematiche e altre funzioni comuni utilizzate nelle applicazioni C e C++. È essenziale per il corretto funzionamento dei programmi compilati con il compilatore Microsoft Visual C++.

WININET.DLL è una libreria di sistema di Windows che fornisce funzionalità per la comunicazione via Internet. È utilizzata dalle applicazioni Windows per effettuare richieste HTTP, inviare e ricevere dati tramite protocolli di rete come HTTP, FTP e Gopher, gestire cookie, cache, proxy e altre operazioni di rete. WININET.DLL permette alle applicazioni di connettersi a server web, scaricare file, effettuare richieste di ricerca e altre attività legate alla comunicazione via Internet. È ampiamente utilizzata dai browser web e da altre applicazioni che richiedono accesso a risorse Internet.

Nella libreria **KERNEL32** ritroviamo:



Queste funzioni forniscono meccanismi di sincronizzazione, gestione del tempo e controllo dei thread che sono essenziali per lo sviluppo di applicazioni su piattaforma Windows.

Nella libreria **ADVAPI32** ritroviamo:

```
CreateServiceA
StartServiceCtrlDispatcherA
OpenSCManagerA
```

Questa libreria offre le capacità necessarie per interagire con i servizi di Windows e gestirli in modo programmato fornendo le funzioni per la creazione, gestione e controllo dei servizi.

Nella libreria **MSVCRT** ritroviamo:

```
_exit
_XcptFilter
exit
__p__initenv
__getmainargs
__initterm
__setusermatherr
__adjust_fdiv
__p__commode
__p__fmode
__set_app_type
_except_handler3
_controlfp
```

Queste funzioni consentono di gestire aspetti fondamentali dell'esecuzione del programma, come l'uscita, la gestione delle eccezioni, la gestione degli argomenti della riga di comando e altre operazioni comuni nella programmazione in C++.

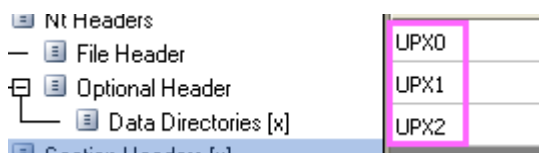
Nella libreria **WININET** ritroviamo:

```
InternetOpenUrlA
InternetOpenA
```

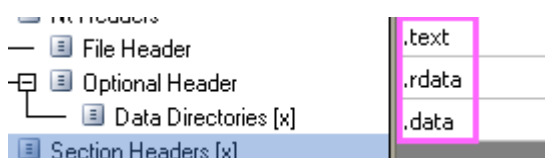
Entrambe queste funzioni fanno parte dell'API di WinINet, che fornisce una serie di funzioni per la comunicazione con i server Internet e l'accesso alle risorse web. Queste funzioni sono spesso utilizzate per implementare funzionalità di navigazione web, download di file o comunicazione con server remoti all'interno delle applicazioni Windows.

ANALISI SEZIONI

Le sezioni sono nascoste da UPX. Rimuovendo o nascondendo le sezioni, UPX può ridurre la dimensione del file.



Tramite **UPX utility** faccio l'unpacking, cioè il ripristino delle sezioni originali.



Salvo il file e lo apro su IDA.

Strings window			
Address	Length	T...	String
.rdata:0...	0000000D	C	KERNEL32.DLL
.rdata:0...	0000000D	C	ADVAPI32.dll
.rdata:0...	0000000B	C	MSVCRT.dll
.rdata:0...	0000000C	C	WININET.dll
.rdata:0...	00000015	C	SystemTimeToFileTime
.rdata:0...	00000013	C	GetModuleFileNameA
.rdata:0...	00000015	C	CreateWaitableTimerA
.rdata:0...	0000000C	C	ExitProcess
.rdata:0...	0000000B	C	OpenMutexA
.rdata:0...	00000011	C	SetWaitableTimer
.rdata:0...	00000014	C	WaitForSingleObject
.rdata:0...	0000000D	C	CreateMutexA
.rdata:0...	0000000D	C	CreateThread
.rdata:0...	0000000F	C	CreateServiceA
.rdata:0...	0000001C	C	StartServiceCtrlDispatcherA
.rdata:0...	0000000F	C	OpenSCManagerA
.rdata:0...	00000006	C	_exit
.rdata:0...	0000000C	C	_XcptFilter
.rdata:0...	00000005	C	exit
.rdata:0...	0000000E	C	__p__initenv
.rdata:0...	0000000E	C	__getmainargs
.rdata:0...	0000000A	C	_initterm
.rdata:0...	00000011	C	__setusermatherr
.rdata:0...	0000000D	C	_adjust_fdiv
.rdata:0...	0000000D	C	__p__commode
.rdata:0...	0000000B	C	__p__fmode
.rdata:0...	0000000F	C	__set_app_type
.rdata:0...	00000011	C	_except_handler3
.rdata:0...	0000000B	C	_controlfp
.rdata:0...	00000011	C	InternetOpenUrlA
.rdata:0...	0000000E	C	InternetOpenA
.data:00...	0000000B	C	MalService
.data:00...	0000000B	C	Malservice
.data:00...	00000007	C	HGL345
.data:00...	00000023	C	http://www.malwareanalysisbook.com
.data:00...	00000016	C	Internet Explorer 8.0

Dalle stringhe posso fare delle considerazioni.

Oltre le librerie mostrate precedentemente e, conseguentemente, i loro utilizzi richiamati dalle varie stringhe, possiamo notare come interagiscano in particolar alcune di esse.

CreateMutexA crea un oggetto mutex per evitare l'esecuzione di più malware contemporaneamente e con **OpenMutexA** controlla se il malware sia già in esecuzione.

I malware spesso utilizzano i mutex per prevenire l'esecuzione multipla, cioè per evitare che più istanze del malware vengano avviate contemporaneamente. Creando un mutex con un nome specifico, il malware può verificare se esiste già un'istanza in esecuzione controllando la disponibilità del mutex. In questo modo, il malware può limitare la propria esecuzione a una sola istanza per evitare rilevamenti o conflitti interni.

CreateServiceA crea oggetto per garantire la persistenza del malware.

L'obiettivo della persistenza del malware è quello di garantire che il malware sopravviva il più a lungo possibile sul sistema infetto, consentendo al suo autore di mantenere il controllo o di continuare le operazioni malevole, come il furto di informazioni, il controllo remoto del sistema o l'installazione di altri componenti dannosi.

StartServiceCtrlDispatcherA quando viene avviato il servizio attende che il processo chiami la funzione dopo l'avvio.

OpenSCManagerA stabilisce la connessione con il service control manager per consentire al malware di interagire con il servizio di gestione dei servizi di Windows.

InternetOpenA fa partire l'uso delle funzioni WinINet nell'applicazione, questo viene usato per avviare la connessione.

InternetOpenUrlA apre un URL FTP o HTTP, probabilmente per scaricare o caricare file.

Come considerazioni si può supporre che il malware stia cercando di installare un servizio per garantire la persistenza utilizzando il traffico HTTP per ricevere comandi dal server.

```
push offset Name ; "HGL345"
push 0 ; bInitialOwner
push 0 ; lpMutexAttributes
call ds:CreateMutexA
push 3 ; dwDesiredAccess
push 3 ; lpDatabaseName
push 3 ; lpMachineName
call ds:OpenSCManagerA ; Establish a connection to the service
; control manager on the specified computer
; and opens the specified database

mov esi, eax
lea eax, [esp+404h+BinaryPathName]
push 3E8h ; nSize
push eax ; lpFilename
push 0 ; hModule
call ds:GetModuleFileNameA
push 0 ; lpPassword
push 0 ; lpServiceStartName
push 0 ; lpDependencies
push 0 ; lpdwTagId
lea ecx, [esp+414h+BinaryPathName]
push 0 ; lpLoadOrderGroup
push ecx ; lpBinaryPathName
push 0 ; dwErrorControl
push 2 ; dwStartType
push 10h ; dwServiceType
push 2 ; dwDesiredAccess
push offset DisplayName ; "Malservice"
push offset DisplayName ; "Malservice"
push esi ; hSCManager
call ds:CreateServiceA
xor edx, edx

push esi
push edi
push 0 ; dwFlags
push 0 ; lpzProxyBypass
push 0 ; lpzProxy
push 1 ; dwAccessType
push offset szAgent ; "Internet Explorer 8.0"
call ds:InternetOpenA
mov edi, ds:InternetOpenUrlA
mov esi, eax

push 0 ; CODE XREF: StartAddress+30↓j
push 0 ; dwContext
push 80000000h ; dwFlags
push 0 ; dwHeadersLength
push 0 ; lpzHeaders
push offset szUrl ; "http://www.malwareanalysisbook.com"
push esi ; hInternet
call edi ; InternetOpenUrlA
jmp short loc_40116D
endb
```