

# TRACCIA FINALE W2404

## MALWARE ANALYSIS

### TRACCIA:

IL MALWARE DA ANALIZZARE È NELLA CARTELLA BUILD\_WE-EK\_UNIT\_3 PRESENTE SUL DESKTOP DELLA MACCHINA VIRTUALE DEDICATA.

### ANALISI STATICA

CON RIFERIMENTO AL FILE ESEGUIBILE MALWARE\_BUILD\_WE-EK\_U3, RISPONDERE AI SEGUENTI QUESITI UTILIZZANDO I TOOL E LE TECNICHE APPRESE NELLE LEZIONI TEORICHE:

- QUANTI PARAMETRI SONO PASSATI ALLA FUNZIONE MAIN()?
- QUANTE VARIABILI SONO DICHIARATE ALL'INTERNO DELLA FUNZIONE MAIN()?
- QUALI SEZIONI SONO PRESENTI ALL'INTERNO DEL FILE ESEGUIBILE? DESCRIVETE BREVEMENTE ALMENO 2 DI QUELLE IDENTIFICATE
- QUALI LIBRERIE IMPORTA IL MALWARE? PER OGNUNA DELLE LIBRERIE IMPORTATE, FATE DELLE IPOTESI SULLA BASE DELLA SOLA ANALISI STATICA DELLE FUNZIONALITÀ CHE IL MALWARE POTREBBE IMPLEMENTARE. UTILIZZATE LE FUNZIONI CHE SONO RICHIAMATE ALL'INTERNO DELLE LIBRERIE PER SUPPORTARE LE VOSTRE IPOTESI.

L'ESECUZIONE DELL'ESERCIZIO RICHIEDE LA COMBINAZIONE DI VARIE ANALISI VISTE DURANTE IL MODULO 6 DEL CORSO. PER QUESTA PRIMA PARTE SERVIRÀ L'APPLICAZIONE DELL'ANALISI STATICA BASICA ED AVANZATA.

L'ANALISI STATICA SI RIFERISCE ALL'ISPEZIONE DEL CODICE SORGENTE O DEL CODICE BINARIO DI UN PROGRAMMA (IN QUESTO CASO, UN MALWARE) PER IDENTIFICARNE LA FUNZIONALITÀ, LE CARATTERISTICHE E LE POTENZIALI MINACCE SENZA ESEGUIRLO. QUESTO APPROCCIO SI CONTRAPPONE ALL'ANALISI DINAMICA, DOVE IL CODICE VIENE ESEGUITO IN UN AMBIENTE CONTROLLATO (SANDBOX) PER OSSERVARE IL SUO COMPORTAMENTO.

L'ANALISI STATICA BASICA CONSISTE NELL'ESAMINARE UN ESEGUIBILE SENZA VEDERE LE ISTRUZIONI CHE LO COMPONGONO E LA SUA FUNZIONE È CONFERMARE SE UN DATO FILE È MALEVOLO E FORNIRE INFORMAZIONI GENERICHE CIRCA LE SUE FUNZIONALITÀ. L'ANALISI STATICA AVANZATA PRESUPPONE LA CONOSCENZA DEI FONDAMENTI DI «REVERSE-ENGINEERING» AL FINE DI IDENTIFICARE IL COMPORTAMENTO DI UN MALWARE A PARTIRE DALL'ANALISI DELLE ISTRUZIONI CHE LO COMPONGONO. QUESTO PASSAGGIO È ESSENZIALE PER CAPIRE ESATTAMENTE COSA FA IL MALWARE A LIVELLO DI ISTRUZIONI DELLA CPU. SI POSSONO INOLTRE ESTRARRE STRINGHE DI TESTO, URL, CHIAVI DI CIFRATURA, E ALTRE RISORSE DAL CODICE DEL MALWARE, CHE POSSONO INDICARE IL SUO COMPORTAMENTO O INTENTO E SE NE PUÒ ESAMINARE IL CODICE RELATIVO ALLA RETE PER COMPRENDERE COME IL MALWARE COMUNICA.

PRIMA DI PROCEDERE ALL'ANALISI MI ASSICURO CHE SIA DI FATTO UN MALWARE, ESTRAENDONE L'HASH CON MD5DEEP E CONTROLLANDO SU VIRUSTOTAL LA SUA REPUTAZIONE CHE SI BASA SU VARI RISCONTRI DI SOFTWARE ANTIVIRUS.

IN QUESTA PRIMA ANALISI POSSO VEDERE CHE IL VIRUS È NOTO, SI TRATTA DI UN MALWARE DI TIPO TROJAN COMPILATO IN DATA 11-06-2011 IN C++, ANALIZZATO L'ULTIMA VOLTA IN DATA 17 APRILE 2024. È UN MALWARE PROGETTATO PER COLPIRE LA MACCHINA INTEL 386 E PROCESSORI SUCESSIVI/COMPATIBILI.

HA 5255 ENTRY POINTS, 4 SEZIONI ED IMPORTA 2 LIBRERIE: KERNEL32.DLL - UNA DELLE LIBRERIE FONDAMENTALI DI WINDOWS, CONTIENE NUMEROSE FUNZIONI CHE GESTISCONO LA MEMORIA, I PROCESSI E I THREAD. I MALWARE LA UTILIZZANO PER MANIPOLARE I PROCESSI E PER ACCEDERE A DIVERSE API DI SISTEMA ED OTTENERE PERSISTENZA.

ADVAPI32.DLL - FORNISCE FUNZIONI RELATIVE ALLA SICUREZZA E ALLA GESTIONE DI ACCOUNT, CHE I MALWARE POSSONO SFRUTTARE PER MODIFICARE PERMESSI, ACCEDERE A TOKEN DI SICUREZZA E ALTERARE IL REGISTRO DI SISTEMA, AD ESEMPIO PER ESSERE AVVIATI ALL'AVVIO DEL SISTEMA OPERATIVO.

ANDRÒ A CONFERMARE TUTTI QUESTI DATI TRAMITE TOOL DEDICATI COME CFFEXPLORER/EXEINFOPE E IDA PRO.

### #HASH CON MD5DEEP64

```
C:\Users\user\Desktop\Software Malware analysis\md5deep-4.3>cd md5deep-4.3
C:\Users\user\Desktop\Software Malware analysis\md5deep-4.3\md5deep-4.3>md5deep64
a9c55bb87a7c5c3c923c4fa12940e719 C:\Users\user\Desktop\Software Malware analysis\md5deep-4.3\md5deep-4.3\malware_Build_Week_U3.exe
```

### #VIRUSTOTAL

52 / 71

Community Score

52/71 security vendors and no sandboxes flagged this file as malicious

Reanalyze Similar More

57d8d248a8741176348b5d12dcf29f34c8f48ede0ca13c30d12e5ba0384056d7

Size 52.00 KB

Last Modification Date 1 hour ago

Lab3.exe

peexe spreader armadillo checks-user-input

DETECTION DETAILS RELATIONS BEHAVIOR TELEMETRY COMMUNITY 10

Join the VT Community and enjoy additional community insights and crowdsourced detections, plus an API key to automate checks.

Popular threat label trojan.doina/totbrick Threat categories trojan Family labels doina totbrick genericrcqc

Security vendors' analysis Do you want to automate checks?

AhnLab-V3	Trojan/Win32.Agent.C39204	Alibaba	Trojan:Win32/Totbrick.db39e83f
AliCloud	Backdoor	ALYac	Gen:Variant.Doina.65814
Antiy-AVL	Trojan/Win32.Agent	Arcabit	Trojan.Doina.D10116
Avast	Win32:Trojan-gen	AVG	Win32:Trojan-gen
Avira (no cloud)	TR/Agent.53248.465	BitDefender	Gen:Variant.Doina.65814
BitDefenderTheta	Gen:NN.ZedlaF.36802.aq4@a0c1rOb	Bkav Pro	W32.AIDetectMalware
ClamAV	Win.Trojan.Agent-595082	CrowdStrike Falcon	Win/malicious_confidence_100% (W)
Cylance	Unsafe	Cynet	Malicious (score: 99)

DeepInStinct	MALICIOUS	DrWeb	BackDoor.Siggen2.1689
Elastic	Malicious (moderate Confidence)	Emsisoft	Gen:Variant.Doina.65814 (B)
eScan	Gen:Variant.Doina.65814	ESET-NOD32	Win32/Agent.WOJ
Fortinet	W32/Agent.WOJ!tr	GData	Gen:Variant.Doina.65814
Google	Detected	Ikarus	Trojan-Dropper.Agent
Kingsoft	Malware.kb.a.990	Malwarebytes	Backdoor.Agent.MSGN
MAX	Malware (ai Score=94)	MaxSecure	Trojan.Malware.4145763.susgen
McAfee	GenericRXCQ-GTIA9C55BB87A7C	Microsoft	Trojan:Win32/Totbrick!MTB
NANO-Antivirus	Trojan.Win32.Agent.douhj	Rising	Trojan.Agent!B.B1E (TFE:S:kbaGuqs4fVE)
Sangfor Engine Zero	Trojan.Win32.Totbrick.V0yt	Skyhigh (SWG)	GenericRXCQ-GTIA9C55BB87A7C
Sophos	Mal/Generic-S	Symantec	Trojan.Gen.2
TACHYON	Trojan/W32.Agent.53248.CTF	Tencent	Malware.Win32.Gencirc.10b6ad10
Trellix (FireEye)	Generic.mg.a9c55bb87a7c5c3c	TrendMicro	TROJ_GEN.R014C0DCI21
TrendMicro-HouseCall	TROJ_FR5.0NA103.JQ23	VBA32	Trojan.Occamy
VIPRE	Gen:Variant.Doina.65814	ViriT	Backdoor.Win32.Agent.APVW
ViRobot	Backdoor.Win32.A.Agent.53248.BB	Webroot	W32.Trojan.Gen
WithSecure	Trojan.TR/Agent.53248.465	Xcitiium	Malware@#Ireyd8ywbfp2v
Yandex	Backdoor.Agent!VqksATK3A60	Zillya	Backdoor.Agent.Win32.38748

Portable Executable Info

Compiler Products

[C++] VS98 (6.0) SP6 build 8804 count=1

[C] VS98 (6.0) SP6 build 8804 count=55

[...] Unmarked objects count=54

[RES] VS98 (6.0) SP6 cvtres build 1736 count=1

id: 0xe, version: 7299 count=15

id: 0x13, version: 8034 count=5

id: 0x15, version: 9782 count=1

Header

Target Machine Intel 386 or later processors and compatible processors

Compilation Timestamp 2011-11-06 18:55:06 UTC

Entry Point 5255

Contained Sections 4

Sections

Name	Virtual Address	Virtual Size	Raw Size	Entropy	MD5	Ch2
.text	4096	22086	24576	6.23	6bb361ab84e6ea32f545b12825db9c07	304301.84
.rdata	28672	2478	4096	3.77	23fde5162e5b17a6e440a468b942c3b8	290048.5
.data	32768	16040	12288	0.6	e433b4c400efc11a593220e77ab72779	2826623.75
.rsrc	49152	6768	8192	4.15	9d561586eeb5ecda6c3214cd6a35d6f3	573983.75

Imports

+ KERNEL32.dll

+ ADVAPI32.dll

Contained Resources By Type

BINARY 1

## # CFF EXPLORER:

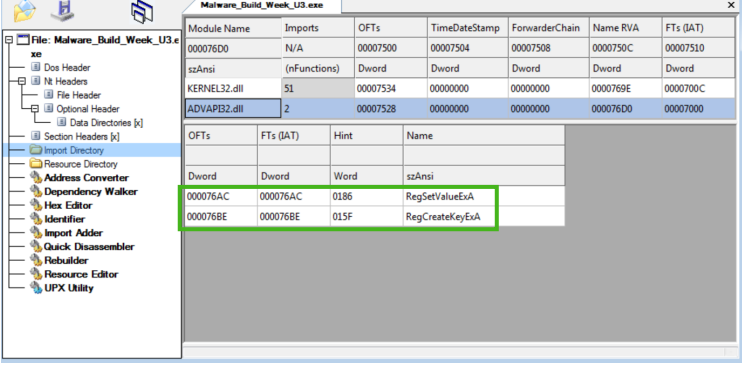
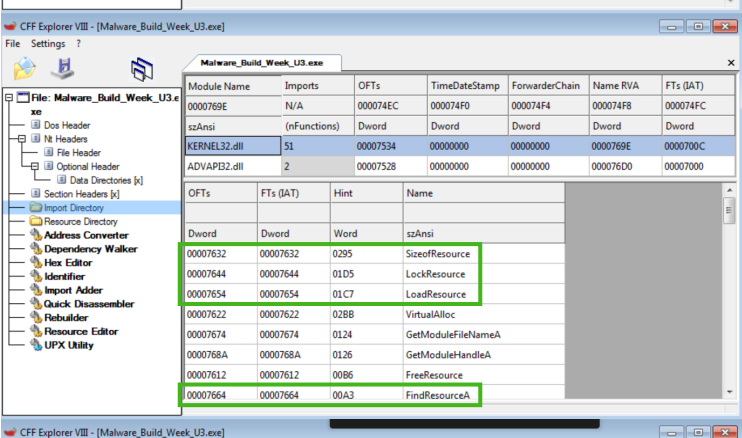
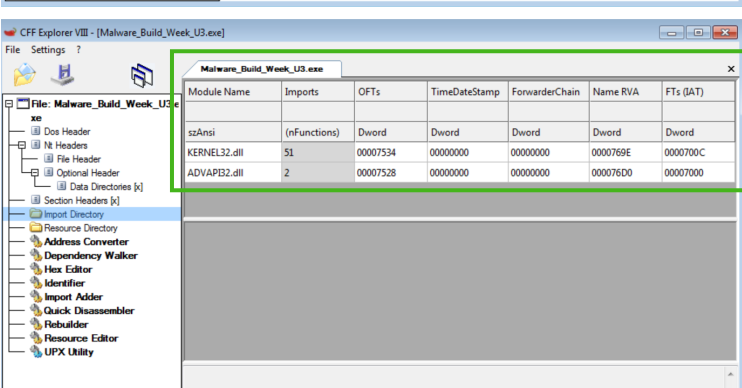
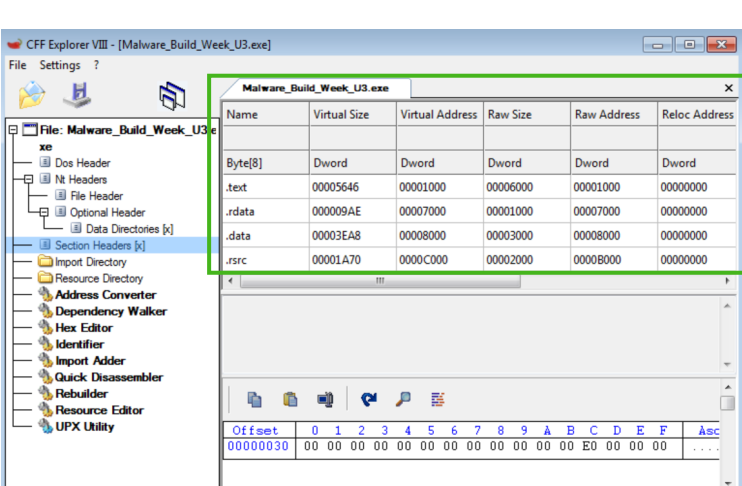
PER CONTROLLARE LE FUNZIONI IMPORTATE ED ESPORTATE DA UN MALWARE, SI PUÒ UTILIZZARE CFF EXPLORER, UN TOOL DA INSTALLARE SULLE MACCHINE VIRTUALI DEDICATE ALL'ANALISI DEI MALWARE. SE MI SPOSTO SU «**IMPORT DIRECTORY**» POSSO CONTROLLARE LE LIBRERIE E LE FUNZIONI IMPORTATE, MENTRE SU «**SECTION HEADERS**» POSSO VEDERE LE SEZIONI PRESENTI ALL'INTERNO DEL FILE ESEGUIBILE:

**.TEXT** - CONTIENE LE RIGHE DI CODICE, ISTRUZIONI, CHE LA CPU ESEGUIRÀ ALL'AVVIO DEL MALWARE.

**.RDATA** - SEZIONE CHE INCLUDE LE INFORMAZIONI CIRCA LE LIBRERIE E LE FUNZIONI IMPORTATE ED ESPORTATE DALL'ESEGUIBILE, DATI CHE IL PROGRAMMA LEGGE MENTRE È IN FUNZIONE.

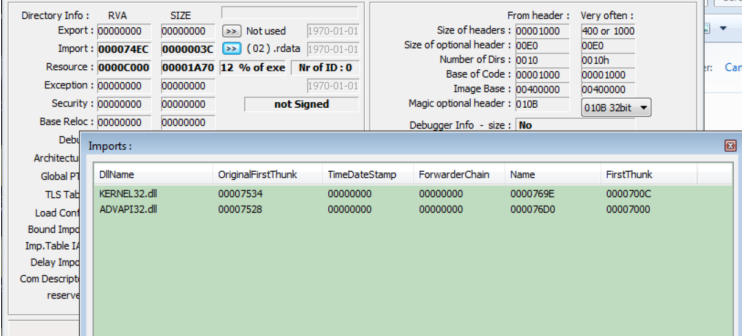
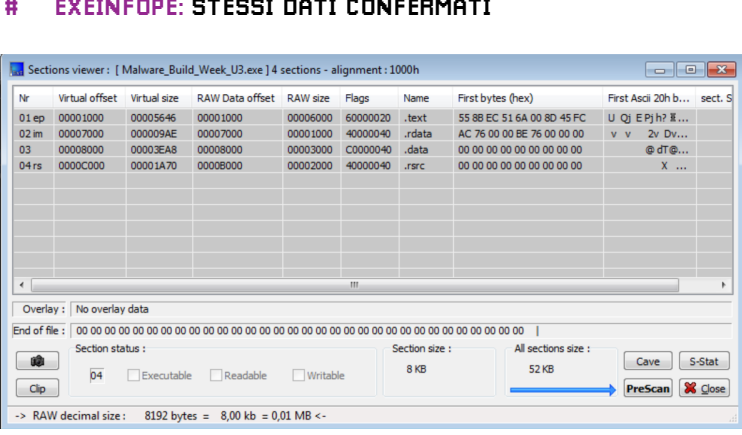
**.DATA** - CONTIENE I DATI / LE VARIABILI GLOBALI DEL PROGRAMMA ESEGUIBILE, CHE DEVONO ESSERE DISPONIBILI DA QUALSIASI PARTE DEL PROGRAMMA E POSSONO ESSERE MODIFICATI.

**.RSRC** - INCLUDE LE RISORSE UTILIZZATE DALL'ESEGUIBILE COME AD ESEMPIO ICONE, IMMAGINI, MENU E STRINGHE CHE NON SONO PARTE DELL'ESEGUIBILE STESSO.



PER QUEL CHE RIGUARDA LE DUE LIBRERIE IMPORTATE, CHE SONO **KERNEL32.DLL** (1) E **ADVAPI32.DLL** (2), POSSO IPOTIZZARE CHE IL MALWARE CERCHI DI OTTENERE PERSISTENZA E MODIFICARE LE CHIAVI DI REGISTRO (**REGSETVALUEEXA/REGCREATEKEYEXA**) PER POTER ESSERE ESEGUITO IN AUTONOMIA. INOLTRE LA PRESENZA DI FUNZIONI COME **SIZEOFRESOURCE/LOCKRESOURCE/LOADRESOURCE/FINDRESOURCEA** FA PRESUPPORRE CHE SIA UN DROPPER, CIOÈ UN MALWARE CHE CONTIENE AL SUO INTERNO UN ALTRO MALWARE, ED UTILIZZA QUESTE APIS CHE PERMETTONO DI LOCALIZZARE ALL'INTERNO DELLA SEZIONE «RISORSE» IL MALWARE DA ESTRARRE, E SUCCESSIVAMENTE DA CARICARE IN MEMORIA PER L'ESECUZIONE.

## # EXEINFOPE: STESSI DATI CONFERMATI



## # IDAPRO:

ORA PER INFORMAZIONI SU **VARIABILI LOCALI** E **PARAMETRI** DELLA FUNZIONE MAIN (), USERÒ **IDA PRO** (INTERACTIVE DISASSEMBLER PROFESSIONAL) CHE È UNO STRUMENTO AVANZATO DI REVERSE ENGINEERING SOFTWARE CHE OFFRE CAPACITÀ DI DISASSEMBLAGGIO, DEBUGGING E ANALISI STATICA.

**VARIABILI LOCALI:** LE VARIABILI LOCALI IN UNA FUNZIONE ASSEMBLY SONO TIPICAMENTE ALLOCATE NELLO STACK. QUESTO È SPESSO FATTO ATTRAVERSO ISTRUZIONI DI TIPO PUSH ALL'INIZIO DI UNA FUNZIONE O CON UN'ISTRUZIONE SUB CHE AUMENTA IL PUNTATORE DELLO STACK (SP) PER CREARE SPAZIO.

**PARAMETRI:** SOLITAMENTE I PARAMETRI SONO PASSATI AI REGISTRI O ATTRAVERSO L'USO DELLO STACK PRIMA DELLA CHIAMATA DELLA FUNZIONE. I COMMENTI NEL CODICE POSSONO FORNIRE INDICAZIONI SU DOVE E QUANTI PARAMETRI SONO PASSATI.

NELL'ASSEMBLY, LE ETICHETTE CHE INIZIANO CON VAR\_ TENDONO A INDICARE VARIABILI LOCALI, MENTRE QUELLE CHE INIZIANO CON ARG\_ INDICANO ARGOMENTI O PARAMETRI PASSATI ALLA FUNZIONE.

VALORI OFFSET: GLI OFFSET (COME VAR\_54H) SONO UTILIZZATI PER ACCEDERE A DATI SPECIFICI SULLO STACK. GLI OFFSET NEGATIVI RISPETTO ALL'INDIRIZZO BASE DEL FRAME DELLA FUNZIONE (AD ESEMPIO EBP SU X86) DI SOLITO INDICANO VARIABILI LOCALI, MENTRE GLI OFFSET POSITIVI INDICANO PARAMETRI.

IN QUESTO CASO LE **VARIABILI LOCALI** IN EVIDENZA CHE POSSIAMO RILEVARE AD UN'OCCHIATA SONO 5: **hModule** / **Data** / **var\_117** / **var\_8** / **var\_4**.

MENTRE I **PARAMETRI** SONO 3 - **argc** / **argv** / **envp**.

