

Traccia Giorno 3

Fate riferimento al malware: Malware_U3_W3_L3, presente all'interno della cartella Esercizio_Pratico_U3_W3_L3 sul desktop della macchina virtuale dedicata all'analisi dei malware. Rispondete ai seguenti quesiti utilizzando OllyDBG.

1 All'indirizzo 0040106E il Malware effettua una chiamata di funzione alla funzione «CreateProcess». Qual è il valore del parametro «CommandLine» che viene passato sullo stack? (1)

2 Inserite un breakpoint software all'indirizzo 004015A3. Qual è il valore del registro EDX? (2) Eseguite a questo punto uno «step-into». Indicate qual è ora il valore del registro EDX (3) motivando la risposta (4). Che istruzione è stata eseguita? (5)

3 Inserite un secondo breakpoint all'indirizzo di memoria 004015AF. Qual è il valore del registro ECX? (6) Eseguite un step-into. Qual è ora il valore di ECX? (7) Spiegate quale istruzione è stata eseguita (8).

4 BONUS: spiegare a grandi linee il funzionamento del malware

1

All'indirizzo 0040106E il Malware effettua una chiamata di funzione alla funzione «CreateProcess». Qual è il valore del parametro «CommandLine» che viene passato sullo stack? (1)



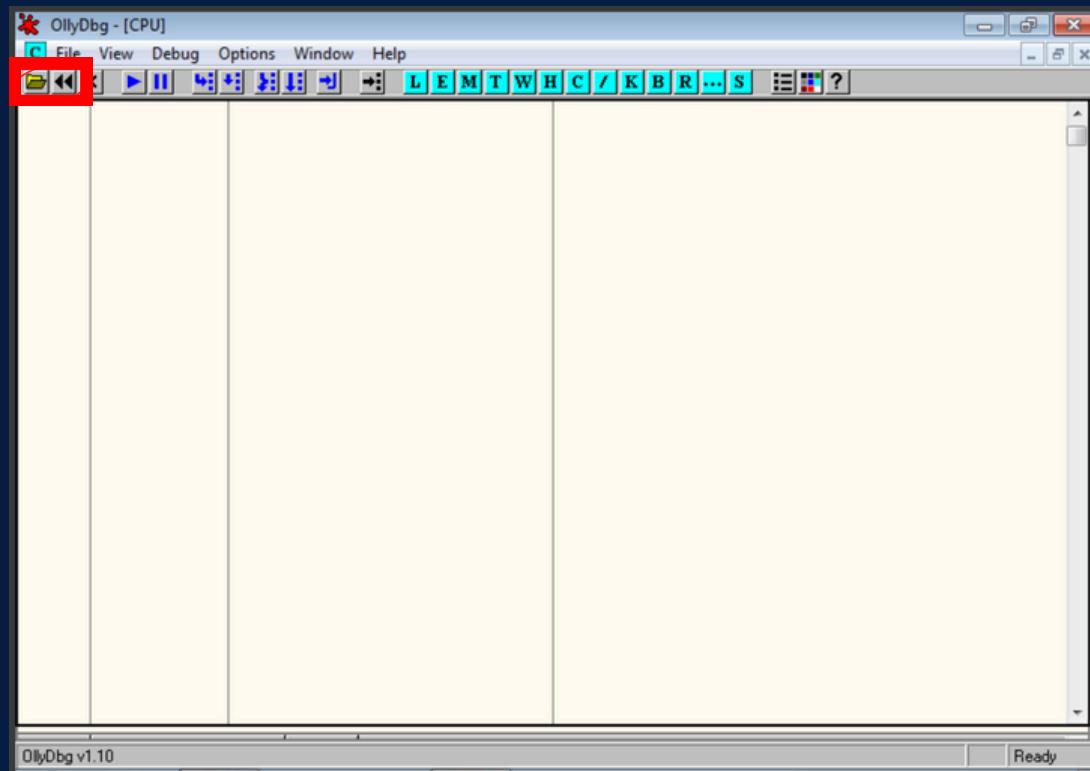
Per effettuare l'analisi dinamica sul Malware ci siamo avvalsi dell'utilizzo del Tool OLLYDBG che troviamo sulla nostra macchina di Windows 7

Scegliamo il Malware da esaminare e quindi selezioniamo il
Malware_U3_W3_L3



1

All'indirizzo 0040106E il Malware effettua una chiamata di funzione alla funzione «CreateProcess». Qual è il valore del parametro «CommandLine» che viene passato sullo stack? (1)



Per caricare il file MALWARE_U3_W3_L3 bisogna fare click su "file">>"nuovo" e selezionare il file, oppure si può aprire il file dalla cartellina in giallo cerchiata nella foto di fianco.

Dopo aver fatto click sul tasto play e quindi aver avviato l'esecuzione del codice, cerchiamo l'indirizzo 0040106E in cui vediamo che viene effettuata la chiamata alla funzione **"CreateProcessA"**, a cui vengono passati diversi parametri inseriti precedentemente nello stack tramite l'istruzione **PUSH**, tra cui "CommandLine" con valore **"cmd"**.

```

00401050  . 894D E4      MOV DWORD PTR SS:[EBP-1C],ECX
00401053  . 8055 F0      LEA EDX,DWORD PTR SS:[EBP-10]
00401056  . 52          PUSH EDX
00401057  . 8D45 A8      LEA EAX,DWORD PTR SS:[EBP-58]
0040105A  . 50          PUSH EAX
0040105B  . 6A 00        PUSH 0
0040105D  . 6A 00        PUSH 0
0040105F  . 6A 00        PUSH 0
00401061  . 6A 01        PUSH 1
00401063  . 6A 00        PUSH 0
00401065  . 6A 00        PUSH 0
00401067  . 68 30504000  PUSH Malware_.00405030
0040106C  . 6A 00        PUSH 0
0040106E  . FF15 04404000 CALL DWORD PTR DS:[&KERNEL32.CreatePro
00401074  . 8945 EC      MOV DWORD PTR SS:[EBP-14],EAX
00401077  . 6A FF        PUSH -1
00401079  . 8B4D F0      MOV ECX,DWORD PTR SS:[EBP-10]
0040107C  . 51          PUSH ECX
0040107D  . FF15 00404000 CALL DWORD PTR DS:[&KERNEL32.WaitForSi
00401083  . 33C0          XOR EAX,EAX
00401085  . 8BE5          MOV ESP,EBP
00401087  . 50          POP EBP

```

pProcessInfo
pStartupInfo
CurrentDir = NULL
pEnvironment = NULL
CreationFlags = 0
InheritHandles = TRUE
pThreadSecurity = NULL
pProcessSecurity = NULL
CommandLine = "cmd"
ModuleFileName = NULL
CreateProcessA
Timeout = INFINITE
hObject
WaitForSingleObject

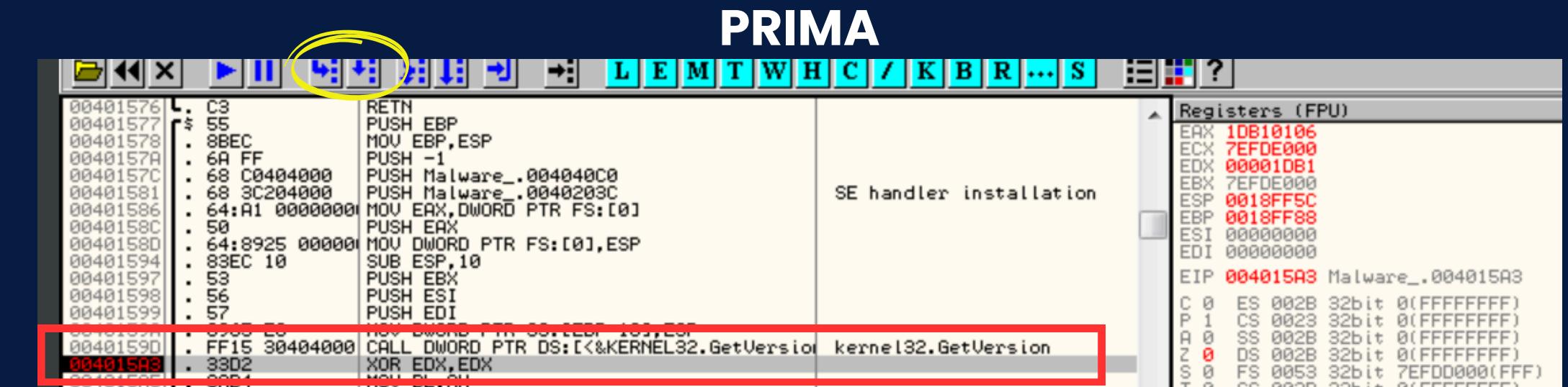
2

Inserite un breakpoint software all'indirizzo 004015A3. Qual è il valore del registro EDX?

(2) Eseguite a questo punto uno «step-into». Indicate qual è ora il valore del registro EDX (3) motivando la risposta (4). Che istruzione è stata eseguita? (5)

Il'indirizzo 004015A3 facciamo click con il tasto destro del mouse > Breakpoint > Toggle. A questo punto l'indirizzo verrà evidenziato in rosso per confermare la presenza di un breakpoint in quel punto. Fermando l'esecuzione del codice prima dell'istruzione XOR possiamo notare che

EDX = 00001DB1



A questo punto, mandiamo avanti l'esecuzione clickando su una delle due frecce evidenziate (step-into e step-over) nella figura sopra.

notiamo come il valore di EDX sia cambiato. In questo caso, dopo un'operazione di XOR, il valore del registro viene settato a 0 e infatti possiamo vederlo evidenziato in rosso nell'elenco laterale.

EAX=00000000



3

Inserite un secondo breakpoint all'indirizzo di memoria 004015AF. Qual è il valore del registro ECX? (6) Eseguite un step-into. Qual è ora il valore di ECX? (7) Spiegate quale istruzione è stata eseguita (8).

Nel dettaglio, l'istruzione esegue l'AND logico sui bit di ECX e del valore esadecimale OFF.

Per prima cosa portiamo entrambi i valori in formato binario e poi eseguiamo l'AND logico tra i bit.

Eseguendo l'AND logico tra i bit uno ad uno

Che in Esadecimale è 00000006

Ecco spiegato il valore di ECX dopo l'istruzione AND ECX, OFF

Registers (FPU)

EAX	10DB10106
ECX	10DB10106
EDX	00000001
EBX	7EFDE000
ESP	0018FF5C
EBP	0018FF88
ESI	00000000
EDI	00000000

EIP 004015AF Malware_.004015AF

C 0	ES	002B	32bit	0(FFFFFFF)
P 1	CS	0023	32bit	0(FFFFFFF)
A 0	SS	002B	32bit	0(FFFFFFF)
Z 1	DS	002B	32bit	0(FFFFFFF)
S 0	FS	0053	32bit	7EFDD000(FFF)
T 0	GS	002B	32bit	0(FFFFFFF)
D 0	O 0	LastErr	ERROR_SUCCESS	(000
EFL	00000246	(NO,NB,E,BE,NS,PE,		

The screenshot shows a debugger interface with the following details:

Registers (FPU):

EAX	1DB10106
ECX	00000006
EDX	00000001
EBX	7EFDE000
ESP	0018FF5C
EBP	0018FF88
ESI	00000000
EDI	00000000

Stack Dump:

C 0	ES	002B	32bit	0(FFFFFF)
P 1	CS	0023	32bit	0(FFFFFF)
A 0	SS	002B	32bit	0(FFFFFF)
Z 0	DS	002B	32bit	0(FFFFFF)
S 0	FS	0053	32bit	7EFDD000(FFF)

Assembly Code:

```
00401586: . 64:A1 00000000 MOV EAX,DWORD PTR FS:[0]
0040158C: . 50 PUSH EAX
0040158D: . 64:8925 000000 MOV DWORD PTR FS:[0],ESP
00401594: . B3 EC 10 SUB ESP,10
00401597: . B3 53 PUSH EBX
00401598: . B3 56 PUSH ESI
00401599: . B3 57 PUSH EDI
0040159A: . 8965 E8 MOV DWORD PTR SS:[EBP-18],ESP
0040159D: . FF15 30404000 CALL DWORD PTR DS:[<&KERNEL32.GetVersion] kernel32.GetVersion
004015A3: . 33D2 XOR EDX,EDX
004015A5: . B8 D4 MOV DL,AH
004015A7: . 8915 D4524000 MOV DWORD PTR DS:[405204],EDX
004015AD: . B8 C8 MOV ECX,EAX
004015AF: . B1 E1 FF000000 AND ECX,0FF
004015B5: . 890D D0524000 MOV DWORD PTR DS:[405200],ECX
004015BB: . C1E1 08 SHL ECX,8
004015BE: . 03C0 ADD ECX,EDX
```

ECX=1DB10106 = 11101101100010000000100000110

OFF = 11111111

AND ECX, OFF

4

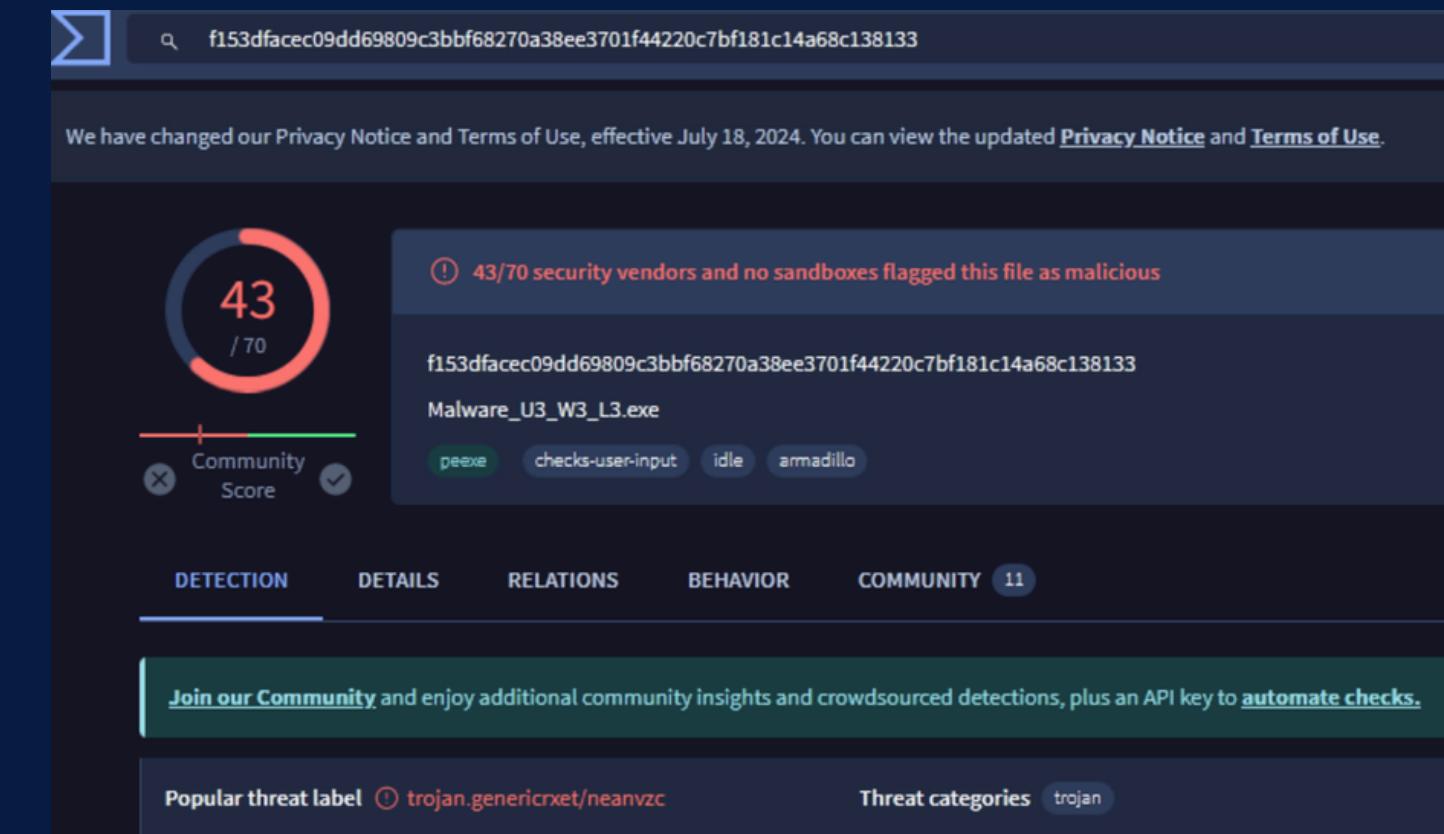
BONUS: spiegare a grandi linee il funzionamento del malware

Per fare delle considerazioni più precise riguardo il malware analizzato, abbiamo utilizzato il tool VirusTotal, che ci ha fornito tutte le informazioni necessarie in modo più chiaro rispetto al codice Assembly x86. Come mostrato nell'immagine seguente, il malware è un Trojan Horse. Studiando le diverse funzioni che utilizza, abbiamo osservato i seguenti comportamenti:

1. **Connessione:** tramite la funzione WSAsocketA, funzione dell'API Windows Sockets (Winsock), crea un socket per la comunicazione di rete.

2. **Persistenza:** Per ottenere la persistenza nel sistema, il malware utilizza `GetModuleFileNameA` per leggere il percorso del malware. Successivamente, aggiunge questo percorso a una chiave di registro di avvio, assicurando che il malware venga eseguito ogni volta che il sistema si avvia.

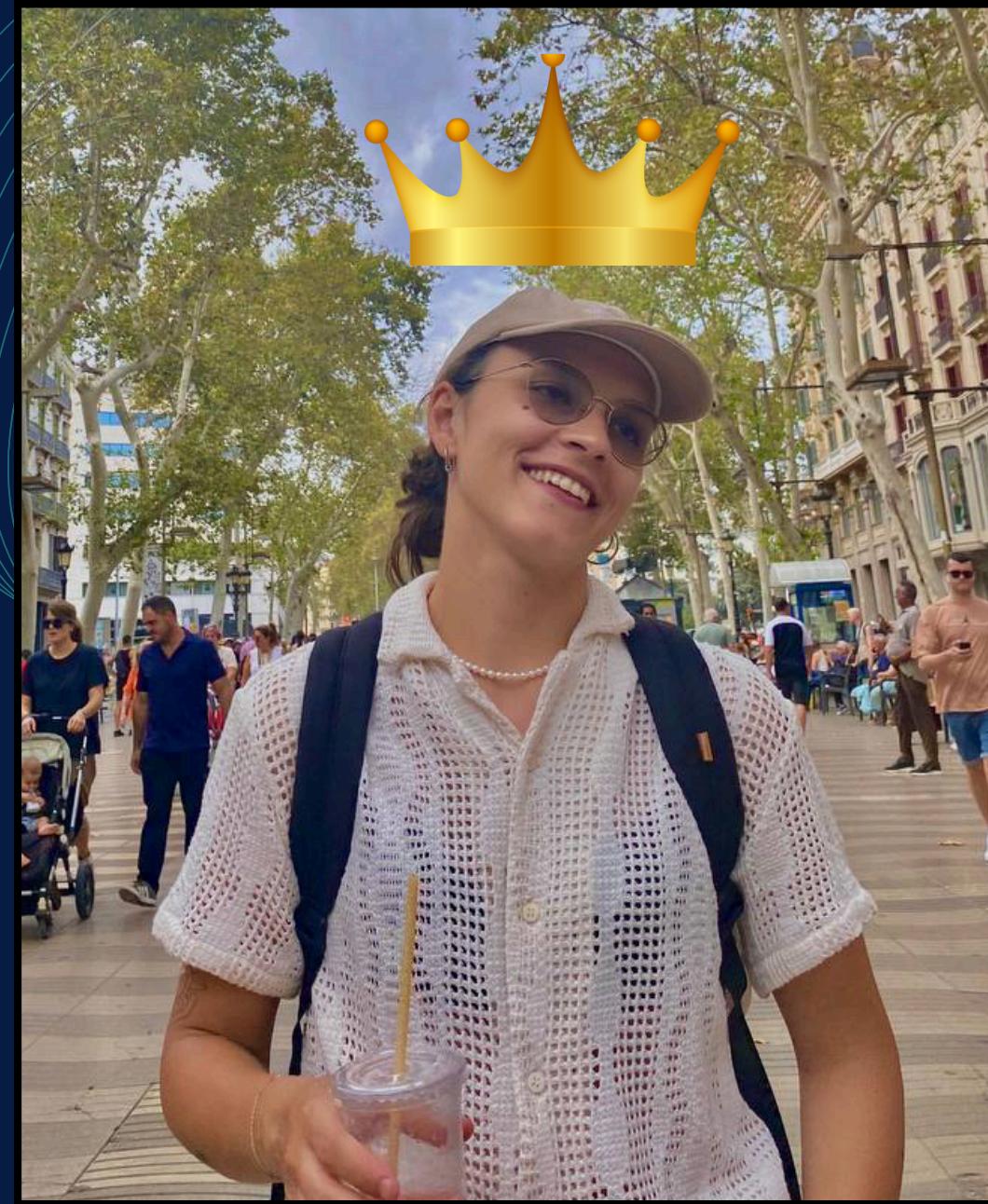
3. **Modifica delle porte:** è una funzione della libreria di rete che converte un numero di porta da formato host a formato network. "htons" sta per "host to network short". Questa funzione è particolarmente utile quando si lavora con i socket di rete, dove è necessario garantire che i numeri di porta siano in formato big-endian, che è il formato standard utilizzato nella comunicazione di rete.



Le librerie importate nel codice Assembly x86 del malware sono quelle mostrate di seguito.

Imports
+ KERNEL32.dll
+ WS2_32.dll

Our Team



Mara Dello Russo



ZhongShi Liu



Mario Marsicano



Andrè V



Anapaula Palacin



Roberta Mercadante

Thank you!