

Report Analisi Malware

Introduzione:

In questo report, analizziamo il comportamento di un malware utilizzando **OllyDbg**, un debugger di livello utente per Microsoft Windows. **OllyDbg** è uno strumento potente per il debugging di codice binario e assembly, che permette di monitorare l'esecuzione di programmi, ispezionare registri, stack e memoria, e inserire breakpoints per analizzare il flusso di esecuzione in modo dettagliato.



Traccia



Traccia:

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.

- 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)**
- 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)**
- 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)**.
- **BONUS:** spiegare a grandi linee il funzionamento del malware

Quesito 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?

All'indirizzo 0040106E, il malware esegue una chiamata alla funzione CreateProcessA della libreria kernel32.dll. Il valore del parametro "CommandLine" passato allo stack è "cmd", come si osserva all'indirizzo 00401067. Questo parametro indica che il malware sta tentando di avviare il prompt dei comandi di Windows.

00401056	. 52	PUSH EDI	pProcessInfo
00401057	. 8045 A8	LEA EAX,DWORD PTR SS:[EBP-58]	pStartupInfo
0040105A	. 50	PUSH EAX	CurrentDir = NULL
0040105B	. 6A 00	PUSH 0	pEnvironment = NULL
0040105D	. 6A 00	PUSH 0	CreationFlags = 0
0040105F	. 6A 00	PUSH 0	InheritHandles = TRUE
00401061	. 6A 01	PUSH 1	pThreadSecurity = NULL
00401063	. 6A 00	PUSH 0	pProcessSecurity = NULL
00401065	. 6A 00	PUSH 0	CommandLine = "cmd"
00401067	. 68 30504000	PUSH Malware_.00405030	ModuleFileName = NULL
0040106C	. 6A 00	PUSH 0	
0040106E	. FF15 04404000	CALL DWORD PTR DS:[<&KERNEL32.CreateProcessA>]	CreateProcessA

Quesito 2

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

All'indirizzo 004015A3, inseriamo un breakpoint software. Prima di eseguire lo step-into, il valore del registro EDX è 00001DB1. Dopo aver eseguito lo step-into, il valore del registro EDX diventa 00000000. Questo cambiamento è dovuto all'istruzione XOR EDX, EDX che azzerava il contenuto di EDX.

Prima

Registers (FPU)

EAX 1DB10106
ECX 7EFDE000
EDX 00001DB1
EBX 7EFDE000
ESP 0018FF5C
EBP 0018FF88
ESI 00000000
EDI 00000000
EIP 004015A3 Malware_.004015A3
C 0 ES 002B 32bit 0(FFFFFFFF)
P 1 CS 0023 32bit 0(FFFFFFFF)
A 0 SS 002B 32bit 0(FFFFFFFF)
Z 0 DS 002B 32bit 0(FFFFFFFF)
S 0 FS 0053 32bit 7EFD0000(FFF)
T 0 GS 002B 32bit 0(FFFFFFFF)
O 0
O 0 LastErr: ERROR_SUCCESS (00000000)
EFL 00000206 (NO,NB,NE,A,NS,PE,GE,G)
ST0 empty 0.0
ST1 empty 0.0
ST2 empty 0.0
ST3 empty 0.0
ST4 empty 0.0
ST5 empty 0.0
ST6 empty 0.0
ST7 empty 0.0
FST 0000 Cond 0 0 0 0 Err 0 0 0 0 0 0 0 (GT)
FCW 027F Prec NEAR,S8 Mask 1 1 1 1 1 1

Breakpoints

Address	Module	Active	Disassembly
004015A3	Malware_	Always	XOR EDX,EDX

Dopo

Registers (FPU)

EAX 1DB10106
ECX 7EFDE000
EDX 00000000
EBX 7EFDE000
ESP 0018FF5C
EBP 0018FF88
ESI 00000000
EDI 00000000
EIP 004015A5 Malware_.004015A5
C 0 ES 002B 32bit 0(FFFFFFFF)
P 1 CS 0023 32bit 0(FFFFFFFF)
A 0 SS 002B 32bit 0(FFFFFFFF)
Z 1 DS 002B 32bit 0(FFFFFFFF)
S 0 FS 0053 32bit 7EFD0000(FFF)
T 0 GS 002B 32bit 0(FFFFFFFF)
O 0
O 0 LastErr: ERROR_SUCCESS (00000000)
EFL 00010246 (NO,NB,E,BE,NS,PE,GE,LE)
ST0 empty 0.0
ST1 empty 0.0
ST2 empty 0.0
ST3 empty 0.0
ST4 empty 0.0
ST5 empty 0.0
ST6 empty 0.0
ST7 empty 0.0
FST 0000 Cond 0 0 0 0 Err 0 0 0 0 0 0 0 (GT)
FCW 027F Prec NEAR,S8 Mask 1 1 1 1 1 1

Quesito 3

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

All'indirizzo 004015AF, inseriamo un secondo breakpoint. Prima di eseguire lo step-into, il valore del registro ECX è 1DB10106. Dopo aver eseguito lo step-into, il valore del registro ECX diventa 00000006. Questo cambiamento è dovuto all'istruzione AND ECX, FF, che esegue un'operazione logica AND tra ECX e il valore esadecimale FF, mantenendo solo i bit meno significativi.

Prima

```

ECX 1DB10106
ECX 1DB10106
EDX 00000001
EBX 7EFDE000
ESP 0018FF5C
EBP 0018FF58
ESI 00000000
EDI 00000000
EIP 004015AF Malware_.004015AF
C 0 ES 002B 32bit 0(FFFFFFFF)
P 1 CS 0023 32bit 0(FFFFFFFF)
A 0 SS 002B 32bit 0(FFFFFFFF)
Z 1 DS 002B 32bit 0(FFFFFFFF)
S 0 FS 0053 32bit 7EFD0000(FFF)
T 0 GS 002B 32bit 0(FFFFFFFF)
D 0
O 0 LastErr ERROR_SUCCESS (00000000)
EFL 00000246 (NO,NB,E,BE,NS,PE,GE,LE)
ST0 empty 0.0
ST1 empty 0.0
ST2 empty 0.0
ST3 empty 0.0
ST4 empty 0.0
ST5 empty 0.0
ST6 empty 0.0
ST7 empty 0.0
FST 0000 Cond 0 0 0 0 Err 0 0 0 0 0 0 0 0 (GT)
FCW 027F Prec 00000000 Mask 1 1 1 1 1 1
```

B Breakpoints			
Address	Module	Active	Disassembly
004015A3	Malware_	Always	XOR EDX,EDX
004015AF	Malware_	Always	AND ECX,0FF

Dopo

00401576 C9 LEAVE

00401576 C8 RETN

00401577 55 PUSH EBP

00401578 8BEC MOV EBP,ESP

0040157A 6A FF PUSH -1

0040157C 68 00404000 PUSH Malware_.00404000

00401581 68 3C204000 PUSH Malware_.0040203C

00401586 64:01 00000000 MOV EAX,0

00401590 50 PUSH EAX

00401590 64:01 00000000 MOV EDI,0

00401594 33EC SUB ESP,10

00401597 53 PUSH EBX

00401598 56 PUSH ESI

00401599 57 PUSH EDI

0040159A 8B45 MOV EAX,DWORD PTR DS:[EBP-10],ESP

0040159A FF15 30404000 CALL DWORD PTR DS:[<&KERNEL32.GetVersion

0040159D 33C2 XOR EDX,EDX

004015A5 8D44 MOV DL,AH

004015A7 8B45 MOV EAX,DWORD PTR DS:[4052D4],EDX

004015A8 8BC8 MOV ECX,EAX

004015AF 01E1 FF000000 AND ECX,0FF

004015B0 8B45 MOV EAX,DWORD PTR DS:[4052D0],ECX

004015B5 C1E1 08 SHL ECX,8

004015B6 8BCA ADD ECX,EDX

004015C0 8B45 MOV EAX,DWORD PTR DS:[4052CC],ECX

004015C6 C1E8 10 SHR EAX,10

004015C9 8B45 MOV EAX,DWORD PTR DS:[4052C0],EAX

004015CE 6A 00 PUSH 0

004015D0 E8 33090000 CALL Malware_.00401F08

004015D5 59 POP ECX

004015D6 85C0 TEST EAX,EAX

004015D8 75 08 JNZ SHORT Malware_.004015E2

004015DA 6A 1C PUSH 1C

004015DC E8 9A000000 CALL Malware_.00401678

004015E1 59 POP ECX

004015E2 8B45 MOV EAX,DWORD PTR SS:[EBP-4],0

004015E6 E8 72070000 CALL Malware_.00401D5D

004015E8 FF15 2C404000 CALL DWORD PTR DS:[<&KERNEL32.GetComm

004015F1 A3 D0574000 MOV DWORD PTR DS:[4057D0],EAX

004015F6 E8 30060000 CALL Malware_.00401C2B

004015FB A3 D0524000 MOV DWORD PTR DS:[4052D0],EAX

00401600 E8 D9030000 CALL Malware_.004019DE

00401605 E8 1B030000 CALL Malware_.00401925

SE handler: installation

kernel32.GetVersion

CGetCommandLineA

ECX: 00000006

DS:[004052D0]=00000000

Malware_-(ModuleEntryPoint)+3FE

Registers (FPU)

ECX 1DB10106

ECX 00000006

EDX 00000001

EBX 7EFDE000

ESP 0018FF5C

EBP 0018FF58

ESI 00000000

EDI 00000000

EIP 004015B5 Malware_.004015B5

C 0 ES 002B 32bit 0(FFFFFFFF)

P 1 CS 0023 32bit 0(FFFFFFFF)

A 0 SS 002B 32bit 0(FFFFFFFF)

Z 0 DS 002B 32bit 0(FFFFFFFF)

S 0 FS 0053 32bit 7EFD0000(FFF)

T 0 GS 002B 32bit 0(FFFFFFFF)

D 0

O 0 LastErr ERROR_SUCCESS (00000000)

EFL 00010206 (NO,NB,NE,A,NS,PE,GE,GI)

ST0 empty 0.0

ST1 empty 0.0

ST2 empty 0.0

ST3 empty 0.0

ST4 empty 0.0

ST5 empty 0.0

ST6 empty 0.0

ST7 empty 0.0

FST 0000 Cond 0 0 0 0 Err 0 0 0 0 0 0 0 0 (GT)

FCW 027F Prec 00000000 Mask 1 1 1 1 1 1

Quesito bonus

Funzionamento generale del malware:

1. Creazione di Processi:

- Il malware utilizza la funzione `CreateProcessA` per creare nuovi processi. In questo caso, crea un'istanza del Command Prompt (`cmd`). Ciò consente al malware di eseguire comandi di sistema arbitrari, che possono includere il lancio di ulteriori payloads, l'esecuzione di comandi dannosi o la manipolazione di file e configurazioni di sistema.

2. Manipolazione dei Registri:

- Le istruzioni come `XOR EDX, EDX` e `AND ECX, FF` sono utilizzate per manipolare i registri. `XOR EDX, EDX` azzerava il registro `EDX`, mentre `AND ECX, FF` isola l'ultimo byte di `ECX`, azzerando i restanti bit. Queste tecniche sono utilizzate per preparare i valori nei registri per ulteriori operazioni e per evitare il rilevamento.

3. Utilizzo di Funzioni di Sistema:

- Il malware fa uso di diverse funzioni di sistema, tra cui `GetVersion`, `GetCommandLineA`, `WaitForSingleObject`, ecc. Queste funzioni gli permettono di raccogliere informazioni sul sistema, eseguire comandi specifici e sincronizzarsi con altri processi.

4. Networking:

- Funzioni come `WSAStartup`, `WSASocketA`, `connect`, `closesocket`, ecc., indicano che il malware tenta di stabilire connessioni di rete. Queste connessioni possono essere utilizzate per comunicare con un server di comando e controllo (C&C), scaricare ulteriori componenti del malware o esfiltrare dati dal sistema compromesso.

5. Persistenza:

- Il malware può includere meccanismi per garantirsi la persistenza sul sistema infetto. Ciò può includere la modifica di chiavi di registro, la creazione di task schedulati o l'utilizzo di tecniche di iniezione di codice per assicurarsi che venga eseguito anche dopo un riavvio del sistema.
-