

S11/L3 Esercizio 06/12/2023

Nell'esercizio di oggi ci viene richiesto, in base al malware fornito, di rispondere ai seguenti quesiti usando OllyDBG:

- Qual è il valore del parametro «CommandLine» che viene passato sullo stack all'indirizzo 0040106E
- Inserire un breakpoint software all'indirizzo 004015A3 e dire qual'è il valore del registro EDX
- Eseguire uno «step-into» e indicare a questo punto qual è ora il valore del registro EDX
- Motivare la risposta
- Spiegare quale istruzione è stata eseguita
- Inserire un secondo breakpoint all'indirizzo di memoria 004015AF e dire qual'è il valore del registro ECX
- Eseguite uno step-into e dire qual è ora il valore di ECX
- Spiegare quale istruzione è stata eseguita
- BONUS: spiegare a grandi linee il funzionamento del malware

Quesito 1

Come possiamo vedere dall'immagine il valore del parametro «CommandLine» che viene passato sullo stack è «cmd»

00401056	52	PUSH EDI	pProcessInfo
00401057	3D45 A8	LEA EAX, DWORD PTR SS:[EBP-58]	pStartupInfo
00401059	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	
00401067	68 30504000	PUSH Malware_.00405030	CommandLine = "cmd"
0040106C	6A 00	PUSH 0	ModuleFileName = NULL
0040106E	FF15 04404000	CALL DWORD PTR DS:[<&KERNEL32.CreateProcessA	CreateProcessA
00401074	8945 EC	MOV DWORD PTR SS:[EBP-14], EAX	Timeout = INFINITE
00401077	6A FF	PUSH -1	hObject:
00401079	8B4D F0	MOV ECX, DWORD PTR SS:[EBP-10]	WaitForSingleObject
0040107C	51	PUSH ECX	
0040107D	FF15 00404000	CALL DWORD PTR DS:[<&KERNEL32.WaitForSingleObject	
00401082	5D	POP ESI	


Quesito 2

Dopo aver inserito il breakpoint software attraverso i comandi <<tasto dx/Breakpoint/Toogle>> e aver avviato il programma possiamo rilevare il valore EDX 00000A28

00401577	53	PUSH EBX	
00401578	56	PUSH ESI	
00401579	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	8AD4	MOV DL, AH	
004015A7	8915 04524000	MOV DWORD PTR DS:[4052D4], EDX	
004015AD	8BC8	MOV ECX, EAX	

Registers (FPU)	
EAX	0A280105
ECX	7FDD0000
EDX	00000A28
EBX	7FDD0000
ESP	0012FF94

Quesito 3-4-5

Eseguendo uno “step-into” con il comando  il valore EDX viene azzerato, questo perché l’istruzione “XOR EDX, EDX” è comunemente utilizzata per azzerare il registro confrontando con la funzione “XOR” tutti i bit presenti dando 0 come risultato nel caso risultino uguali e 1 nel caso fossero diversi

```
004015A5 | . 33D2 | XOR EDX,EDX
004015A6 | . 8AD4 | MOV DL,AH
004015A7 | . 8915 D4524000 | MOV DWORD PTR DS:[4052D4],EDX

EDX 00000000
```

Quesito 6

Dopo aver inserito il breakpoint software possiamo rilevare il valore ECX 0A280105

```
004015AD | . 8BC8 | MOV ECX,EAX
004015AE | . 81E1 FF000000 | AND ECX,0FF
004015B5 | . 890D D0524000 | MOV DWORD PTR DS:[4052D0],ECX

Registers (FPU)
EAX 0A280105
ECX 0A280105
EDX 00000001
```

Quesito 7-8

Eseguendo uno “step-into” il valore ECX diventa 00000005. L’istruzione AND ECX, 0FF “maschera” tutti i bit tranne gli ultimi 8 bit in ECX, il risultato sarà che solo l’ultimo byte di ECX sarà mantenuto, mentre gli altri byte verranno azzerati.

```
004015AF | . 81E1 FF000000 | AND ECX,0FF
004015B5 | . 890D D0524000 | MOV DWORD PTR DS:[4052D0],ECX

ECX 00000005
```

Quesito BONUS

Analizzando altre linee di codice troviamo la dicitura “call” su Socket Connect quindi potremmo ipotizzare che ci troviamo davanti ad una backdoor client

<pre>00401317 . 68 0F200000 PUSH 20F 0040131C . FF15 00404000 CALL DWORD PTR DS:[<MS2_32.#9>] 00401322 . 66:8905 36FEF MOV WORD PTR SS:[EBP-1CA],AX 00401329 . 66:C785 34FEF MOV WORD PTR SS:[EBP-1CC],2 00401332 . 6A 10 PUSH 10 00401334 . 8D85 34FEFFFF LEA EAX,DWORD PTR SS:[EBP-1CC] 0040133A . 50 PUSH EAX 0040133B . 8B8D FCFCFFFF MOV ECX,DWORD PTR SS:[EBP-304] 00401341 . 51 PUSH ECX 00401342 . FF15 00404000 CALL DWORD PTR DS:[<MS2_32.#4>] 00401348 . 8985 4CFEFFFF MOV DWORD PTR SS:[EBP-1B4],EAX 0040134E . 83BD 4CFEFFFF CMP DWORD PTR SS:[EBP-1B4],-1 00401355 . 75 23 JNZ SHORT Malware_.0040137A 00401357 . 8B95 FCFCFFFF MOV EDX,DWORD PTR SS:[EBP-304] 0040135D . 52 PUSH EDX 0040135E . FF15 00404000 CALL DWORD PTR DS:[<MS2_32.#3>] 00401364 . FF15 00404000 CALL DWORD PTR DS:[<MS2_32.#116>] 0040136A . 68 30750000 PUSH 7530 0040136F . FF15 00404000 CALL DWORD PTR DS:[<KERNEL32.Sleep>] 00401375 . ^E9 D2FEFFFF JMP Malware_.0040124C 0040137A . 8B85 FCFCFFFF MOV EAX,DWORD PTR SS:[EBP-304] 00401380 . 50 PUSH EAX 00401381 . 83EC 10 SUB ESP,10 00401384 . 8BCC MOV ECX,ESP 00401386 . 8B95 34FEFFFF MOV EDX,DWORD PTR SS:[EBP-1CC] 0040138C . 8911 MOV DWORD PTR DS:[ECX],EDX 0040138E . 8B85 38FEFFFF MOV EAX,DWORD PTR SS:[EBP-1C8] 00401394 . 8941 04 MOV DWORD PTR DS:[ECX+4],EAX 00401397 . 8B95 3CFEFFFF MOV EDX,DWORD PTR SS:[EBP-1C4] 0040139D . 8951 08 MOV DWORD PTR DS:[ECX+8],EDX 004013A0 . 8B85 40FEFFFF MOV EAX,DWORD PTR SS:[EBP-1C0] 004013A6 . 8941 0C MOV DWORD PTR DS:[ECX+C],EAX 004013A9 . E8 52FCFFFF CALL Malware_.00401000</pre>	<pre>[NetShort = 20F ntohs AddrLen = 10 (16.) pSockAddr Socket connect Socket closesocket WSACleanup Timeout = 30000. ms Sleep Malware_.00401000</pre>
--	--