ANALISI STATICA AVANZATA CON TOOL IDA PRO

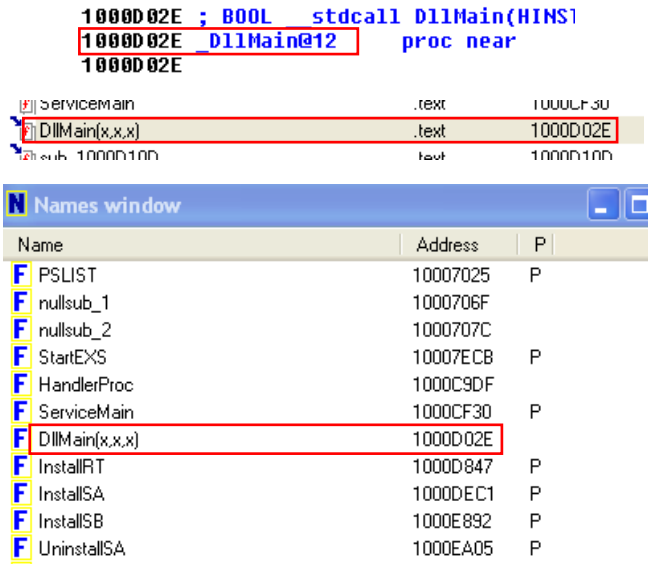
Traccia:

Lo scopo dell'esercizio di oggi è di acquisire esperienza con IDA, un tool fondamentale per l'analisi statica.
A tal proposito, con riferimento al malware chiamato «**Malware_U3_W3_L2**» presente all'interno della cartella «**Esercizio_Pratico_U3_W3_L2**» sul Desktop della macchina virtuale dedicata all'analisi dei malware, rispondere ai seguenti quesiti, utilizzando IDA Pro.

1. Individuare **l'indirizzo** della funzione **DLLMain** (così com'è, in esadecimale)
2. Dalla scheda «imports» individuare la funzione «**gethostbyname**». Qual è l'indirizzo dell'import?
3. Quante sono le variabili locali della **funzione** alla locazione di memoria 0x10001656?
4. Quanti sono, invece, i parametri della funzione sopra?
5. Inserire altre considerazioni macro livello sul malware (comportamento)

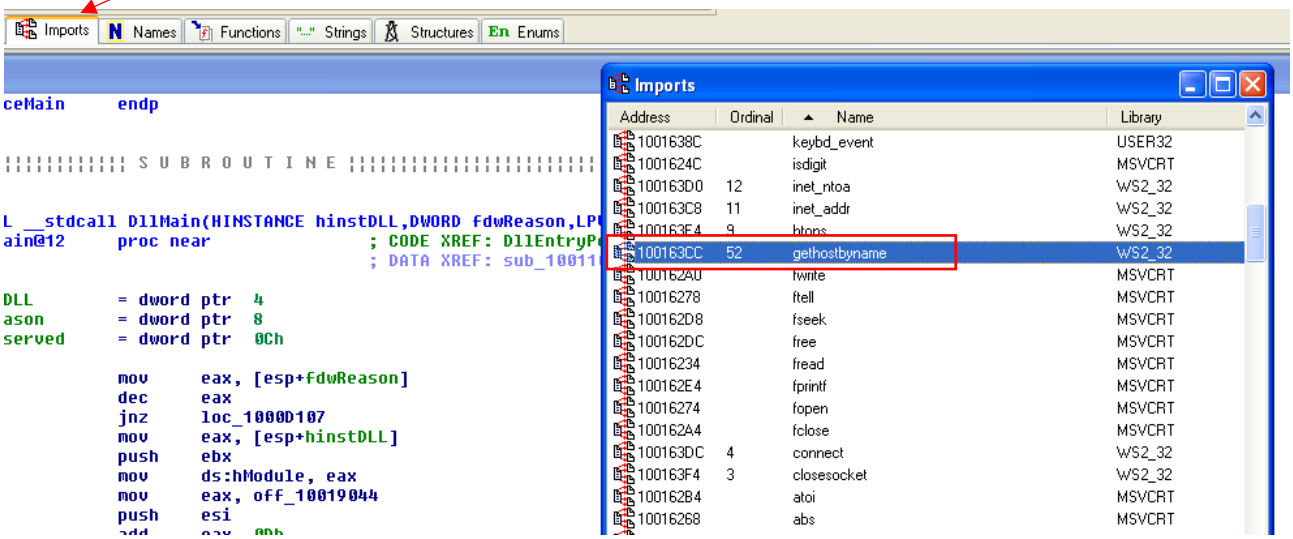
1.

Per facilitare la ricerca, possiamo utilizzare il jump to function che ci permette di cercare direttamente il nome della funzione; altrimenti possiamo scorrere fino a quando non vediamo la funzione nelle finestre IDA view o Names window:



2.

Dalla scheda Imports, scorriamo fino a trovare la funzione richiesta; sulla prima colonna di sinistra c'è associato l'indirizzo di memoria:



3.

Per capire quali sono le variabili locali della funzione, dobbiamo andare a cercare quelle con valore negativo perché il tool IDA le distingue dai parametri mettendole con un offset negativo:

```
.text:10001656
.text:10001656 ; ╔═══════════════ S U B R O U T I N E ═══════════════════════════════════════
.text:10001656
.text:10001656
.text:10001656 ; DWORD __stdcall sub_10001656(LPVOID)
.text:10001656 sub_10001656    proc near               ; DATA XREF: DllMain(x,x,x)+C8↓o
.text:10001656
.text:10001656 var_675         = byte ptr -675h
.text:10001656 var_674         = dword ptr -674h
.text:10001656 hModule         = dword ptr -670h
.text:10001656 timeout         = timeval ptr -66Ch
.text:10001656 name            = sockaddr ptr -664h
.text:10001656 var_654         = word ptr -654h
.text:10001656 in              = in_addr ptr -650h
.text:10001656 Parameter       = byte ptr -644h
.text:10001656 CommandLine     = byte ptr -63Fh
.text:10001656 Data            = byte ptr -638h
.text:10001656 var_544         = dword ptr -544h
.text:10001656 var_50C         = dword ptr -50Ch
.text:10001656 var_500         = dword ptr -500h
.text:10001656 var_4FC         = dword ptr -4FCh
.text:10001656 readfds         = fd_set ptr -4BCh
.text:10001656 phkResult       = HKEY__ ptr -3B8h
.text:10001656 var_3B0         = dword ptr -3B0h
.text:10001656 var_1A4         = dword ptr -1A4h
.text:10001656 var_194         = dword ptr -194h
.text:10001656 WSAData         = WSAData ptr -190h
.text:10001656 arg_0           = dword ptr  4
.text:10001656
.text:10001656                 sub     esp, 678h
```

4.

Stesso discorso per quanto riguarda l'individuazione dei parametri, con la differenza che IDA li mostra con offset positivo:

```
.text:10001656
.text:10001656 ; ╔═══════════════ S U B R O U T I N E ═══════════════════════════════════════
.text:10001656
.text:10001656
.text:10001656 ; DWORD __stdcall sub_10001656(LPVOID)
.text:10001656 sub_10001656    proc near               ; DATA XREF: DllMain(x,x,x)+C8↓o
.text:10001656
.text:10001656 var_675         = byte ptr -675h
.text:10001656 var_674         = dword ptr -674h
.text:10001656 hModule         = dword ptr -670h
.text:10001656 timeout         = timeval ptr -66Ch
.text:10001656 name            = sockaddr ptr -664h
.text:10001656 var_654         = word ptr -654h
.text:10001656 in              = in_addr ptr -650h
.text:10001656 Parameter       = byte ptr -644h
.text:10001656 CommandLine     = byte ptr -63Fh
.text:10001656 Data            = byte ptr -638h
.text:10001656 var_544         = dword ptr -544h
.text:10001656 var_50C         = dword ptr -50Ch
.text:10001656 var_500         = dword ptr -500h
.text:10001656 var_4FC         = dword ptr -4FCh
.text:10001656 readfds         = fd_set ptr -4BCh
.text:10001656 phkResult       = HKEY__ ptr -3B8h
.text:10001656 var_3B0         = dword ptr -3B0h
.text:10001656 var_1A4         = dword ptr -1A4h
.text:10001656 var_194         = dword ptr -194h
.text:10001656 WSAData         = WSAData ptr -190h
.text:10001656 arg_0           = dword ptr  4
.text:10001656
.text:10001656                 sub     esp, 678h
```

5.

Cercando tra le stringhe, mi sono imbattuto in questa in basso che diceva chiaramente che il malware è una backdoor:

```
\r\n[1] Enter Current Directory '%s'
\r\n\r\n**********\r\n[BackDoor Server Update Setup]\r\n*...
-warn
```

Cercando anche su Virus Total ho avuto la conferma che l'eseguibile crea una backdoor:

AhnLab-V3	Backdoor/Win32.Agent.R9408	Alibaba	Backdoor:Win32/Idicaf.9f3a5556
ALYac	Backdoor.XIW	Antiy-AVL	Trojan[Backdoor]/Win32.Agent
Arcabit	Backdoor.XIW	Avast	Win32:Agent-OLH [Trj]
AVG	Win32:Agent-OLH [Trj]	Avira (no cloud)	BDS/Agen.twe.134160
BitDefender	Backdoor.XIW	ClamAV	Win.Trojan.Idicaf-9937585-0
Cynet	Malicious (score: 100)	Cyren	W32/Backdoor.LTKC-2937
DrWeb	BackDoor.Siggen.47995	Elastic	Malicious (high Confidence)
Emsisoft	Backdoor.XIW (B)	eScan	Backdoor.XIW
ESET-NOD32	A Variant Of Win32/Idicaf.C	Fortinet	W32/Idicaf.KItr
GData	Backdoor.XIW	Google	Detected