

CONSEGNA S11/L2

Analisi statica avanzata con IDA

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
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?

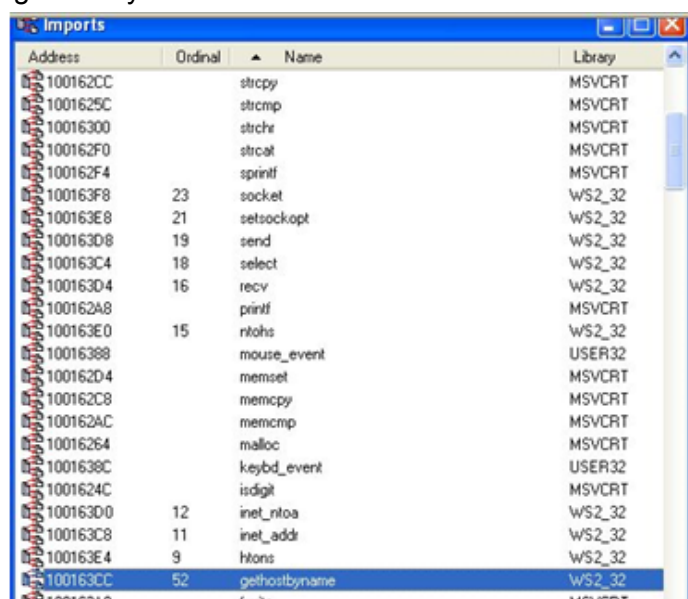
1 - Individuare l'indirizzo della funzione DLLMain

Per trovare l'indirizzo della funzione DllMain carichiamo il .exe in IDA Pro. Successivamente premiamo la barra per passare nella modalità testuale e trovare l'indirizzo della funzione main che è: **1000D02E**

```
.text:1000D02E  
.text:1000D02E ; BOOL __stdcall DllMain(HINSTANCE hinstDLL,DWORD fdwReason,LPUVOID lpvReserved)  
.text:1000D02E _DllMain@12      proc near                                ; CODE XREF: DllEntryPoint+4B↓p  
.text:1000D02E                                     ; DATA XREF: sub_100110FF+2D↓o  
.text:1000D02E hinstDLL      = dword ptr 4
```

2 - Indirizzo dell'import "gethostbyname"

Basterà aprire la finestra degli "imports" da IDA Pro e localizzare la funzione ricercata. "gethostbyname" si trova all'indirizzo **100163CC**



| Address | Ordinal | Name | Library |
|----------|---------|---------------|---------|
| 100162CC | | strcpy | MSVCRT |
| 1001629C | | strcmp | MSVCRT |
| 10016300 | | strchr | MSVCRT |
| 100162F0 | | strcat | MSVCRT |
| 100162F4 | | sprintf | MSVCRT |
| 100163F8 | 23 | socket | WS2_32 |
| 100163E8 | 21 | setsockopt | WS2_32 |
| 100163D8 | 19 | send | WS2_32 |
| 100163C4 | 18 | select | WS2_32 |
| 100163D4 | 16 | recv | WS2_32 |
| 100162A8 | | printf | MSVCRT |
| 100163E0 | 15 | ntohs | WS2_32 |
| 10016388 | | mouse_event | USER32 |
| 100162D4 | | memset | MSVCRT |
| 100162C8 | | memcpy | MSVCRT |
| 100162AC | | memcmp | MSVCRT |
| 10016264 | | malloc | MSVCRT |
| 1001638C | | keybd_event | USER32 |
| 1001624C | | isdigit | MSVCRT |
| 100163D0 | 12 | inet_ntoa | WS2_32 |
| 100163C8 | 11 | inet_addr | WS2_32 |
| 100163E4 | 9 | htonl | WS2_32 |
| 100163CC | 52 | gethostbyname | WS2_32 |
| 100162A0 | | function | MSVCRT |

3 - Quante variabili locali della funzione alla locazione di memoria 0x10001656

Ci spostiamo all'indirizzo ricercato tramite la ricerca o la barra laterale.

A questo indirizzo troviamo **20 variabili** con offset negativo rispetto ad EBP.

```
.text:10001656 : SUBROUTINE
.text:10001656
.text:10001656
.text:10001656 ; DWORD __stdcall sub_10001656(LPVOID)
.text:10001656 sub_10001656 proc near ; DATA XREF:
.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 WSADData = WSADData ptr -190h
.text:10001656 arg_0 = dword ptr 4
.text:10001656
```

4 - Quanti parametri presenti nella funzione sopra

Dalla stessa figura notiamo **un solo argomento** passato alla funzione, avente offset positivo rispetto ad ABP. Ida ha chiamato questo parametro "arg_0"