

# S11/L2 Epicode Cybersecurity

Analisi statica con IDA  
Pro





Nell'esercizio di oggi ci viene richiesto di analizzare un codice malware utilizzando un nuovo programma, IDA Pro, e di:

- individuare l'indirizzo della funzione DLLMain;
- individuare l'indirizzo dell'import gethostbyname dalla scheda "Imports";
- quantificare le variabili presenti alla locazione di memoria 0x10001656;
- quantificare i parametri della locazione indicata sopra.

Per la prima parte dell'esercizio, avviamo IDA Pro e, nella barra di ricerca, inseriamo DLLMain, che ci condurrà direttamente ad esso; possiamo vedere così che la funzione si trova all'indirizzo "**1000D02E**".

```
.text:1000D02E
.text:1000D02E ; ===== S U B R O U T I N E =====
.text:1000D02E
.text:1000D02E
.text:1000D02E ; BOOL __stdcall DllMain(HINSTANCE hinstDLL, DWORD fdwReason, LPVOID lpvReserved)
.text:1000D02E _DllMain@12      proc near                ; CODE XREF: DllEntryPoint+4B↓p
.text:1000D02E                                     ; DATA XREF: sub_100110FF+2D↓o
.text:1000D02E
```

La seconda parte dell'esercizio ci richiede invece di andare nella scheda "Imports" e trovare la funzione 'gethostbyname' e il suo indirizzo. Andiamo quindi nella scheda richiesta e, tramite la barra di ricerca, cerchiamo la funzione, che scopriamo trovarsi all'indirizzo **"100163CC"**.

Address	Ordinal	Name	Library
 100163C4	18	select	WS2_32
 100163C8	11	inet_addr	WS2_32
 100163CC	52	gethostbyname	WS2_32
 100163D0	12	inet_ntoa	WS2_32

La terza consegna ci richiede di quantificare il numero di variabili presenti alla locazione di memoria “10001656”.

Raggiungendo l'indirizzo, scopriamo che esso è composto da **23 variabili**, riconoscibili per l'offset negativo rispetto ad EBP.

```
.text:10001656 ; ===== SUBROUTINE =====
.text:10001656
.text:10001656
.text:10001656 ; DWORD __stdcall sub_10001656(LPU0ID)
.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 hLibModule = dword ptr -670h
.text:10001656 timeout = timeval ptr -66Ch
.text:10001656 name = sockaddr ptr -664h
.text:10001656 var_654 = word ptr -654h
.text:10001656 Dst = dword ptr -650h
.text:10001656 Parameter = byte ptr -644h
.text:10001656 var_640 = byte ptr -640h
.text:10001656 CommandLine = byte ptr -63Fh
```

```
.text:10001656 Source = byte ptr -63Dh
.text:10001656 Data = byte ptr -638h
.text:10001656 var_637 = byte ptr -637h
.text:10001656 var_544 = dword ptr -544h
.text:10001656 var_50C = dword ptr -50Ch
.text:10001656 var_500 = dword ptr -500h
.text:10001656 Buf2 = byte ptr -4FCh
.text:10001656 readfds = fd_set ptr -4BCh
.text:10001656 phkResult = byte 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
```

Per l'ultima consegna dell'esercizio ci viene richiesto di quantificare, invece della variabili, i parametri presenti nella stessa locazione (10001656). Analizzando gli argomenti, constatiamo che è presente **un solo parametro**, riconoscibile in questo caso per l'offset positivo rispetto ad EBP.

```
.text:10001656 Source          = byte ptr -63Dh
.text:10001656 Data            = byte ptr -638h
.text:10001656 var_637         = byte ptr -637h
.text:10001656 var_544         = dword ptr -544h
.text:10001656 var_50C         = dword ptr -50Ch
.text:10001656 var_500         = dword ptr -500h
.text:10001656 Buf2           = byte ptr -4FCh
.text:10001656 readfds        = fd_set ptr -48Ch
.text:10001656 phkResult      = byte 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
```