EPICODE

EPICODE

CYBERSECURITY COURSE

BY MAX ALDROVANDI

17/05/2024

Web-site: https://epicode.com/it

Locality:

Via dei Magazzini Generali,
6 Roma, Lazio 00154, IT

PRACTICE EXERCISE S11/L2

Track:

The purpose of today's exercise is to gain experience with IDA, a fundamental tool for statistical analysis.

In this regard, with reference to the malware named "**Malware_U3_W3_L2**" found within the folder "**Exercise_Practical_U3_W3_L2**" on the Desktop of the virtual machine dedicated to malware analysis, answer the following questions, using IDA Pro.

1. Locate the address of the DLLMain function (as is, in hexadecimal)

2. From the "**imports**" tab, locate the function "**gethostbyname**". What is the address of the import? What does the function do?

3. How many variables are local to the function at **memory location 0x10001656**?

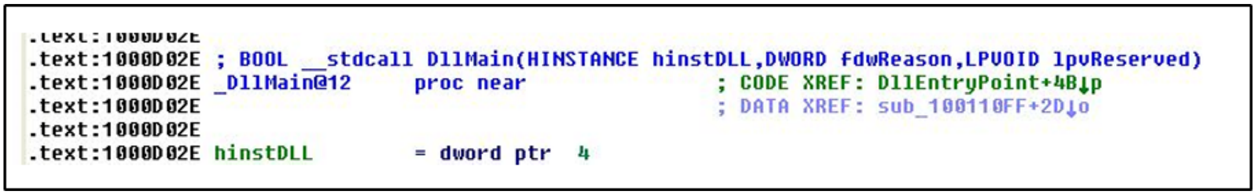
4. How many, on the other hand, are the parameters of the function above?

5. Insert other macro-level considerations about malware (behavior)

Solution

Locating the address of the DLLMain function

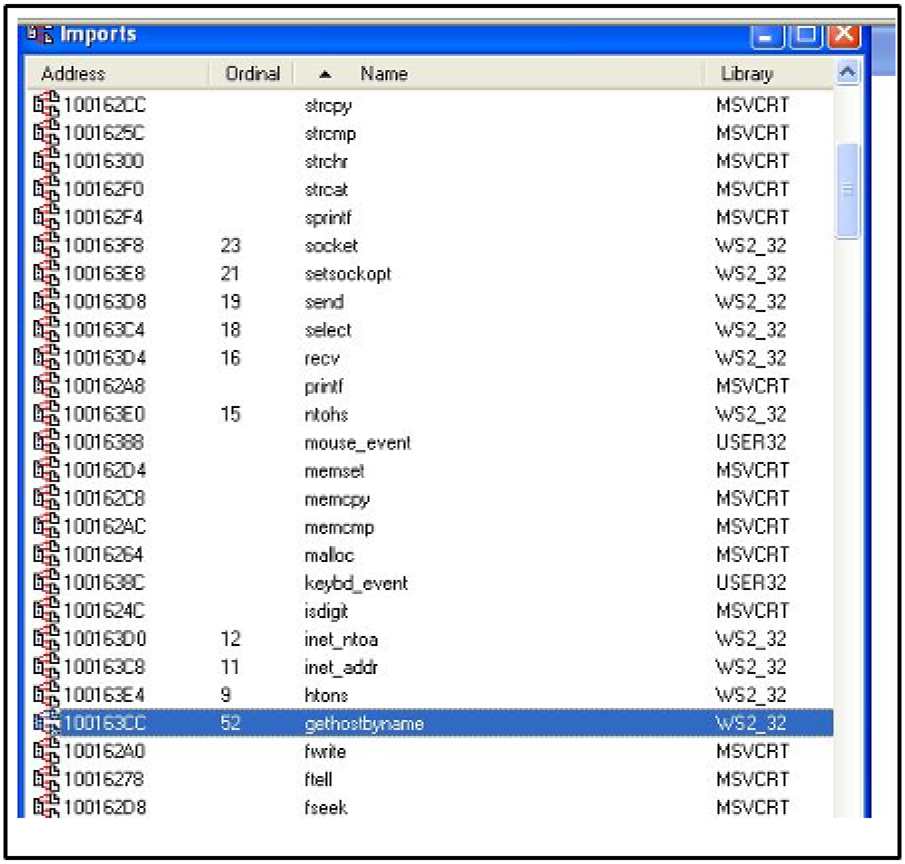
In order to find the address of the DllMain function, we load the executable into IDA Pro. Once done, we press the slash to switch to text mode and retrieve the address of the main function which will be: **1000D02E**



```
.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
.text:1000D02E hinstDLL        = dword ptr  4
```

The address of the "gethostbyname" import?

Let's open the "imports" window from IDA Pro, and locate the function we are looking for. "**gesthostbyname**" is at address **100163CC**, as shown in the figure:



Address	Ordinal	▲ Name	Library
100162CC		strcpy	MSVCRT
1001625C		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	htons	WS2_32
100163CC	52	gethostbyname	WS2_32
100162A0		fwrite	MSVCRT
10016278		ftell	MSVCRT
100162D8		fseek	MSVCRT

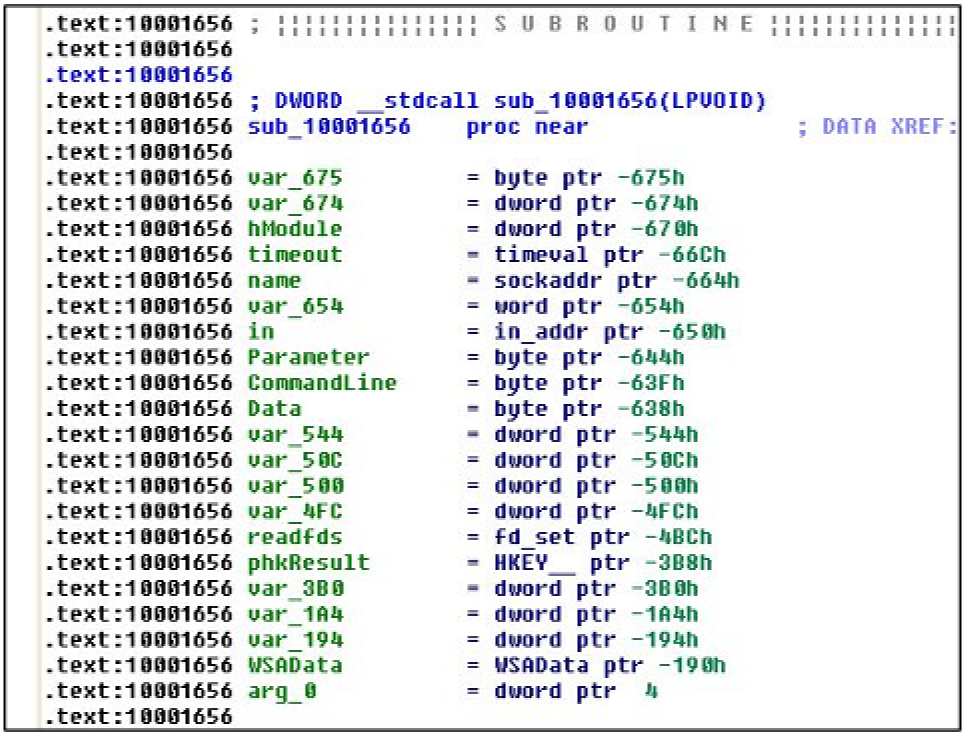
Local variables and argument of the function to the memory location 10001656

First we need to move to the address searched via the search or sidebar.

At this address we find **20 variables** with **negative offset** from **EBP**.

We can also see only one argument passed to the function, having **positive offset** from **EBP**.

IDA named this parameter "**arg_0.**"



```
.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:
.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
```