# S11-L2

by Hmich Otman & Andreoli Michael

The purpose of today's exercise is to gain experience with IDA, a fundamental tool for static analysis. In this regard, referring to the malware called "**Malware_U3_W3_L2**" located in the "**Esercizio_Pratico_U3_W3_L2**" folder on the Desktop of the virtual machine dedicated to malware analysis, answer the following questions using IDA Pro:

1. Identify the address of the **DLLMain** function (as it is, in hexadecimal).

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

3. How many local variables are there in the function at memory location 0x10001656?

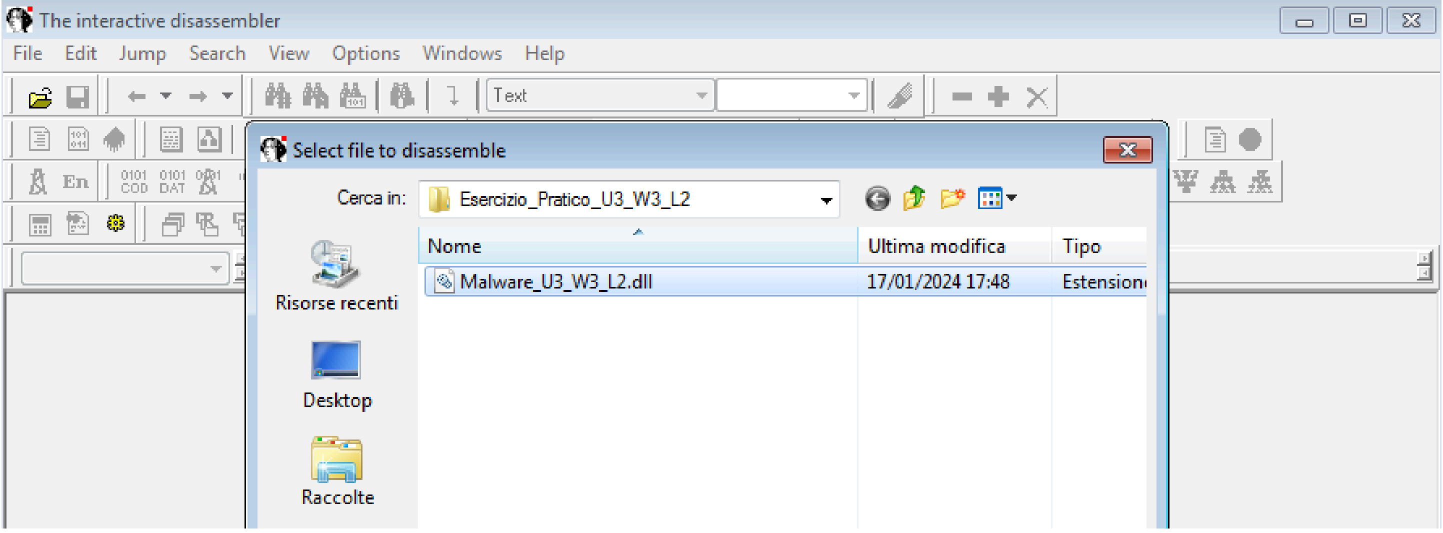
4. How many parameters does the above function have?

5. Provide additional high-level considerations about the malware (behavior).

---

## IDA

**IDA (Interactive DisAssembler)** is a software tool primarily used for analyzing and reverse engineering executable code. It is known for its ability to disassemble program code from machine language into human-readable assembly language, enabling analysts to examine, understand, and modify the internal workings of programs and firmware. In addition to disassembly, **IDA** also supports debugging and advanced static analysis, making it a fundamental tool for cybersecurity experts, developers, and researchers.


The interactive disassembler
File Edit Jump Search View Options Windows Help
Select file to disassemble

Cerca in: Esercizio_Pratico_U3_W3_L2

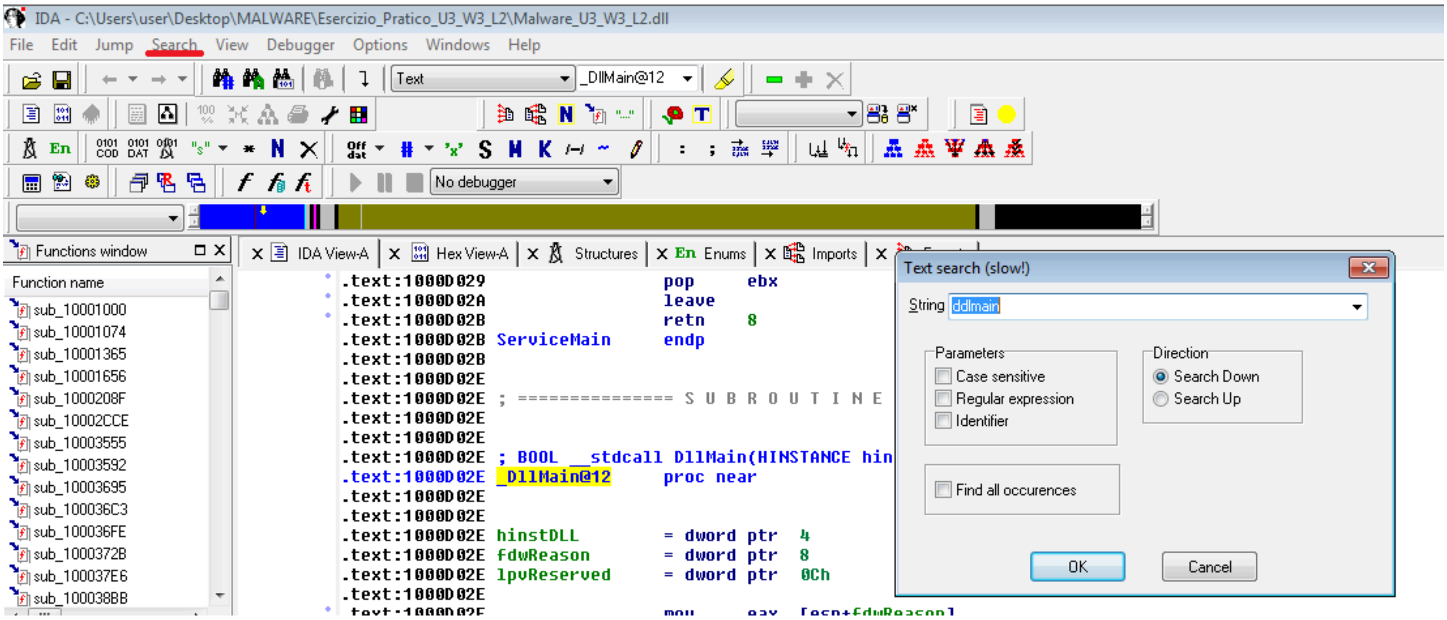
| Nome | Ultima modifica | Tipo |
|---|---|---|
| Malware_U3_W3_L2.dll | 17/01/2024 17:48 | Estension |

Risorse recenti
Desktop
Raccolte

Let's open the IDA tool and load the malware. Click on the folder icon and select the file to analyze.

---

Identify the address of the **DLLMain** function (as it is, in hexadecimal).


IDA - C:\Users\user\Desktop\MALWARE\Esercizio_Pratico_U3_W3_L2\Malware_U3_W3_L2.dll
File Edit Jump Search View Debugger Options Windows Help
Text
_DllMain@12
No debugger
Functions window
IDA View-A
Hex View-A
Structures
Enums
Imports

Function name
sub_10001000
sub_10001074
sub_10001365
sub_10001656
sub_1000208F
sub_10002CCE
sub_10003555
sub_10003592
sub_10003695
sub_100036C3
sub_100036FE
sub_1000372B
sub_100037E6
sub_100038BB

```
.text:1000D029                 pop     ebx
.text:1000D02A                 leave
.text:1000D02B                 retn    8
.text:1000D02B ServiceMain     endp
.text:1000D02B
.text:1000D02E
.text:1000D02E ; =============== S U B R O U T I N E
.text:1000D02E
.text:1000D02E
.text:1000D02E ; BOOL __stdcall DllMain(HINSTANCE hin
.text:1000D02E _DllMain@12     proc near
.text:1000D02E
.text:1000D02E
.text:1000D02E hinstDLL        = dword ptr  4
.text:1000D02E fdwReason       = dword ptr  8
.text:1000D02E lpvReserved     = dword ptr  0Ch
.text:1000D02E
.text:1000D02E                 mov     eax, [esp+fdwReason]
```

Text search (slow!)

String: ddlmain

Parameters
□ Case sensitive
□ Regular expression
□ Identifier

Direction
● Search Down
○ Search Up

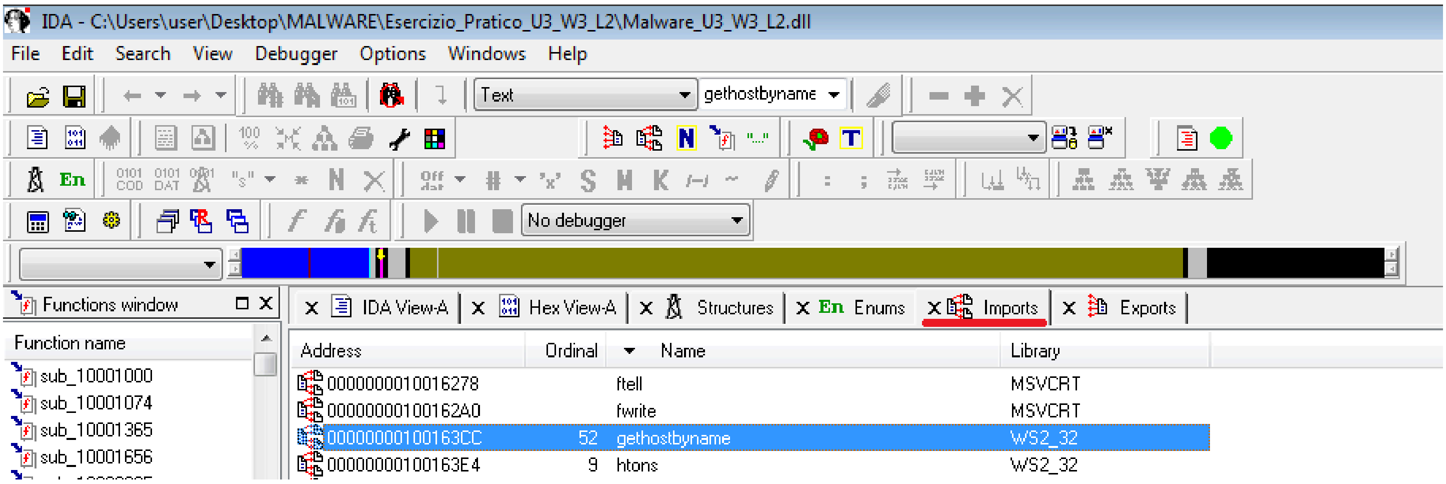
□ Find all occurences

OK Cancel

Using the search function, enter the text to search for, in this case, **DLLMain**, and you can see that it shows the function in the code, allowing us to extract its address, which is **1000D02E**.

---

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



IDA - C:\Users\user\Desktop\MALWARE\Esercizio_Pratico_U3_W3_L2\Malware_U3_W3_L2.dll
File Edit Search View Debugger Options Windows Help
Text
gethostbyname
No debugger
Functions window
IDA View-A
Hex View-A
Structures
Enums
Imports
Exports

```
Function name
sub_10001000
sub_10001074
sub_10001365
sub_10001656
```

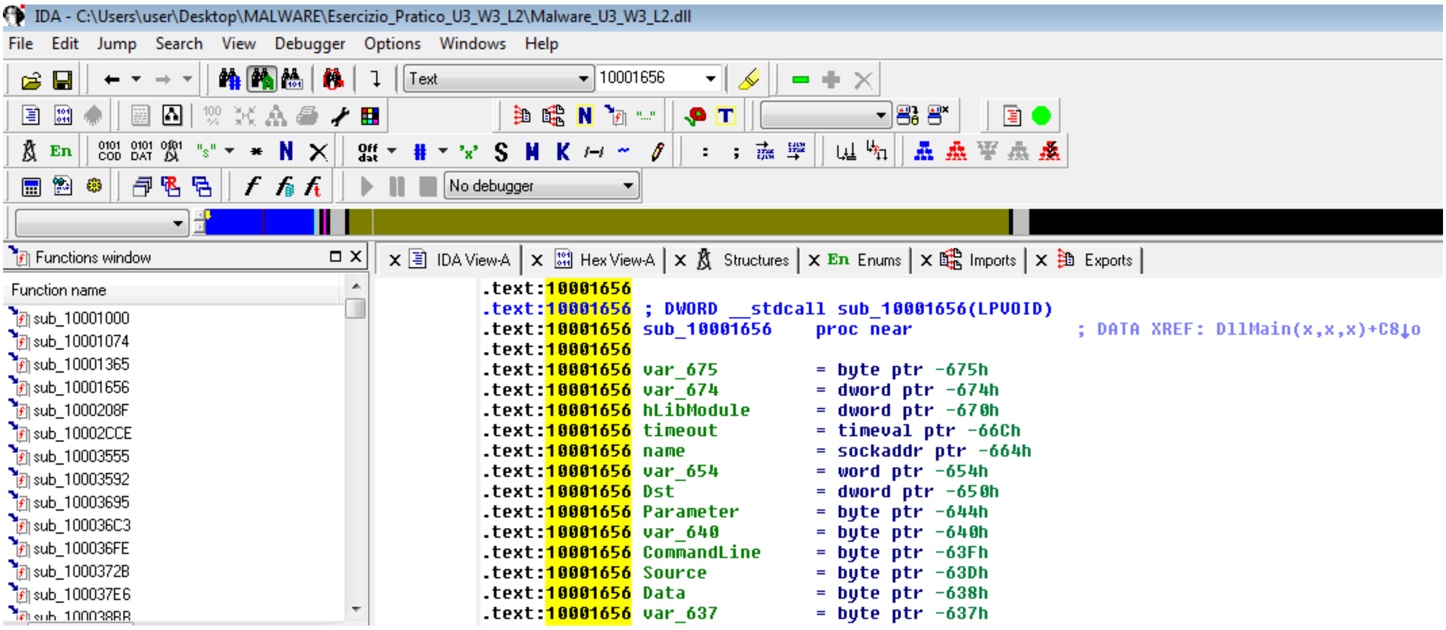
| Address | Ordinal | Name | Library |
|---|---|---|---|
| 0000000010016278 | | ftell | MSVCRT |
| 00000000100162A0 | | fwrite | MSVCRT |
| 00000000100163CC | 52 | gethostbyname | WS2_32 |
| 00000000100163E4 | 9 | htons | WS2_32 |

By selecting the "imports" section and searching for the "**gethostbyname**" function, we can see that it is located at address **100163CC**.
**"gethostbyname"** is a function used in various programming languages to obtain information about a host by its name. That is, it requires the host name and can provide us with various pieces of information, such as its IP address.

---

How many local variables are there in the function at memory location 0x10001656?

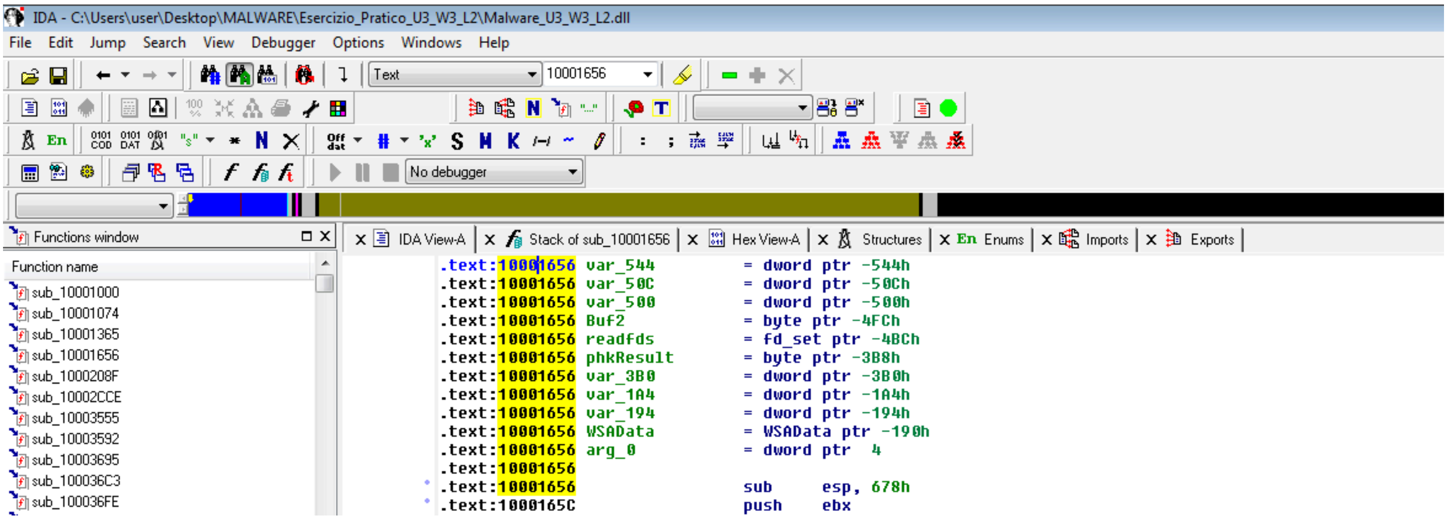
How many parameters does the above function have?



IDA - C:\Users\user\Desktop\MALWARE\Esercizio_Pratico_U3_W3_L2\Malware_U3_W3_L2.dll
File Edit Jump Search View Debugger Options Windows Help
Text
10001656
No debugger
Functions window
IDA View-A
Hex View-A
Structures
Enums
Imports
Exports

```
Function name
sub_10001000
sub_10001074
sub_10001365
sub_10001656
sub_1000208F
sub_10002CCE
sub_10003555
sub_10003592
sub_10003695
sub_100036C3
sub_100036FE
sub_1000372B
sub_100037E6
sub_100038BB
```

```
.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 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
.text:10001656                 sub     esp, 678h
.text:1000165C                 push    ebx
```

From the images, we can observe that at the memory address **10001656**, there are 23 local variables. There is only one parameter, named **arg_0**.