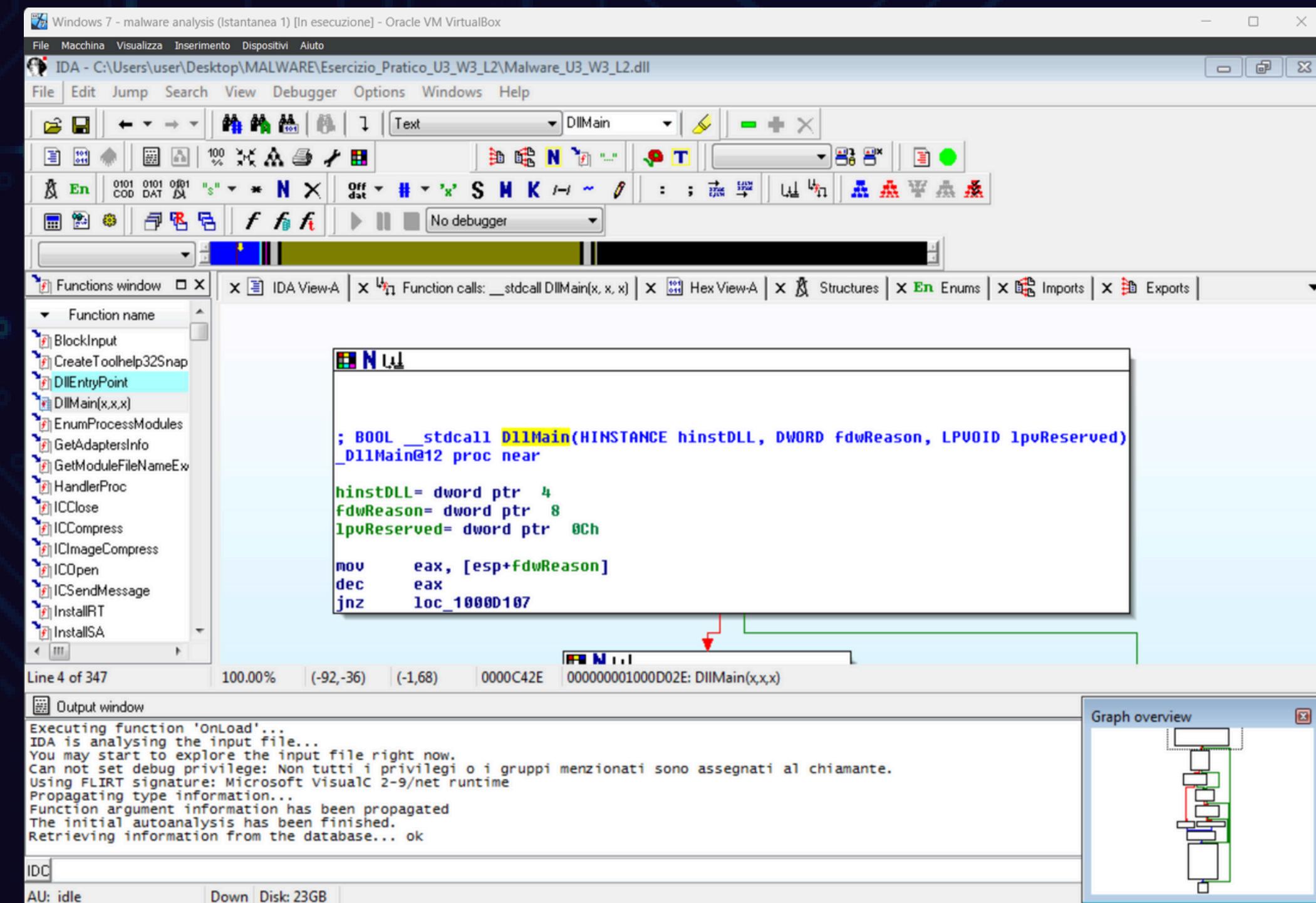


S11-L2

Traccia: Esercizio Analisi statica 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? Cosa fa la funzione?
3. Quante sono le variabili locali della funzione alla locazione di memoria 0x10001656?
Quanti sono, invece, i parametri della funzione sopra?
4. Inserire altre considerazioni macro livello sul malware (comportamento)

All'apertura del programma viene generato un grafico a blocchi per facilitare la comprensione del malware in questione, per trovare l'indirizzo della funzione DllMain ci basterà cliccare sul nome e premere la barra spaziatrice sulla tastiera per passare alla visione del codice.



1- L'indirizzo della funzione DllMain è 1000D02E

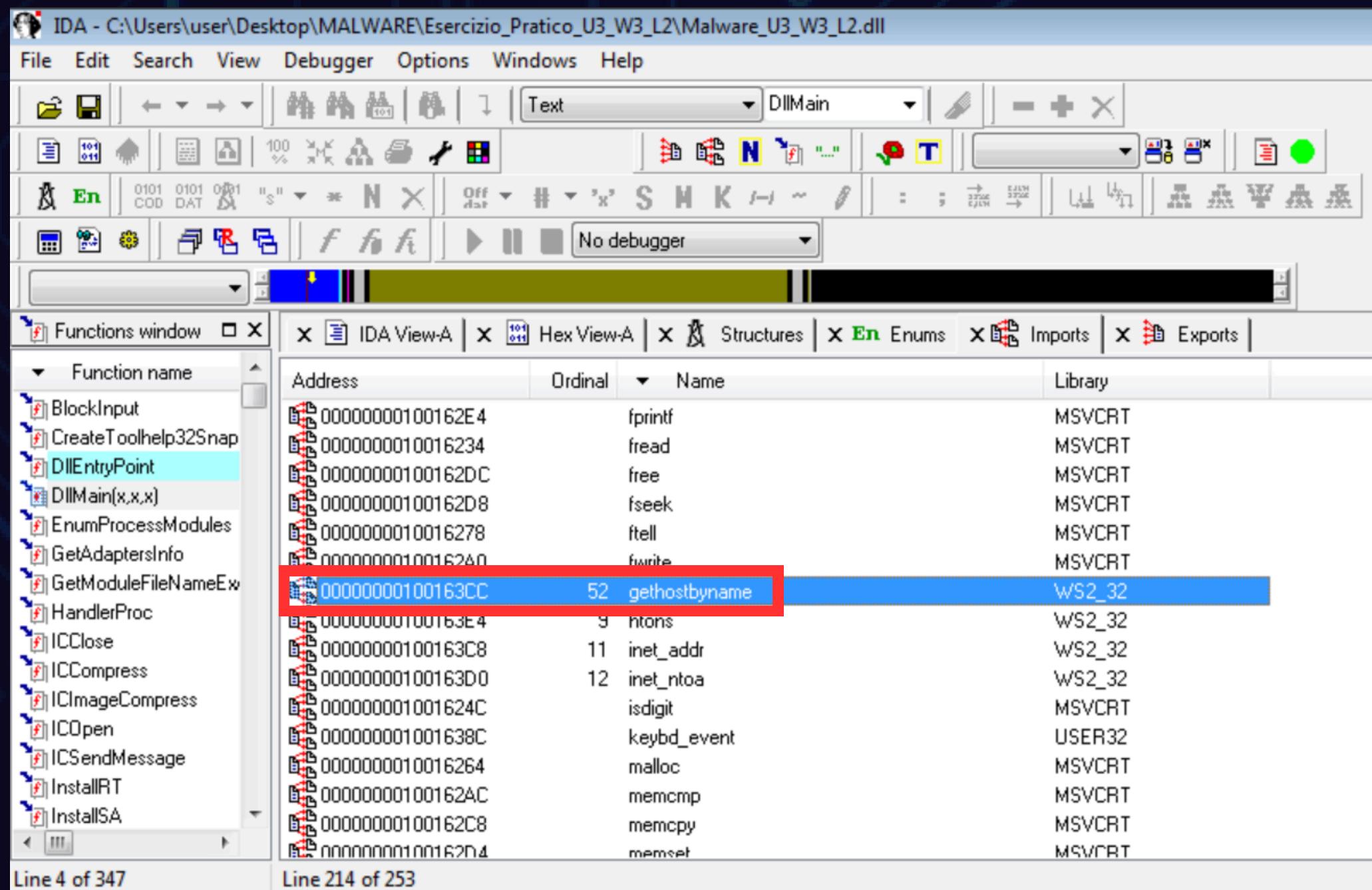
The screenshot shows the IDA Pro interface with the file `Malware_U3_W3_L2.dll` loaded. The assembly dump window displays the `DllMain` function. The instruction at address `1000D02E` is highlighted with a red box. The assembly code for `DllMain` is as follows:

```
.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 hinstDLL, DWORD FdwReason, LPVOID lpvReserved)
.text:1000D02E _DllMain@12    proc near
.text:1000D02E ; CODE XREF: DllEntryPoint+4B↓
.text:1000D02E ; DATA XREF: sub_100110FF+2D↓
.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:1000D032     dec    eax
.text:1000D033     jnz    loc_1000D107
```

The status bar at the bottom shows the address `0000C42E` and the function name `000000001000D02E: DllMain(x,x,x)`.

2- Spondandosi sulla scheda “imports” troviamo la funzione `gethostbyname` che si trova all’indirizzo 00000000100163CC.

La funzione `gethostbyname` è una funzione di rete utilizzata per risolvere un nome di dominio in un indirizzo IP. Viene comunemente utilizzata nelle applicazioni di rete per convertire un nome host leggibile dall'uomo in un indirizzo IP che può essere utilizzato per stabilire una connessione.



3- Possiamo inserire l'indirizzo 10001656 nella casella evidenziata in alto per trovare la funzione immediatamente, verranno evidenziate tutte le righe con le variabili e parametri appartenenti alla funzione.

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

Functions window

Function name

- BlockInput
- CreateToolhelp32Snap
- DllEntryPoint
- DllMain(x,x,x)
- EnumProcessModules
- GetAdaptersInfo
- GetModuleFileNameEx
- HandlerProc
- ICClose
- ICCompress
- ICImageCompress
- ICOOpen
- ICSendMessage
- InstallRT
- InstallSA
- InstallSB
- Module32First
- Module32Next
- PSLIST
- Process32First
- Process32Next
- ServiceMain

text-10001656

```
.text:10001656 ; ===== S U B R O U T I N E =====
.text:10001656 ; DWORD __stdcall sub_10001656(LPVOID)
.text:10001656 sub_10001656 proc near ; DATA XREF: DllMai
.var_675      = byte ptr -675h
.var_674      = dword ptr -674h
.hLibModule   = dword ptr -670h
.timeout     = timeval ptr -66Ch
.name        = sockaddr ptr -664h
.var_654      = word ptr -654h
.Dst          = dword ptr -650h
.Parameter    = byte ptr -644h
.var_640      = byte ptr -640h
.CommandLine  = byte ptr -63Fh
.Source       = byte ptr -63Dh
.Data         = byte ptr -638h
.var_637      = byte ptr -637h
.var_544      = dword ptr -544h
.var_50C      = dword ptr -50Ch
.var_500      = dword ptr -500h
.Buf2         = byte ptr -4FCh
.readfds     = fd_set ptr -4BCh
```

Line 4 of 347

00000A56 0000000010001656: sub_10001656

```
; DWORD __stdcall sub_10001656(LPUOID)
sub_10001656 proc near

var_675= byte ptr -675h
var_674= dword ptr -674h
hLibModule= dword ptr -670h
timeout= timeval ptr -66Ch
name= sockaddr ptr -664h
var_654= word ptr -654h
Dst= dword ptr -650h
Parameter= byte ptr -644h
var_640= byte ptr -640h
CommandLine= byte ptr -63Fh
Source= byte ptr -63Dh
Data= byte ptr -638h
var_637= byte ptr -637h
var_544= dword ptr -544h
var_50C= dword ptr -50Ch
var_500= dword ptr -500h
Buf2= byte ptr -4FCh
readfds= fd_set ptr -4BCh
phkResult= byte ptr -3B8h
var_3B0= dword ptr -3B0h
var_1A4= dword ptr -1A4h
var_194= dword ptr -194h
WSAData= WSADATA ptr -190h
arg_0= dword ptr 4
```

VARIABILI

PARAMETRO

In questo caso abbiamo 23 variabili e 1 parametro.
Le variabili si riconoscono perché hanno offset negativo
rispetto al EBP mentre i parametri offset positivo.

Se un malware utilizza la funzione “gethostbyname”, è probabile che stia cercando di risolvere un nome di dominio in un indirizzo IP. Questa operazione è tipica di malware che hanno bisogno di connettersi a server remoti per vari motivi:

- Il malware potrebbe utilizzare gethostbyname per risolvere il nome di dominio del server di comando e controllo (C&C). Una volta ottenuto l'indirizzo IP, il malware può connettersi al server per ricevere comandi, aggiornare la sua configurazione o inviare dati rubati.
- Se il malware è progettato per condurre attacchi mirati, potrebbe risolvere nomi di dominio di specifici target per lanciare attacchi di phishing, DDoS, o altri tipi di attacco.

In questo caso il malware utilizzava anche altre funzioni per la creazione di sotto-processi o per ricavare informazioni sul sistema, si potrebbe verificarne il comportamento in maniera approfondita con un'analisi dinamica.