Progetto S11 L2

Davide Andreozzi

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.

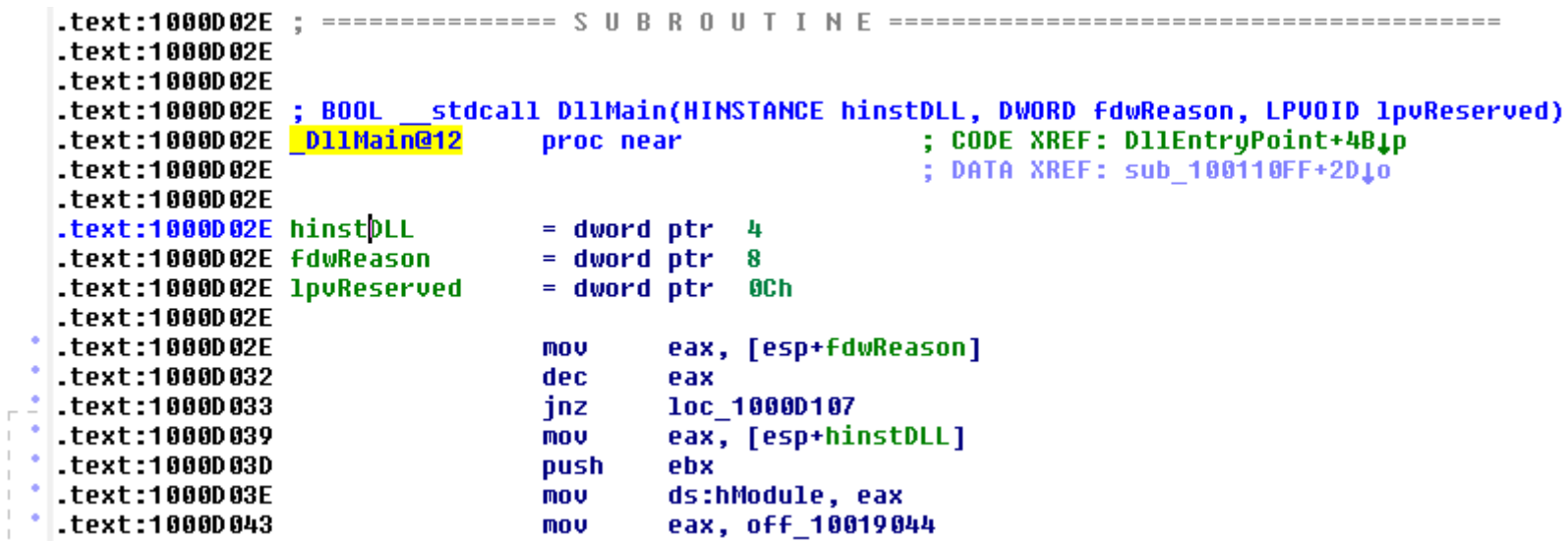
1. Individuare **l'indirizzo** della funzione **DLLMain** (così com'è, in esadecimale)
2. Dalla scheda «imports» individuare la funzione «**gethostbyname** ». Qual è l'indirizzo dell'import? **Cosa fa la funzione?**

3. Quante sono le **variabili locali** della **funzione** alla locazione di memoria 0x10001656?
4. Quanti sono, invece, i **parametri** della funzione sopra?

5. Inserire altre considerazioni macro livello sul malware (comportamento)

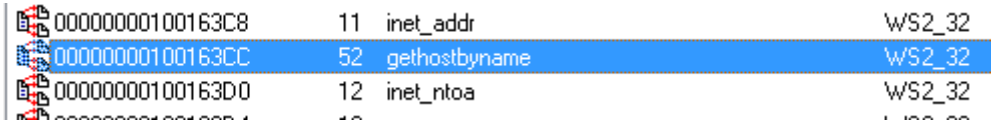
1) Individuare l'indirizzo della funzione DLLMain

Su IDA andiamo a cercare nella sezione **Search** «DLLMain» e possiamo vedere che la funzione è localizzata in **1000D02E**.



2) Individuare gethostbyname – Cosa fa?

- Andando nella sezione import possiamo cercare o trovare la funzione all'indirizzo **100163CC**.
- Questa funzione recupera le informazioni host corrispondenti a un nome host, in sostanza va ad ottenere l'indirizzo IPv4 di un nome di dominio.

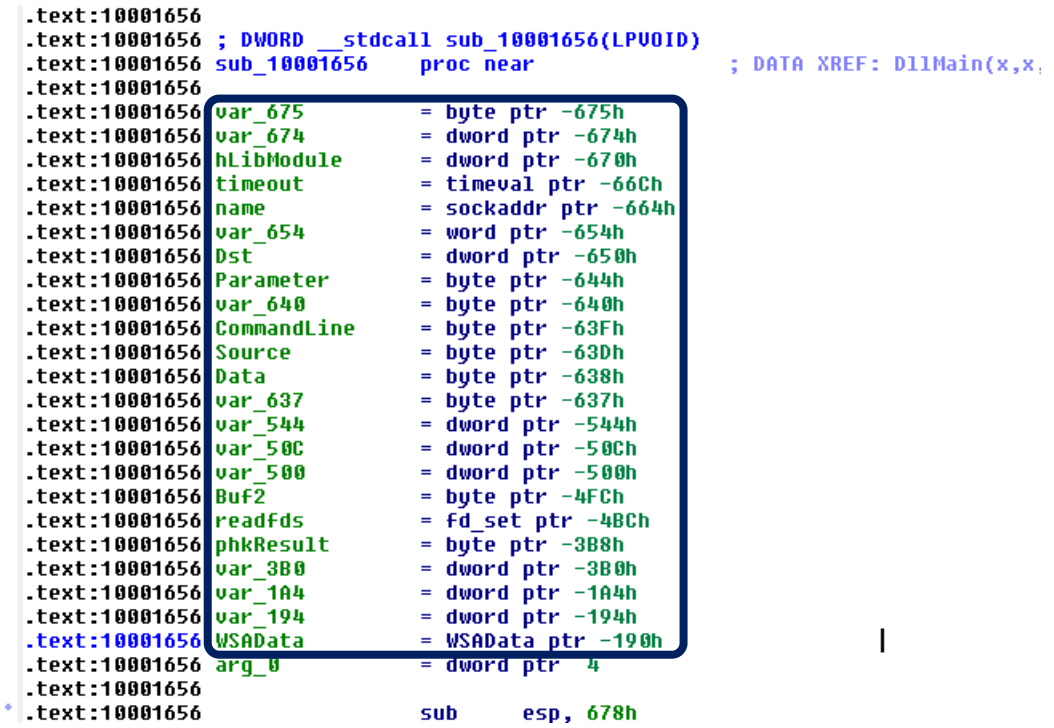



```
.idata:100163CC ; struct hostent *__stdcall gethostbyname(const char *name)
.idata:100163CC                 extrn gethostbyname:dword
.idata:100163CC                                         ; CODE XREF: sub_10001074:loc_100011AF↑p
.idata:100163CC                                         ; sub_10001074+1D3↑p ...
```

3) Quante variabili in 0x10001656

Come sappiamo le variabili sono valori con offset negativo.

Spostandoci all'indirizzo sopracitato possiamo contare 20 Variabili come evidenziato nella figura accanto.

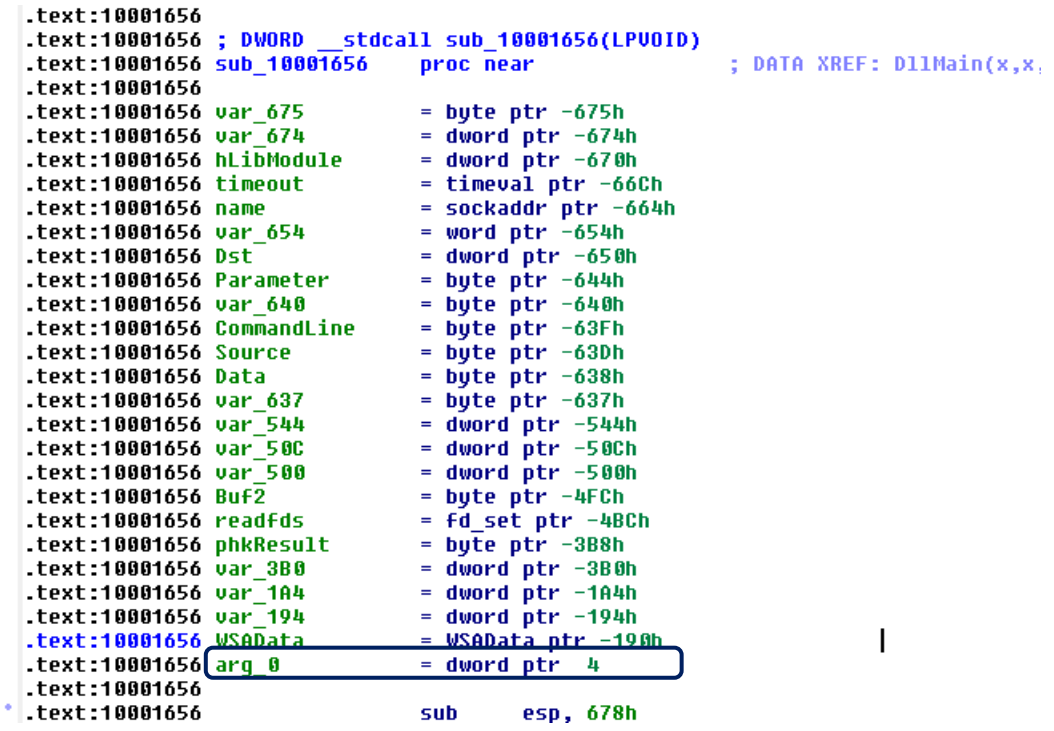


4) Quanti parametri

A differenza delle variabili, i parametri sono valori con offset positivo.

In funzione di quanto detto possiamo notare la presenza di un solo parametro evidenziato in figura accanto.

arg_0



5) Considerazioni sul malware

Questo malware va a importare le seguenti librerie:

GDI32.dll	17	00016BE0	00000000	00000000	000170A2	00016084
PSAPI.DLL	2	00016E88	00000000	00000000	000170DA	0001632C
WS2_32.dll	15	00016F20	00000000	00000000	000170E4	000163C4
iphlpapi.dll	1	00016F60	00000000	00000000	00017102	00016404
KERNEL32.dll	89	00016C28	00000000	00000000	0001773E	000160CC
USER32.dll	26	00016E94	00000000	00000000	00017920	00016338
ADVAPI32.dll	32	00016B5C	00000000	00000000	00017BA6	00016000
ole32.dll	5	00016F68	00000000	00000000	00017C0C	0001640C
OLEAUT32.dll	2	00016E7C	00000000	00000000	00017C16	00016320
MSVFW32.dll	5	00016E64	00000000	00000000	00017C68	00016308
WINMM.dll	7	00016F00	00000000	00000000	00017CEC	000163A4
MSVCRT.dll	52	00016D90	00000000	00000000	00017EC8	00016234

Il malware potrebbe andare a modificare valori di registro leggere e gestire i processi di sistema e installare una backdoor.

Analizzando il codice in Assembly possiamo notare una parte in cui tramite l'utilizzo di cmd.exe si va ad avviare una shell.

Inoltre, in una parte di codice il programma tramite la funzione «sleep» permette al processo di mettersi in pausa temporaneamente.