

ESERCIZIO S10 L2

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)

3

```

1)
ServiceMain .text 10001F30 000000FE H . . . B I .
DllMain(x,x,x) .text 1000D02E 000000DF R . . . T .
sub_1000D10D .text 1000D10D 000000C6 R . . . B T .

.text:1000D02E
.text:1000D02E ; SUBROUTINE
.text:1000D02E
.text:1000D02E
.text:1000D02E ; 800L __stdcall DllMain(HINSTANCE hinstDLL,DWORD fdwReason,LPUVOID lpvReserved)
rol Flow |t: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
.text:1000D02E fdwReason = dword ptr 8
.text:1000D02E lpvReserved = dword ptr 0Ch
.text:1000D02E
.text:1000D02E mov eax, [esp+fdwReason]
.text:1000D032 dec eax

2)
100163CC 52 gethostbyname WS2_32
100163FA 9 gethostbyname WS2_32
idata:100163CC ; struct hostent * __stdcall gethostbyname(const char *name)
idata:100163CC extrn gethostbyname:dword
idata:100163CC ; DATA XREF: sub_10001074:loc_100011AF↑r
idata:100163CC ; sub_10001074+1D3↑r ...
idata:100163CC = char * __stdcall host_ntos(struct in_addr in)

```

gethostbyname, che è una funzione di una libreria di sistema di più alto livello, generalmente utilizzata in linguaggi di programmazione ad alto livello come C o Python per risolvere un nome di dominio in un indirizzo IP.

```
.text:10001651
.text:10001656 ; SUBROUTINE
.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 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
of flow t.txt:10001656 var_544 = dword ptr -544h
.txt:10001656 var_50C = dword ptr -50Ch
.txt:10001656 var_500 = dword ptr -500h
.txt:10001656 var_4FC = dword ptr -4FCh
.txt:10001656 readfds = fd_set ptr -48Ch
.txt:10001656 phkResult = HKEY__ ptr -388h
.txt:10001656 var_3B0 = dword ptr -3B0h
.txt:10001656 var_1A4 = dword ptr -1A4h
.txt:10001656 var_194 = dword ptr -194h
.txt:10001656 WSADATA = WSADATA ptr -190h
.txt:10001656 arg_0 = dword ptr 4
.txt:10001656
.txt:10001656 sub esp, 678h
.txt:10001656
```

```

::10001656 WSAData      = WSAData ptr -190h
::10001656 arg_0         = dword ptr  4
::10001656

```

Dopo aver preso il codice hash del malware usando CFF explorer:

e possiamo usarlo in virus total per avere un primo sguardo

Ed in effetti è una backdoor