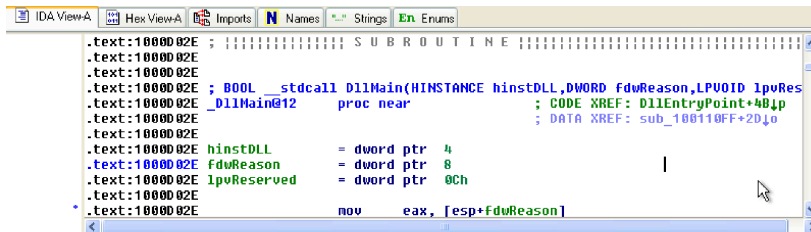


2. Malware analysis: Analisi statica avanzata

- Indirizzo della funzione DLLMain

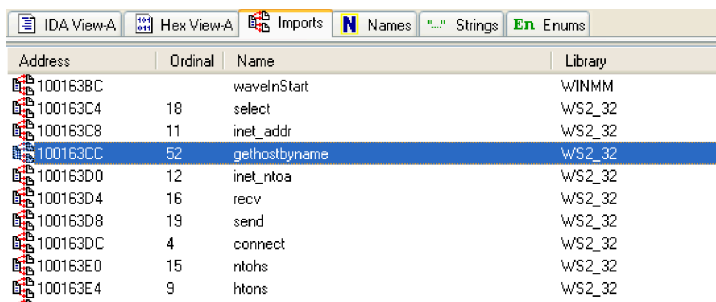
Attraverso IDA andiamo ad analizzare il file eseguibile del malware. Modificando la visualizzazione a modalità testuale possiamo andare a ricercare la funzione DLLMain.



```
.text:1000D02E ; SUBROUTINE
.text:1000D02E
.text:1000D02E
.text:1000D02E ; 0001_stdcall_DllMain(HINSTANCE hinstDLL,DWORD fdwReason,LPUUID lpvRes
.text:1000D02E _DlMain@12 proc near ; CODE XREF: DllEntryPoint+4B1p
.text:1000D02E ; DATA XREF: sub_100110FF+2D10
.text:1000D02E
.text:1000D02E hinstDLL = dword ptr 4
.text:1000D02E fdwReason = dword ptr 8
.text:1000D02E lpvReserved = dword ptr 0Ch
.text:1000D02E
.text:1000D02E nov eax, [esp+fdwReason]
```

L'indirizzo della funzione main è **1000D02E**.

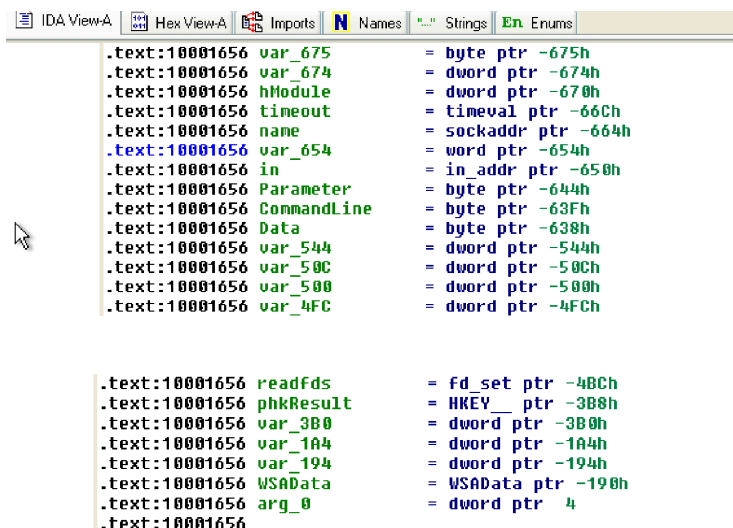
- Indirizzo dell'import 'gethostbyname'.



Address	Ordinal	Name	Library
100163BC		waveInStart	WINMM
100163C4	18	select	WS2_32
100163C8	11	inet_addr	WS2_32
100163CC	52	gethostbyname	WS2_32
100163D0	12	inet_ntoa	WS2_32
100163D4	16	recv	WS2_32
100163D8	19	send	WS2_32
100163DC	4	connect	WS2_32
100163E0	15	ntohs	WS2_32
100163E4	9	htons	WS2_32

Aperto la finestra degli imports andiamo a trovare la funzione richiesta ed il relativo indirizzo: **100163CC**.

- Variabili locali della funzione all'allocazione di memoria 0x10001656.



```
.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
```

Le variabili locali sono **20**, tutte con offset negativo.

- Parametri della funzione sopra.

```
.text:10001656 var_194      = dword ptr -194h
.text:10001656 WSADData  = WSADData ptr -190h
.text:10001656 arg_0     = dword ptr  4
* .text:10001656          sub     esp, 678h
```

Possiamo vedere che contiene un solo parametro: **arg_0**

In questo caso viene passato solo un argomento alla funzione con offset positivo rispetto ad EBP.

- Considerazioni sul comportamento del malware.

```
74 ; char aBackdoorServer[]
74 aBackdoorServer db 0Dh,0Ah ; DATA XREF: sub_100042DB+B5↑to
74 db 0Dh,0Ah
74 db '*****',0Dh,0Ah
74 db '[BackDoor Server Update Setup]',0Dh,0Ah
74 db '*****',0Dh,0Ah
74 db 0Dh,0Ah,0
DB align 4
```

Il malware è una backdoor.

Non viene installato se rileva che la macchina è una virtual machine.

Inserendo l' MD5 del malware su **Virustotal**, possiamo avere un riscontro positivo.

50 / 61

50 security vendors and no sandboxes flagged this file as malicious

eb1079bcd96bc9cc19c38b76342113a09666aad47518ff1a7536eebff8aa
db4a
X-doorc
pedll corrupt armadillo overlay

130.94 KB
Size

2023-02-14 11:11:32 UTC
1 month ago

DLL

DETECTION DETAILS RELATIONS BEHAVIOR COMMUNITY 19

Join the VT Community and enjoy additional community insights and crowdsourced detections, plus an API key to [automate checks](#).

Popular threat label Trojan:Idcaf Threat categories trojan Family labels idcaf

Security vendors' analysis

Security vendors' analysis		Do you want to automate checks?	
AhnLab-V3	Backdoor:Win32.Agent.R9408	Alibaba	Backdoor:Win32/Idcaf.9f3a5556
ALYac	Backdoor:XIW	Antiy-AVL	Trojan[Backdoor]Win32.Agent
Arcabit	Backdoor:XIW	Avast	Win32:Agent-OLH [Trj]
AVG	Win32:Agent-OLH [Trj]	Avira (no cloud)	BDS/Agent.twe.134160
BitDefender	Backdoor:XIW	CiamAV	Win.Trojan.Idcaf-9937585-0