```
.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               ; 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
.text:1000D033                 jnz     loc_1000D107
.text:1000D039                 mov     eax, [esp+hinstDLL]
.text:1000D03D                 push    ebx
.text:1000D03E                 mov     ds:hModule, eax
.text:1000D043                 mov     eax, off_10019044
.text:1000D048                 push    esi
.text:1000D049                 add     eax, 0Dh
.text:1000D04C                 push    edi
.text:1000D04D                 push    eax             ; char *
.text:1000D04E                 call    strlen
.text:1000D053                 mov     ebx, ds:CreateThread
.text:1000D059                 mov     esi, ds:_strnicmp
.text:1000D05F                 xor     edi, edi
.text:1000D061                 pop     ecx
.text:1000D062                 test    eax, eax
.text:1000D064                 jz      short loc_1000D089
.text:1000D066                 mov     eax, off_10019044
.text:1000D06B                 push    7               ; size_t
.text:1000D06D                 add     eax, 0Dh
.text:1000D070                 push    offset aHttp    ; "http:////"
.text:1000D075                 push    eax             ; char *
.text:1000D076                 call    esi ; _strnicmp
.text:1000D078                 add     esp, 0Ch
.text:1000D07B                 test    eax, eax
```

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

```
.idata:100163C8                extrn inet_addr:dword   ; DATA XREF: sub_10001074+11E↑r
.idata:100163C8                                        ; sub_10001074+1BF↑r ...
.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:100163D0 ; char *__stdcall inet_ntoa(struct in_addr in)
.idata:100163D0                extrn inet_ntoa:dword   ; DATA XREF: sub_10001074:loc_10001311↑r
.idata:100163D0                                        ; sub_10001365:loc_10001602↑r ...
.idata:100163D4 ; int __stdcall recv(SOCKET s,char *buf,int len,int flags)
.idata:100163D4                extrn recv:dword        ; DATA XREF: sub_10001656+2D5↑r
.idata:100163D4                                        ; sub_10001656+3F2↑r ...
.idata:100163D8 ; int __stdcall send(SOCKET s,const char *buf,int len,int flags)
.idata:100163D8                extrn send:dword        ; DATA XREF: sub_10001656+290↑r
.idata:100163D8                                        ; sub_10001656+2AB↑r ...
.idata:100163DC ; int __stdcall connect(SOCKET s,const struct sockaddr *name,int namelen)
.idata:100163DC                extrn connect:dword     ; DATA XREF: sub_10001656+251↑r
.idata:100163DC                                        ; sub_1000208F+43C↑r ...
.idata:100163E0 ; u_short __stdcall ntohs(u_short netshort)
.idata:100163E0                extrn ntohs:dword       ; DATA XREF: sub_10001656+214↑r
.idata:100163E0                                        ; sub_10006EE1+52↑r ...
.idata:100163E4 ; u_short __stdcall htons(u_short hostshort)
.idata:100163E4                extrn htons:dword       ; DATA XREF: sub_1000208F+382↑r
.idata:100163E4                                        ; sub_1000208F+3CF↑r ...
.idata:100163E8 ; int __stdcall setsockopt(SOCKET s,int level,int optname,const char *optval,int optlen)
.idata:100163E8                extrn setsockopt:dword  ; DATA XREF: sub_1000208F+42F↑r
.idata:100163E8                                        ; sub_1000208F+A43↑r ...
.idata:100163EC ; int WSACleanup(void)
.idata:100163EC                extrn WSACleanup:dword  ; DATA XREF: sub_1000208F:loc_10002CB4↑r
.idata:100163EC                                        ; sub_10002CCE+823↑r ...
.idata:100163F0 ; int __stdcall WSAStartup(WORD wVersionRequested,LPWSADATA lpWSAData)
.idata:100163F0                extrn WSAStartup:dword  ; DATA XREF: sub_10001656+4E↑r
.idata:100163F0                                        ; sub_1000208F+342↑r ...
.idata:100163F4 ; int __stdcall closesocket(SOCKET s)
.idata:100163F4                extrn closesocket:dword ; DATA XREF: sub_10001656:loc_100016F5↑r
.idata:100163F4                                        ; sub_10001656:loc_100019AB↑r ...
.idata:100163F8 ; SOCKET __stdcall socket(int af,int type,int protocol)
.idata:100163F8                extrn socket:dword      ; DATA XREF: sub_10001656+AB↑r
.idata:100163F8                                        ; sub_1000208F+3F4↑r ...
.idata:100163FC ; int WSAGetLastError(void)
```

L’indirizzo di import della funzione *gethostbyname* è *0x100163CC*. Questa funzione fa parte della libreria *winsock* e viene utilizzata per ottenere le informazioni su un host in base al suo nome.

```
.text:10001649                call    ds:Sleep
.text:1000164F                xor     ebp, ebp
.text:10001651                jmp     loc_100013A5
.text:10001651 sub_10001365   endp
.text:10001651
.text:10001656
.text:10001656 ; ||||||||||||||| S U B R O U T I N E |||||||||||||||||||||||||||||||||||||||
.text:10001656
.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
.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
.text:10001656                sub     esp, 678h
.text:1000165C                push    ebx
.text:1000165D                push    ebp
.text:1000165E                push    esi
.text:1000165F                push    edi
.text:10001660                call    sub_10001000
```

Le variabili locali della funzione alla allocazione di memoria *0x10001656* sono 23. Queste variabili sono identificate da nomi e offset negativi, che rappresentano la distanza tra la base della pila e la variabile.
I nomi delle variabili locali sono i seguenti:

- *name*
- *in*
- *Parameter*
- *CommandLine*
- *Data*
- *readfds*
- *phkResults*
- *var 380*
- *var 184*
- *var 194*
- *WSAData*

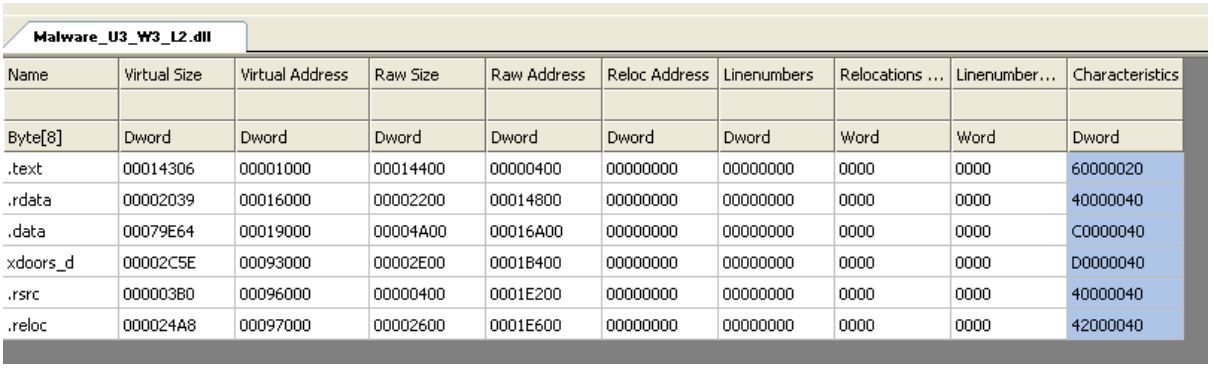
- *arg_0*

Gli offset delle variabili locali sono i seguenti:

- *-654h*
- *-650h*
- *-644h*
- *-63Fh*
- *-638h*
- *-4BCh*
- *-380h*
- *-1A4h*
- *-194h*
- *-190h*
- *4*

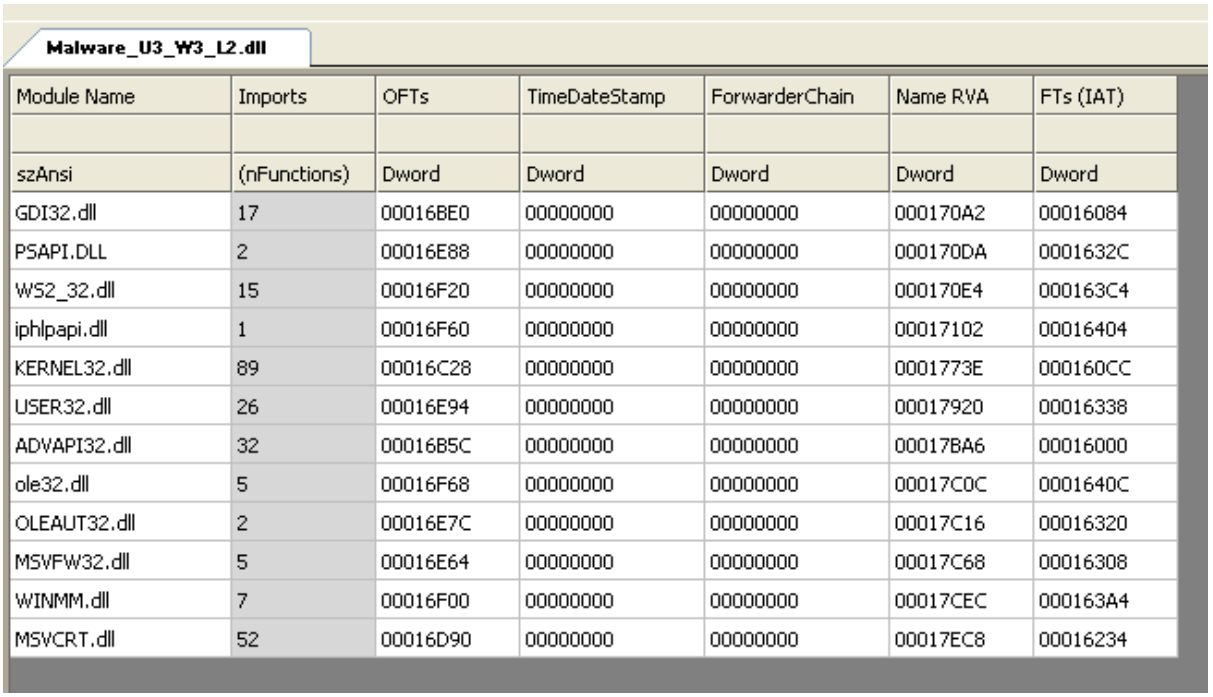
La funzione ha un parametro, che è un puntatore a una struttura *WSAData*. Questo parametro è identificato dall'offset positivo 4.

In base a ciò che vedo attraverso CFF Explorer



Malware_U3_W3_L2.dll

Name	Virtual Size	Virtual Address	Raw Size	Raw Address	Reloc Address	Linenumbers	Relocations ...	Linenumber...	Characteristics
Byte[8]	Dword	Dword	Dword	Dword	Dword	Dword	Word	Word	Dword
.text	00014306	00001000	00014400	00000400	00000000	00000000	0000	0000	60000020
.rdata	00002039	00016000	00002200	00014800	00000000	00000000	0000	0000	40000040
.data	00079E64	00019000	00004A00	00016A00	00000000	00000000	0000	0000	C0000040
xdoors_d	00002C5E	00093000	00002E00	0001B400	00000000	00000000	0000	0000	D0000040
.rsrc	000003B0	00096000	00000400	0001E200	00000000	00000000	0000	0000	40000040
.reloc	000024A8	00097000	00002600	0001E600	00000000	00000000	0000	0000	42000040



Malware_U3_W3_L2.dll

Module Name	Imports	OFTs	TimeDateStamp	ForwarderChain	Name RVA	FTs (IAT)
szAnsi	(nFunctions)	Dword	Dword	Dword	Dword	Dword
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

- può installare un keylogger sul sistema infetto;
- può installare una backdoor sul sistema infetto;
- può reindirizzare delle richieste di pagamento ad un sito web controllato da un utente malintenzionato.

La backdoor inoltre la possiamo notare anche quì

```
xdoors_d:10093D04                db '(3) Move ',27h,'%s',27h,' To ',27h,'%s',27h,' Successfully',0
xdoors_d:10093D29                align 4
xdoors_d:10093D2C ; char a_ubak[]
xdoors_d:10093D2C a_ubak         db '.ubak',0            ; DATA XREF: sub_100042DB+191↑o
xdoors_d:10093D32                align 4
xdoors_d:10093D34 ; char a2GetDllFilenam[]
xdoors_d:10093D34 a2GetDllFilenam db 0Dh,0Ah             ; DATA XREF: sub_100042DB+163↑o
xdoors_d:10093D34                db '(2) Get DLL FileName ',27h,'%s',27h,0
xdoors_d:10093D50 ; char a1EnterCurrentD[]
xdoors_d:10093D50 a1EnterCurrentD db 0Dh,0Ah             ; DATA XREF: sub_100042DB+F2↑o
xdoors_d:10093D50                db '(1) Enter Current Directory ',27h,'%s',27h,0
xdoors_d:10093D73                align 4
xdoors_d:10093D74 ; char aBackdoorServer[]
xdoors_d:10093D74 aBackdoorServer db 0Dh,0Ah             ; DATA XREF: sub_100042DB+B5↑o
xdoors_d:10093D74                db 0Dh,0Ah
xdoors_d:10093D74                db '******************************',0Dh,0Ah
xdoors_d:10093D74                db '[BackDoor Server Update Setup]',0Dh,0Ah
xdoors_d:10093D74                db '******************************',0Dh,0Ah
xdoors_d:10093D74                db 0Dh,0Ah,0
xdoors_d:10093DDB                align 4
xdoors_d:10093DDC ; char aWarn[]
xdoors_d:10093DDC aWarn          db '-warn',0            ; DATA XREF: sub_10004738+198↑o
xdoors_d:10093DE2                align 4
xdoors_d:10093DE4 ; char aErro[]
xdoors_d:10093DE4 aErro          db '-erro',0            ; DATA XREF: sub_10004738+187↑o
xdoors_d:10093DEA                align 4
xdoors_d:10093DEC ; char aStop[]
xdoors_d:10093DEC aStop          db '-stop',0            ; DATA XREF: sub_10004738+176↑o
xdoors_d:10093DF2                align 4
xdoors_d:10093DF4 ; char aShutdown_0[]
xdoors_d:10093DF4 aShutdown_0    db '-shutdown',0        ; DATA XREF: sub_10004738:loc_10004871↑o
xdoors_d:10093DFE                align 10h
xdoors_d:10093E00 ; char Caption[]
xdoors_d:10093E00 Caption        db '-+-ó',0             ; DATA XREF: sub_10004738+107↑o
xdoors_d:10093E00                                        ; sub_10004738+1BB↑o
xdoors_d:10093E05                align 4
xdoors_d:10093E08 ; char aReboot_0[]
xdoors_d:10093E08 aReboot_0      db '-reboot',0          ; DATA XREF: sub_10004738+EB↑o
xdoors_d:10093E10 ; char szDesktop[]
xdoors_d:10093E10 szDesktop      db 'Default',0          ; DATA XREF: sub_10004738+59↑o
```