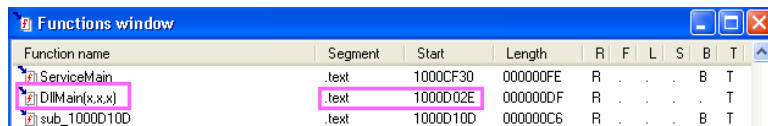


# Report UNIT 3 WEEK 11.2

Malware analysis IDA

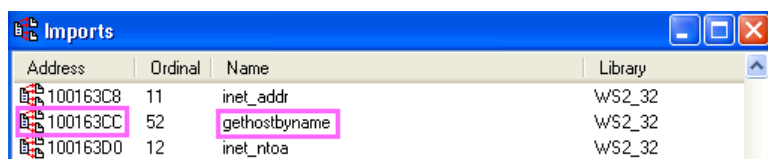
## 1. Individuare l'indirizzo della funzione DLLMAIN (esadecimale).



Function name	Segment	Start	Length	R	F	L	S	B	T
ServiceMain	.text	1000CF30	000000FE	R	.	.	.	B	T
DLLMain(x.x.x)	.text	1000D02E	000000DF	R	.	.	.	.	T
sub_1000D10D	.text	1000D10D	000000C6	R	.	.	.	B	T

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

## 2. Dalla scheda <<imports>> individuare la funzione <<gethostbyname>>. Qual è l'indirizzo dell'import?



Address	Ordinal	Name	Library
100163C8	11	inet_addr	WS2_32
100163CC	52	gethostbyname	WS2_32
100163D0	12	inet_ntoa	WS2_32

L'indirizzo della funzione **gethostbyname** è 100163CC.

## 3. Quante sono le variabili locali della funzione alla locazione di memoria 0x10001656?

## 4. Quanti sono i parametri della funzione sopra?

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

```
var_675= byte ptr -675h
var_674= dword ptr -674h
hModule= dword ptr -670h
timeout= timeval ptr -66Ch
name= sockaddr ptr -664h
var_654= word ptr -654h
in= in_addr ptr -650h
Parameter= byte ptr -644h
CommandLine= byte ptr -63Fh
Data= byte ptr -638h
var_544= dword ptr -544h
var_50C= dword ptr -50Ch
var_500= dword ptr -500h
var_4FC= dword ptr -4FCh
readfds= fd_set ptr -4BCh
phkResult= HKEY__ 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
```

var

par

Le **variabili** sono tutte quelle con offset negativo (20), i **parametri** quelle con offset positivo (1).

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

szAnsi	(nFunctions)
GDI32.dll	17
PSAPI.DLL	2
WS2_32.dll	15
iphlpapi.dll	1
KERNEL32.dll	89
USER32.dll	26
ADVAPI32.dll	32
ole32.dll	5
OLEAUT32.dll	2
MSVFW32.dll	5
WINMM.dll	7
MSVCRT.dll	52

Vengono importate diverse librerie:

### **ADVAPI32**

accesso alle funzionalità avanzate dei servizi di Windows.

### **GDI32**

interazione con i dispositivi di visualizzazione.

### **KERNEL32**

gestione dei processi, dei thread, dei file, della memoria e delle operazioni di I/O.

### **MSVCRT**

funzioni di runtime per i programmi scritti in linguaggio C o C++.

### **MSVFW32**

gestione dei formati video e la riproduzione dei file multimediali.

### **OLEAUT32**

automazione dei componenti software, consentendo la comunicazione tra applicazioni e la manipolazione degli oggetti.

### **PSAPI**

funzioni per ottenere informazioni dettagliate sui processi in esecuzione nel sistema.

### **USER32**

gestione dell'interfaccia utente di Windows.

### **WINMM**

gestione di funzionalità multimediali di base.

### **WS2\_32**

programmazione di applicazioni di rete utilizzando il protocollo TCP/IP.

### **iphlpapi**

accesso alle informazioni di rete.

### **ole32**

gestione degli oggetti COM (Component Object Model) in Windows.

## Considerazioni

```
; BOOL __stdcall DllMain(HINSTANCE hinstDLL,DWORD fdwReason,LPUVOID lpuReserved)
_DllMain@12 proc near

hinstDLL= dword ptr  4
fdwReason= dword ptr  8
lpuReserved= dword ptr 0Ch
```

La **DllMain** esegue una **DLL injection** per inserire una DLL all'interno di un processo in esecuzione. Quando un sistema avvia o termina un processo o un thread, viene chiamata la funzione **DllMain** per ogni DLL caricata, consentendo l'iniezione della DLL nel processo. Questo serve per evitare il rilevamento, assicurando persistenza. La **DllMain** accetta tre parametri:

### **hinstDLL**

handle per il modulo DLL.

### **fdwReason**

variabili che costituiscono l'entry point della funzione.

La DLL verrà chiamata per quattro motivi (Reason): quando si collega una DLL (**DLL\_PROCESS\_ATTACH**), quando si scollega un processo (**DLL\_PROCESS\_DETACH**), quando si allega un thread (**DLL\_THREAD\_ATTACH**) e quando si scollega un thread (**DLL\_THREAD\_DETACH**).

### **lpuReserved**

NULL o non-NULL in base al valore di **fdwReason**.

La funzione **DllMain** restituisce **TRUE** o **FALSE**. Se la funzione **LoadLibrary**, che a sua volta chiama il punto di ingresso della DLL, fallisce (**FALSE**) il sistema richiamerà immediatamente il punto di ingresso, questa volta con il codice **DLL\_PROCESS\_DETACH**. Successivamente, la DLL viene scaricata.

Nella sezione **xdoors\_d** possiamo vedere il funzionamento della backdoor.

```
; HMODULE hModule
hModule dd 0

; HANDLE hThread
hThread dd 0

; HANDLE dword_10093008
dword_10093008 dd 0

align 10h
; char byte_10093010[]
byte_10093010 db 280h dup(0)
dword_10093210 dd 28h
```

### **HMODULE hModule**

chiamata la funzione **GetModuleFileNameA** per ottenere il percorso del file eseguibile o di un modulo caricato in memoria.

### **HANDLE hThread** e **HANDLE dword\_10093008**

chiamano le funzioni **CreateThread** e **SuspendThread** da **DllMain** per creare un nuovo thread di esecuzione nel processo corrente.

### **char byte**

chiamata **RegOpenKeyA** per aprire una chiave di registro, **RegEnumKeyA** per enumerare le chiavi di registro figlie di una chiave di registro e **\_ftol**, che converte un float/double in int/long.

```

; HIC hic
hic dd 0

dword_10093248 dd 0

; HGLOBAL hMem
hMem dd 0

dword_10093230 dd 0
dword_10093250 dd 0

dword_10093234 dd 0
byte_10093254 db 0

; size_t dword_10093238 align 4
dword_10093238 dd 0
; HIC dword_10093258
dword_10093258 dd 0

dword_1009323C dd 0
dword_1009325C dd 0

```

### HGLOBAL hMem e size\_t\_dword\_10093238

chiamano **GlobalUnlock** per sbloccare un blocco di memoria globale, **GlobalFree** per liberare un blocco di memoria globale precedentemente allocato e **??@YAPAXI@Z**, una firma di funzione in C++ per creare spazio di memoria per un oggetto nel programma.

### Hic hic

chiamata **ICCompress** e **ICImageCompress** per la compressione di immagini, **Sleep** per mettere in pausa l'esecuzione del programma (millisecondi), **SystemParametersInfoA** per ottenere/impostare informazioni di configurazione e **SendMessageA** per inviare un messaggio ad una finestra specificata.

```

; char aRundll64_exe[]
aRundll64_exe db 'rundll64.exe',0
align 4
; char aRundll32_exe[]
aRundll32_exe db 'rundll32.exe',0
align 4
; char CmdLine[]
CmdLine db 'ipconfig /flushdns',0

```

### char aRundll64\_exe

chiamata **\_stricmp** per identificare rundll64.exe.

### Char aRundll32\_exe

chiamata **GetCurrentProcessId** per ottenerne l'ID.

### Char CmdLine

chiamata **WinExec** per eseguire ipconfig/flushdns da cmd.

In seguito, vengono caricate varie funzioni da librerie in runtime:

**xkey.dll** (Plug\_KeyLog\_Main), **xproxy.dll** (Plug\_Proxy\_Main), **xflood.dll** (Plug\_Flood\_Main), **xacq.dll** (Plug\_Acq\_Main), **xdev.dll** (Plug\_Dev\_Main), **xsys.dll** (Plug\_Sys\_Main).

Si procede poi all'escalation dei privilegi con iniezione di codice in processi, iniezione di librerie a collegamento dinamico, hijacking di thread, modifica delle autorizzazioni dei file e delle directory, manipolazione di token di accesso, modifica del registro, modifica dell'attributo dei file e rilevamento di ambienti virtualizzati.

```

; char aFgets[]
aFgets db 'fgets',0
align 4

; char aMessage[]
aMessage db 'message',0

; char aStartxservices[]
aStartxservices db 'startxservices',0
align 10h

; char aStartxprocess[]
aStartxprocess db 'startxprocess',0
align 10h

; char aXsys_dll[]
aXsys_dll db 'xsys.dll',0
align 10h

aPlug_sys db 'plug_sys',0
align 4

aUpdate db 'update',0
align 4

aLogoff db 'logoff',0
align 4

aShutdown db 'shutdown',0
align 4

aReboot db 'reboot',0

```

Segue la scoperta della configurazione di rete, la scoperta dei processi, l'interrogazione delle informazioni di sistema, la scoperta di file e directory, il rilevamento di software in esecuzione e la cattura di schermate.

```

; char aStartxsound[]
aStartxsound db 'startxsound',0

; char aStartxvideo[]
aStartxvideo db 'startxvideo',0

; char aStartxreg[]
aStartxreg db 'startxreg',0
align 4

; char aStartxfile[]
aStartxfile db 'startxfile',0
align 10h

; char aStartxscreen[]
aStartxscreen db 'startxscreen',0
align 10h

; char aStartxcmd[]
aStartxcmd db 'startxcmd',0

; char aIeuser[]
aIeuser@ db 'IEuser@',0

; char aAnonymous[]
aAnonymous db 'anonymous',0
align 4

; char aFtp_1[]
aFtp_1 db 'FTP://',0
align 4

; char aFtp_0[]
aFtp_0 db 'ftp://',0

```

```

; char aConnectionKeep[]
aConnectionKeep db 'Connection: Keep-Alive',0
align 10h
; DATA XREF: sub_10002CCE+409f0

; char aUserAgentMozil[]
aUserAgentMozil db 'User-Agent: Mozilla/4.0 (compatible; MSIE 6.00; Windows NT 5.1)',0
; DATA XREF: sub_10002CCE+3EBf0

; char aAcceptImageGif[]
aAcceptImageGif db 'Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, appl'
; DATA XREF: sub_10002CCE+3CDf0
db 'ication/x-shockwave-flash, application/vnd.ms-excel, application/'
db 'vnd.ms-powerpoint, application/msword, */*',0

```

```

aAdjusttokenpri db 'AdjustTokenPrivileges',0
align 10h
; sub_10002CCE+409f0

aLookupprivileg db 'LookupPrivilegeValue',0
align 4
; sub_10002CCE+409f0

; char Name[]
Name db 'SeDebugPrivilege',0
; DATA XREF: sub_10002CCE+409f0

aOpenprocesstok db 'OpenProcessToken',0
; DATA XREF: sub_10005778+

; char a0_0_0_0[]
a0_0_0_0 db '0.0.0.0',0
; DATA XREF: sub_10002CCE+409f0

; char aMicrosoftTvVid[]
aMicrosoftTvVid db 'Microsoft TV/Video Connection',0
; DATA XREF: sub_10002CCE+409f0

; char aUmwareVirtualE[]
aUmwareVirtualE db 'UMware Virtual Ethernet Adapter',0

```

```

; *****',0Dh,0Ah
; [BackDoor Server Update Setup]',0Dh,0Ah
; *****',0Dh,0Ah

```

Considerando i comportamenti descritti, la DLL sembra avere caratteristiche di un Trojan.