Title:

Win Vista DLL Injection (32bit)

Date:

January 25st 2009

Website:

http://www.astalavista.com

Author:

Nicolaou George

Mail:

ishtus<\at>astalavista<\d0t>com

Table of Contents

Introduction

An insight on how to inject a dynamic library (DLL) into a 32 bit process in Windows Vista with the use of Remote Threads and taking into consideration the Address Space Layout Randomization (ASLR). The sample code used is written in assembly language (MASM32) using the WinAsm IDE. It should give you a better understanding on how dynamic libraries can be injected.

Tools

The tools used in this paper are the following:

- WinAsm Studio [http://www.winasm.net/]

Code

injectDLL.asm

```
.486
.model   flat, stdcall
option   casemap :none

include          injectDLL.inc

.code
start:
         invoke   GetModuleHandle, NULL
         mov      hInstance, eax
         invoke   DialogBoxParam, hInstance, 101, 0, ADDR DlgProc, 0
         invoke   ExitProcess, eax

DlgProc  proc hWin        :DWORD,
                  uMsg    :DWORD,
                  wParam  :DWORD,
                  lParam  :DWORD

         .if      uMsg == WM_COMMAND
                  .if      wParam == INJECT
                           invoke GetDlgItemText,hWin,PIDTXT,addr hProcIdb,5; Get PID from txtbox
                           invoke GetDlgItemText,hWin,DLLPATH,addr lib,512          ; Get dll pathname from txtbox
                           invoke InjectDll
                  .elseif  wParam == EXIT
                           invoke EndDialog,hWin,0
;////////////////////////////////////Open File Dialog//////////////////////////////////////////////////////////////
                  .elseif wParam == SELECT
                           mov ofn.lStructSize,SIZEOF ofn
                           mov ofn.lpstrFilter,offset strFilter
                           mov ofn.lpstrFile,offset lib
                           mov ofn.nMaxFile,512
                           mov
ofn.Flags,OFN_FILEMUSTEXIST+OFN_PATHMUSTEXIST+OFN_LONGNAMES+OFN_EXPLORER+OFN_HIDEREADONLY
                           invoke GetOpenFileName,addr ofn
                           .if eax==TRUE
                                    invoke SetDlgItemText,hWin,DLLPATH,addr lib
```

```
                                invoke RtlZeroMemory,addr lib,512
                        .endif
;////////////////////////////////////////////////////////////////////////////////////////////////
                .endif
        .elseif uMsg == WM_CLOSE
                invoke  EndDialog,hWin,0
        .endif
        xor     eax,eax
        ret
DlgProc endp

InjectDll PROC
        local hThread:HANDLE
        local pfnRtn:HANDLE
        local pszLibFileRemote:PCSTR
        local ipbase:HANDLE

        invoke lstrlen,addr hProcIdb
        .if eax == 0                                    ; If no PID inserted then use current process PID
                invoke GetCurrentProcessId
                .if eax == NULL
                        invoke MessageBoxA,0,addr error0,addr error0,0
                        Ret
                .endif
                mov hProcId,eax
        .else
                invoke a2dwc,addr hProcIdb      ;Convert String to DWORD
                mov hProcId,eax
        .endif

        invoke
OpenProcess,PROCESS_QUERY_INFORMATION+PROCESS_CREATE_THREAD+PROCESS_VM_OPERATION+PROCESS_VM_
WRITE,0,hProcId ;Open process from PID
        .if eax == NULL
                invoke Error2Txt
                Ret
        .endif
        mov hProcId,eax


;////Calculate bytes needed to allocate for library pathname//////////////
        invoke lstrlen,addr lib
        inc eax
        mov libstrlen,eax
;////////////////////////////////////////////////////////////////////

        invoke VirtualAllocEx,hProcId,0,libstrlen,MEM_COMMIT,PAGE_READWRITE     ;Allocate memory to write dll pathname
        .if eax == NULL
                invoke Error2Txt
                Ret
        .endif
        mov ipbase,eax
```

```
        invoke WriteProcessMemory,hProcId,ipbase,addr lib,libstrlen,0 ; Write dll pathname to allocated memory
        .if eax == NULL
                invoke Error2Txt
                Ret
        .endif

        invoke GetModuleHandle,addr krn32           ;Get address of kernel32 in memory (ASLR friendly)
        mov pfnRtn,eax

        invoke GetProcAddress, pfnRtn,addr llib                 ;Get address if LoadLibraryA
        .if eax == NULL
                invoke Error2Txt
                Ret
        .endif
        mov pfnRtn,eax

        invoke CreateRemoteThread,hProcId,NULL,0,pfnRtn,ipbase,0,NULL     ;Create remote thread and load our dll using
LoadLibraryA
        .if eax == NULL
                invoke Error2Txt
                Ret
        .endif

        Ret
InjectDll EndP
;a quick fix of masm32.lib a2dw proc
a2dwc proc uses ecx edi edx esi String:DWORD

    ;-----------------------------------------
    ; Convert decimal string into dword value
    ; return value in eax
    ;-----------------------------------------

    xor ecx, ecx
    mov edi, String
    invoke lstrlen, String
            xor ecx,ecx
    .while eax != 0
     xor edx, edx
     mov dl, byte ptr [edi]
     sub dl, "0" ; subtrack each digit with "0" to convert it to hex value
     mov esi, eax
     dec esi
     push eax
     mov eax, edx
     push ebx
     mov ebx, 10
      .while esi > 0
       mul ebx
       dec esi
      .endw
```

```
        pop ebx
        add ecx, eax
        pop eax
        inc edi
        dec eax
      .endw

        mov eax, ecx
        ret

a2dwc endp

Error2Txt proc

            invoke GetLastError
            invoke FormatMessage,FORMAT_MESSAGE_FROM_SYSTEM+FORMAT_MESSAGE_IGNORE_INSERTS,NULL,eax,0,addr
errormsg,512,NULL           ;Convert error code to text
            invoke MessageBoxA,0,addr errormsg,addr error0,MB_ICONERROR
            Ret
Error2Txt EndP


end start
```

InjectDLL.inc

```
include             windows.inc
include             user32.inc
include   kernel32.inc
include             comdlg32.inc
;-------------------------------------------------------------------------------------------------------------------
includelib          user32.lib
includelib          kernel32.lib
includelib          comdlg32.lib
;-------------------------------------------------------------------------------------------------------------------
DlgProc             PROTO  :DWORD,:DWORD,:DWORD,:DWORD
PrintReg PROTO  :DWORD
a2dwc    PROTO  :DWORD
Error2Txt           PROTO
InjectDll  PROTO
;-------------------------------------------------------------------------------------------------------------------
INJECT              equ       1001
EXIT                equ       1002
PIDTXT              equ       1003
DLLPATH             equ       1009
SELECT              equ       1012
;-------------------------------------------------------------------------------------------------------------------
.data
            krn32                       db              "kernel32",0
            llib                        db              "LoadLibraryA",0
            error0                      db              "Error",0

            ; Open file dialog
```


```
        strFilter  db               "Dynamic Libraries (*.dll)",0,"*.dll",0,"All Files",0,"*.*",0,0

.data?
        hInstance       dd          ?
        hProcId         HANDLE      ?
        libstrlen       dd          ?
        hProcIdb        db          5 dup(?)
        lib             db          512 dup(?)
        errormsg        db          512 dup(?)
        ;Open file dialog
        ofn             OPENFILENAME <>
;-------------------------------------------------------------------------------------------------------------------------
```

injectDLL.inc

```
#define EXIT 1002
#define PIDLABEL 1008
#define INJECT 1001
#define PIDTXT 1003
#define DLLLABEL 1011
#define SELECT 1012
#define DLLPATH 1009

101 DIALOGEX 0,0,159,66
CAPTION "DLL Injector"
FONT 8,"Tahoma"
STYLE 0x80c80880
EXSTYLE 0x00000000
BEGIN
        CONTROL "Inject",INJECT,"Button",0x10000001,17,44,50,14,0x00000000
        CONTROL "Exit",EXIT,"Button",0x10000000,93,44,50,14,0x00000000
        CONTROL "",PIDTXT,"Edit",0x10000080,28,7,101,12,0x00000200
        CONTROL "PID",PIDLABEL,"Static",0x50000000,11,9,14,9,0x00000000
        CONTROL "",DLLPATH,"Edit",0x50010080,27,22,101,12,0x00000200
        CONTROL "DLL",DLLLABEL,"Static",0x50000000,10,24,14,9,0x00000000
        CONTROL "...",SELECT,"Button",0x50010000,133,23,21,13,0x00000000
END
```

Why Remote Thread?

The idea behind using a remote thread to inject a dynamic library is to create a new thread in a remote process that calls the **LoadLibrary** API and load our DLL inside the address space of that remote thread. The problem with directly parsing the **LoadLibrary** offset to **CreateRemoteThread** is that it resolves to the address in your process import table which unfortunately is not the same as the remote process import table. To overcome this problem we need to find the offset of LoadLibrary inside the address space layout of our process.

ASLR and LoadLibrary

Since at each reboot (or two) the address of **kernel32.dll** (which contains the **LoadLibrary** procedure) might change we use **GetModuleHandle** to retrieve the address of **LoadLibraryA** which will be the same in the remote thread address space.

How to parse an argument to LoadLibrary

The DLL's pathname cannot be addressed to since it does not reside within the remote process address space. We therefore have to call **VirtualAllocEx** to allocate memory in the remote process and therefore patch the pathname of the DLL we intent to inject. We can do that by using **WriteProcessMemory** API.

Finally

When everything is done we can call the CreateRemoteThread and parse the arguments for injecting the DLL (see code). The DLL main function will receive the **DLL_PROCESS_ATTACH** notification and will start executing. The rest are up to you enjoy. :)