x64 Workshop

Didier Stevens

Go to http://workshop-x64.DidierStevens.com
Unzip x64-workshop.zip to c:\workshop
Install:

- 010EditorWin32Installer402.exe
- nasm-2.10.05-installer.exe
  - SysinternalsSuite.zip
- tdm64-gcc-4.7.1-2.exe
- tdm-gcc-4.7.1-2.exe
Exercise 1:

The litmus test
Start the following programs:

- exercise-01-32.exe
- exercise-01-64.exe
Not OK

The version of this file is not compatible with the version of Windows you're running. Check your computer's system information to see whether you need an x86 (32-bit) or x64 (64-bit) version of the program, and then contact the software publisher.
Take a look with Process Explorer

Take a look with 010 Editor
Exercise 2:

A C program
32 gcc: gcc -o exercise-02-32.exe exercise-02.c
64 gcc: gcc -o exercise-02-64.exe exercise-02.c
exercise-02-32.exe

```assembly
public _main
proc near
    ; CODE XREF: ___mingw_CRTStartup+F8↑p

var_20 = dword ptr -20h
var_1C = dword ptr -1Ch
var_4  = dword ptr -4

push    ebp
mov     ebp, esp
and     esp, 0FFFFFFF0h
sub     esp, 20h
call    __main
mov     [esp+20h+var_4], offset aHelloWorld ; "Hello World\n"
inc     [esp+20h+var_4]
mov     eax, [esp+20h+var_4]
mov     [esp+20h+var_20], eax
call    printf
mov     [esp+20h+var_1C], 4
mov     [esp+20h+var_20], offset aSizeofSzhello ; "sizeof(sz

call    printf
leave
ret
endp
```

```assembly
public main
main proc near ; CODE XREF: __tmainCRTStartup+250↑p
                     ; DATA XREF: .pdata:000000000040A048↓o

var_8     = qword ptr -8
arg_0     = dword ptr 10h
arg_8     = qword ptr 18h

push      rbp
mov       rbp, rsp
sub       rsp, 30h
mov       [rbp+arg_0], ecx
mov       [rbp+arg_8], rdx
call      __main
leaq      rax, aHelloWorld ; "Hello World\n"
mov       [rbp+var_8], rax
add       [rbp+var_8], 1
mov       rax, [rbp+var_8]
mov       rcx, rax ; char *
call      printf
mov       edx, 8
leaq      rcx, aSizeOfSzHelloWorld ; "sizeof(szHelloWorld) = %d\n"
call      printf
add       rsp, 30h
pop       rbp
ret
main      endp
```
Exercise 3:

A C dll
32 gcc: gcc -shared -o exercise-03-32.dll exercise-03.c
64 gcc: gcc -shared -o exercise-03-64.dll exercise-03.c
Exercise 4:

Loading and injecting a dll
32 gcc: gcc -o exercise-04-32.exe exercise-04.c
64 gcc: gcc -o exercise-04-64.exe exercise-04.c
exercise-04-32.exe exercise-03-32.dll
exercise-04-64.exe exercise-03-64.dll
exercise-04-32.exe exercise-03-64.dll
exercise-04-64.exe exercise-03-32.dll
// MessageId: ERROR_BAD_EXE_FORMAT
//
// MessageText:
//
// %1 is not a valid Win32 application.
//
#define ERROR_BAD_EXE_FORMAT 193L
Calc.exe, our favorite test dummy

Start calculator 64-bit and 32-bit:

c:\windows\system32\calc.exe

c:\windows\syswow64\calc.exe
inject-dll-32.exe 4352 exercise-03-32.dll
inject-dll-64.exe 2624 exercise-03-64.dll
inject-dll-32.exe 1472 exercise-03-64.dll*
inject-dll-64.exe 1532 exercise-03-32.dll

* inspect memory
inject-dll-32.exe 4352 exercise-03-32.dll
inject-dll-64.exe 2624 exercise-03-64.dll
inject-dll-32.exe 1472 exercise-03-64.dll
inject-dll-64.exe 1532 exercise-03-32.dll 76A44BC6
hProcess = OpenProcess(PROCESS_ALL_ACCESS, FALSE, _stoi(argv[1]));
if (NULL == hProcess)
{
    printf("OpenProcess error: %d\n", GetLastError());
    return -2;
}

if (argc == 3)
    fpLoadLibraryA = GetProcAddress(GetModuleHandle("kernel32.dll"), "LoadLibraryA");
else
{
    char *endPtr;
    fpLoadLibraryA = (FARPROC) strtol(argv[3], &endPtr, 16);
}
printf("fpLoadLibraryA = %p\n", fpLoadLibraryA);

lpArgument = VirtualAllocEx(hProcess, NULL, strlen(argv[2]) + 1, MEM_COMMIT | MEM_RESERVE, PAGE_READWRITE);
if (NULL == lpArgument)
{
    printf("VirtualAllocEx error: %d\n", GetLastError());
    return -3;
}
printf("lpArgument = %p\n", lpArgument);
if (!WriteProcessMemory(hProcess, lpArgument, argv[2], strlen(argv[2]) + 1, &stWritten))
{
    printf("WriteProcessMemory error: %d\n", GetLastError());
    return -4;
}

hThread = CreateRemoteThread(hProcess, NULL, 0, (LPTHREAD_START_ROUTINE)fpLoadLibraryA, lpArgument, 0, &dwThreadID);
if (NULL == hThread)
{
    printf("CreateRemoteThread error: %d\n", GetLastError());
    return -5;
}

CloseHandle(hProcess);
Exercise 5:

Shellcode
nasm -o exercise-05-32.bin exercise-05-32.asm
nasm -o exercise-05-64.bin exercise-05-64.asm
inject-shellcode-32.exe 1532 exercise-05-32.bin
inject-shellcode-64.exe 1472 exercise-05-64.bin
inject-shellcode-32.exe 3396 exercise-05-64.bin
inject-shellcode-64.exe 4188 exercise-05-32.bin
inject-shellcode-32.exe 1532 exercise-05-32.bin
inject-shellcode-64.exe 1472 exercise-05-64.bin
inject-shellcode-32.exe 3396 exercise-05-64.bin
inject-shellcode-64.exe 4188 exercise-05-32.bin
hProcess = OpenProcess(PROCESS_ALL_ACCESS, FALSE, _tstoi(argv[1]));
if (NULL == hProcess)
{
    printf("OpenProcess error: \d\n", GetLastError());
    return -3;
}

lpArgument = VirtualAllocEx(hProcess, NULL, sizeof(abBuffer), MEM_COMMIT | MEM_RESERVE, PAGE_EXECUTE_READWRITE);
if (NULL == lpArgument)
{
    printf("VirtualAllocEx error: \d\n", GetLastError());
    return -4;
}

printf("lpArgument = %p\n", lpArgument);
if (!WriteProcessMemory(hProcess, lpArgument, abBuffer, sizeof(abBuffer), &stWritten))
{
    printf("WriteProcessMemory error: \d\n", GetLastError());
    return -5;
}

hThread = CreateRemoteThread(hProcess, NULL, 0, (LPTHREAD_START_ROUTINE)lpArgument, 0, 0, &dwThreadId);
if (NULL == hThread)
{
    printf("CreateRemoteThread error: \d\n", GetLastError());
    return -6;
}

CloseHandle(hProcess);
Exercise 6:

Drivers:
Kernel Mode Code Signing
Windows requires a digitally signed driver

A recently installed program tried to install an unsigned driver. This version of Windows requires all drivers to have a valid digital signature. The driver is unavailable and the program that uses this driver might not work correctly.

Uninstall the program or device that uses this driver and check the publisher's support website to get a digitally signed driver.

Driver: Ariad Filter
Service: Ariad
Publisher: Didier Stevens (https://DidierStevens.com)
Location: C:\Windows\System32\drivers\ariad.sys
signtool.exe sign
/v
/sha1 95778C2392E6CDDAD3A725410AA7E13C6FC588EE
/t http://timestamp.verisign.com/scripts/timestamp.dll
ariad.sys

signtool.exe sign
/v
/ph /ac GSRCA.crt
/sha1 95778C2392E6CDDAD3A725410AA7E13C6FC588EE
/t http://timestamp.verisign.com/scripts/timestamp.dll
ariad.sys
The following certificate was selected:

Issued to: Didier Stevens
Issued by: GlobalSign CodeSigning CA - G2
SHA1 hash: 95778C2392E6CDDAD3A725410AA7E13C6FC588EE

Done Adding Additional Store
Successfully signed and timestamped: ariad.sys

Number of files successfully Signed: 1
Number of warnings: 0
Number of errors: 0
The following certificate was selected:
Issued to: Didier Stevens
Issued by: GlobalSign CodeSigning CA - G2
SHA1 hash: 95778C2392E6C6DAD3A725410AA7E13C6FC588EE

Cross certificate chain (using machine store):
Issued to: Microsoft Code Verification Root
Issued by: Microsoft Code Verification Root
Expires: Sat Nov 01 15:54:03 2025
SHA1 hash: 8FBE4D070EF8AB1BCCAF2A9D5CCAE7282A2C66B3

  Issued to: GlobalSign Root CA
  Issued by: Microsoft Code Verification Root
  Expires: Thu Apr 15 22:05:08 2021
  SHA1 hash: CC1DEE6BF6D55C9061BA16F10A0BFA6979A4A32

        Issued to: GlobalSign CodeSigning CA - G2
        Issued by: GlobalSign Root CA
        Expires: Sat Apr 13 12:00:00 2019
        SHA1 hash: 9000401777D2B43393D7B594D2FF4CBA4516B38

               Issued to: Didier Stevens
               Issued by: GlobalSign CodeSigning CA - G2
               SHA1 hash: 95778C2392E6C6DAD3A725410AA7E13C6FC588EE

Done Adding Additional Store
Successfully signed and timestamped: ariad.sys

Number of files successfully Signed: 1
Number of warnings: 0
Number of errors: 0
Certificate

[Image of certificate details]

Learn more about certificate details

OK
signtool verify /kp ariad-signed.sys
Successfully verified: ariad-signed.sys

signtool verify /pa ariad-simple-signed.sys
Successfully verified: ariad-simple-signed.sys
Exercise 7:

WoW64
gcc -o exercise-07.exe exercise-07.c
Wow64DisableWow64FsRedirection

C:\Windows\System32
C:\Windows\SysWOW64
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Windows\AppInit_DLLs

HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Microsoft\Windows NT\CurrentVersion\Windows\AppInit_DLLs
Exercise 8:

VBA 64-bit
http://DidierStevensLabs.com

Windows x64 The Essentials videos: €25
PDF Analysis workshop videos: €25
White Hat Shellcode workshop videos: €25
Bundle of 3 workshops: €60