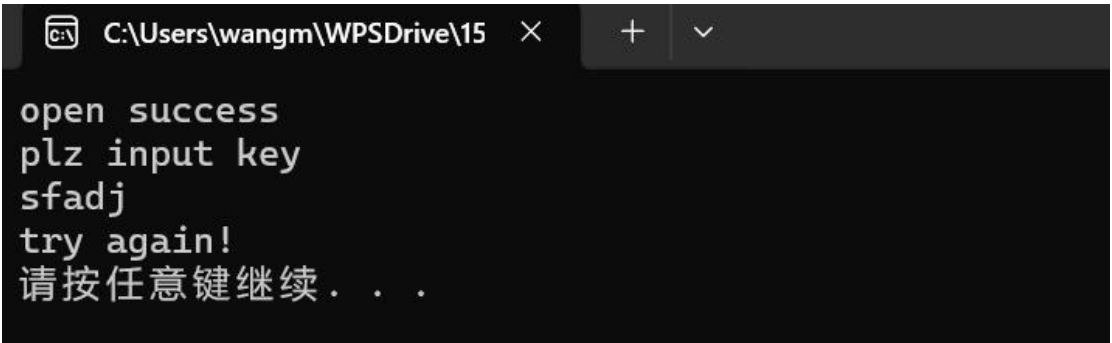


汇编语言与逆向工程第三次作业-HOOK 技术逆向分析

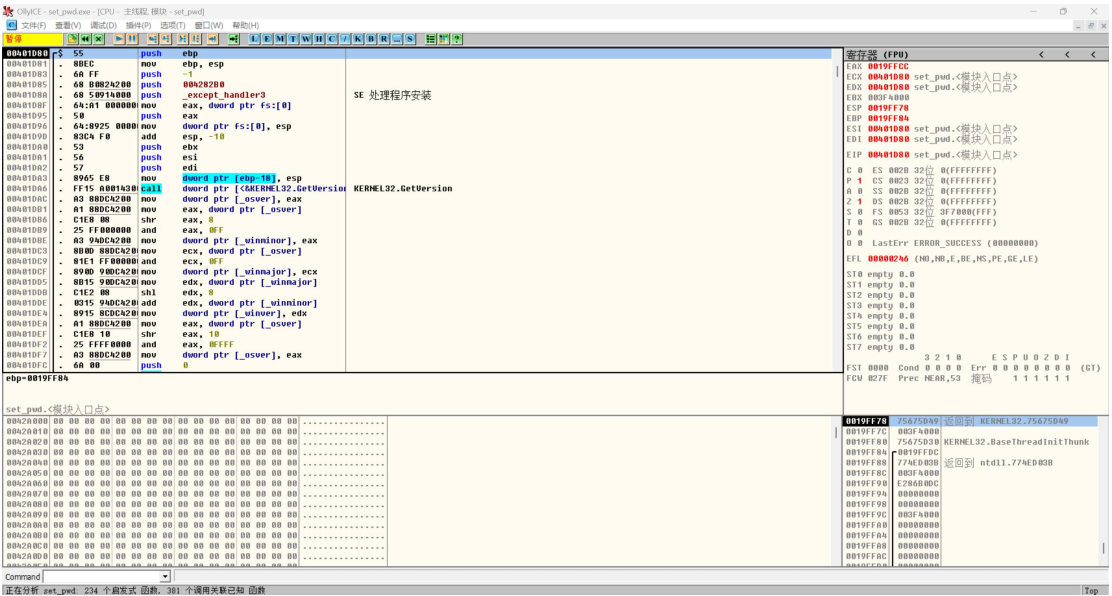
选择 WriteFileAPIHook1 文件逆向分析

首先找到要分析的程序：set_pwd.exe, 尝试运行



随便输入字符，返回“try again!”

用 OllyICE 打开：



在 main 函数处设计断点，

00401004	. CC	int3
00401005	\$~ E9 26000000	jmp unhook
0040100A	~ E9 F1000000	jmp MyWriteFile
0040100F	\$~ E9 BC010000	jmp main
00401014	CC	int3

运行，来到 main 函数开始的位置，

00401223	8BF4	mov	esi, esp	
00401225	68 1C804200	push	0042801C	pModule = "kernel32.dll"
0040122A	FF15 8801430	call	dword ptr [&KERNEL32.GetModuleHandleA]	GetModuleHandleA
00401230	3BF4	cmp	esi, esp	

发现 GetModuleHandleA 函数，取 kernel32.dll 句柄，

00401237	8945 F8	mov	dword ptr [ebp-8], eax	
0040123A	8BF4	mov	esi, esp	
0040123C	68 2C804200	push	0042802C	ProcNameOrOrdinal = "WriteFile"
00401241	8B4D F8	mov	ecx, dword ptr [ebp-8]	
00401244	51	push	ecx	hModule
00401245	FF15 8401430	call	dword ptr [&KERNEL32.GetProcAddress]	GetProcAddress
0040124B	3BF4	cmp	esi, esp	
0040124D	E8 FE050000	call	_chkesp	

句 GetProcAddress 函数取 WriteFile 函数地址，

寄存器 (FPU)			
EAX	7567EED0	jmp 到	KERNELBA.WriteFile
ECX	E29F4F00		
EDX	00000000		
EBX	003F4000		
ESP	0019FE8C		
EBP	0010FF34		

从寄存器区发现 WriteFile 函数地址为 7567EED0

00401250	8BF4	mov	esi, esp	
0040125D	8D45 B8	lea	eax, dword ptr [ebp-48]	
00401260	50	push	eax	pOldProtect
00401261	6A 40	push	40	NewProtect = PAGE_EXECUTE_READWRITE
00401263	6A 05	push	5	Size = 5
00401265	8B4D E8	mov	ecx, dword ptr [ebp-18]	
00401268	51	push	ecx	Address
00401269	FF15 8001430	call	dword ptr [&KERNEL32.VirtualProtect]	VirtualProtect
0040126F	3BF4	cmp	esi, esp	
00401271	E8 DA050000	call	_chkesp	

VirtualProtect 函数将 WriteFile 函数权限更改为“可执行可读写”

00401278	74 0F	jg	short 00401279	
0040127A	6A 05	push	5	n = 5
0040127C	8B55 E8	mov	edx, dword ptr [ebp-18]	
0040127F	52	push	edx	src
00401280	68 64DC4200	push	offset pOrgByte	dest = offset set_pwd.pOrgByte
00401285	E8 86020000	call	memcpy	memcpy
0040128A	83C4 0C	add	esp, 0C	
0040128D	B8 0A104000	mov	eax, 0040100A	

这个 memcpy 函数将 WriteFile 函数地址复制到 pOrgByte 地址中备份，

00401293	. 8BC8	mov	ecx, eax	
00401295	. 8BF2	mov	esi, edx	
00401297	. 8B45 F4	mov	eax, dword ptr [ebp-C]	
0040129A	. 99	cdq		
0040129B	. 2BC8	sub	ecx, eax	KERNEL32.WriteFile
0040129D	. 1BF2	sbb	esi, edx	
0040129F	. 83E9 05	sub	ecx, 5	
004012A2	. 83DE 00	sbb	esi, 0	
004012A5	. 894D B4	mov	dword ptr [ebp-4C], ecx	
004012A8	. 6A 04	push	4	n = 4
004012AA	. 8D55 B4	lea	edx, dword ptr [ebp-4C]	src
004012AD	. 52	push	edx	
004012AE	. 8D45 E1	lea	eax, dword ptr [ebp-1F]	dest
004012B1	. 50	push	eax	memcpy
004012B2	. E8 59020000	call	memcpy	
004012B7	. 83C4 0C	add	esp, 0C	

然后计算偏移地址，也就是在 WriteFile 函数第一句要跳转到的地址连续的减法中，ecx 为目标地址，eax 是 WriteFile 函数地址，再减去 5，最后目标地址 ecx 放入 ebp-4C 的位置。

然后再 memcpy 函数中，将目标地址放入 ebp-1F 中

004012B2	. E8 59020000	call	memcpy	memcpy
004012B7	. 83C4 0C	add	esp, 0C	
004012BA	. 6A 05	push	5	n = 5
004012BC	. 8D4D E0	lea	ecx, dword ptr [ebp-20]	src
004012BF	. 51	push	ecx	
004012C0	. 8B55 F4	mov	edx, dword ptr [ebp-C]	dest
004012C3	. 52	push	edx	memcpy
004012C4	. E8 47020000	call	memcpy	
004012C9	. 83C4 0C	add	esp, 0C	
004012CC	. 8BF4	mov	esi, esp	
004012CE	. 8D45 B8	lea	eax, dword ptr [ebp-48]	pOldProtect
004012D1	. 50	push	eax	NewProtect
004012D2	. 8B4D B8	mov	ecx, dword ptr [ebp-48]	Size = 5
004012D5	. 51	push	ecx	Address
004012D6	. 6A 05	push	5	VirtualProtect
004012D8	. 8B55 F4	mov	edx, dword ptr [ebp-C]	
004012DB	. 52	push	edx	
004012DC	. FF15 80014300	call	dword ptr [&KERNEL32.VirtualProtect]	
004012E2	. 3BF4	cmp	esi, esp	
004012E4	. E8 67050000	call	_chkesp	

再调用 memcpy 函数中，将“跳转到目标地址”这条指令放入 ebp-C 中，

再用 VirtualProtect 函数将 WriteFile 函数权限更改回权限保护。

004012E4	. E8 67050000	call	_chkesp	
004012E9	. 8BF4	mov	esi, esp	
004012EB	. 6A 00	push	0	hTemplateFile = NULL
004012ED	. 68 80000000	push	80	Attributes = NORMAL
004012F2	. 6A 02	push	2	Mode = CREATE_ALWAYS
004012F4	. 6A 00	push	0	pSecurity = NULL
004012F6	. 6A 00	push	0	ShareMode = 0
004012F8	. 68 00000010	push	10000000	Access = GENERIC_ALL
004012FD	. 68 BC004200	push	004280BC	FileName = "pwd.txt"
00401302	. FF15 98014300	call	dword ptr [&KERNEL32.CreateFileA]	CreateFileA
00401308	. 3BF4	cmp	esi, esp	
0040130A	. E8 41050000	call	_chkesp	

这里的 CreateFileA 函数猜测为创建文件函数，

00401310	51	push	ecx	ecx, dword ptr [ebp-5C]	
0040131E	E8 6D050000	call	strlen		[s
00401323	83C4 04	add	esp, 4		nBytesToWrite
00401326	50	push	eax		Buffer
00401327	8D55 A4	lea	edx, dword ptr [ebp-5C]		hFile
0040132A	52	push	edx		WriteFile
0040132B	8B45 FC	mov	eax, dword ptr [ebp-4]		
0040132E	50	push	eax		
0040132F	FF15 94014300	call	dword ptr [00401430]		
00401335	3BF4	cmp	esi, esp		
00401337	E8 14050000	call	_chkesp		

在这里调用 WriteFile 函数，运行应该会跳转到目标地址

7567EED0	-	E9 3521D88A	jmp	set_pwd.0040100A
7567EED5	^	75 CC	jnz	short 7567EEA3
7567EED7		CC	int3	
7567EED8		CC	int3	
7567EED9		CC	int3	
7567EEDA		CC	int3	

步入发现为 E9 3521D88A，跳转到目的地址

0040100A	-	E9 F1000000	jmp	MyWriteFile
0040100F	\$	E9 BC010000	jmp	main
00401014		CC	int3	

再次步入来到 MyWriteFile 的函数跳转处，步入

进入 MyWriteFile 函数

00401116	-	F3:AB	rep	stos dword ptr es:[edi]
00401118	-	A1 38804200	mov	eax, dword ptr [428038]
0040111D	-	8945 F4	mov	dword ptr [ebp-C], eax
00401120	-	8A0D 3C804200	mov	cl, byte ptr [42803C]
00401126	-	884D F8	mov	byte ptr [ebp-8], cl
00401129	-	E8 D7FEFFFF	call	00401005
0040112E	-	8BF4	mov	esi, esp
00401130	-	68 2C804200	push	00401005

一步步运行，发现 call 00401005 函数，步入

00401005	\$	E9 26000000	jmp	unhook
0040100A	-	E9 F1000000	jmp	MyWriteFile
0040100F	\$	E9 BC010000	jmp	main
00401014		CC	int3	

步入后发现调用 unhook 函数

执行到返回，回到 MyWriteFile 函数

00401120	. 8A0D 3C80420	mov	cl, byte ptr [42803C]	
00401126	. 884D F8	mov	byte ptr [ebp-8], cl	
00401129	. E8 D7FEFFFF	call	00401005	
0040112E	. 8BF4	mov	esi, esp	
00401130	. 68 2C804200	push	0042802C	ASCII "WriteFile"
00401135	. 8BFC	mov	edi, esp	
00401137	. 68 1C804200	push	0042801C	
0040113C	. FF15 8801430	call	dword ptr [<KERNEL32.GetModuleHandleA>]	[pModule = "kernel32.dll"
00401142	. 3BFC	cmp	edi, esp	GetModuleHandleA
00401144	. E8 07070000	call	_chkesp	
00401149	. 50	push	eax	
0040114A	. FF15 8401430	call	dword ptr [<KERNEL32.GetProcAddress>]	[hModule
00401150	. 3BF4	cmp	esi, esp	GetProcAddress
00401152	. E8 F9060000	call	_chkesp	
00401157	. 8945 FC	mov	dword ptr [ebp-4], eax	
0040115A	. 8BF4	mov	esi, esp	
0040115C	. 8B55 18	mov	edx, dword ptr [ebp+18]	
0040115F	. 52	push	edx	
00401160	. 8B45 14	mov	eax, dword ptr [ebp+14]	
00401163	. 50	push	eax	

这里的函数调用，再次得到 WriteFile 函数地址，为继续完成 WriteFile 函数的功能

00401152	. E8 F9060000	call	_chkesp	
00401157	. 8945 FC	mov	dword ptr [ebp-4], eax	
0040115A	. 8BF4	mov	esi, esp	
0040115C	. 8B55 18	mov	edx, dword ptr [ebp+18]	
0040115F	. 52	push	edx	
00401160	. 8B45 14	mov	eax, dword ptr [ebp+14]	
00401163	. 50	push	eax	
00401164	. 8D4D F4	lea	ecx, dword ptr [ebp-C]	
00401167	. 51	push	ecx	
00401168	. E8 23070000	call	strlen	[strlen
0040116D	. 83C4 04	add	esp, 4	
00401170	. 50	push	eax	
00401171	. 8D55 F4	lea	edx, dword ptr [ebp-C]	
00401174	. 52	push	edx	
00401175	. 8B45 08	mov	eax, dword ptr [ebp+8]	
00401178	. 50	push	eax	
00401179	. FF55 FC	call	dword ptr [ebp-4]	
0040117C	. 3BF4	cmp	esi, esp	
0040117E	. E8 CD060000	call	_chkesp	
00401183	. B8 01000000	mov	eax, 1	
00401188	. 5F	pop	edi	
00401189	. 5E	pop	esi	
0040118A	. 5B	pop	ebx	
0040118B	. 83C4 4C	add	esp, 4C	
0040118E	. 3BEC	cmp	ebp, esp	
00401190	. E8 8B060000	call	_chkesp	
00401195	. 8BE5	mov	esp, ebp	
00401197	. 5D	pop	ebp	
00401198	. C2 1400	retn	14	
0040119B	. CC	int3		

下面这里完成 WriteFile 函数的写入数据功能

0019FE00	0042802C	ASCII "WriteFile"
0019FE04	00401150	set_pwd.00401150
0019FE08	0040116D	set_pwd.0040116D
0019FE0C	0019FE64	ASCII "hook"
0019FE10	0019FEE4	

运行到压栈后，发现字符串为“hook”

寄存器 (MMX)	
EAX	00000004
ECX	0019FE64 ASCII "hook"
EDX	4BCBCAFF
EBX	00304000

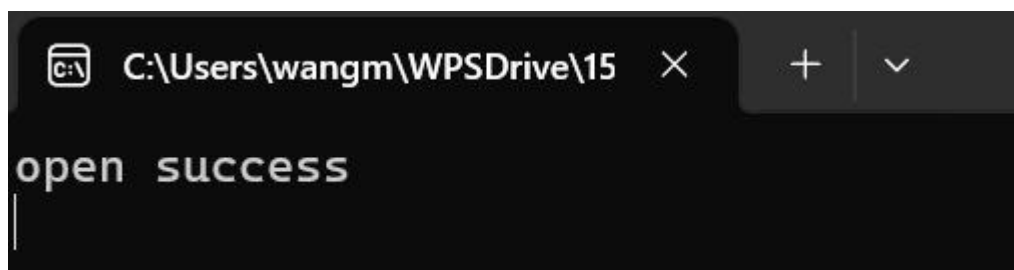
且字符串长度为 4，正确

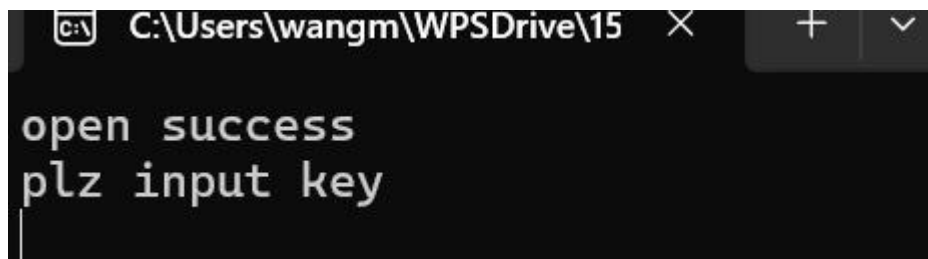
0040132E	. 50	push	eax		hFile
0040132F	. FF15 9401430	call	dword ptr [&KERNEL32.WriteFile]		WriteFile
00401335	. 3BF4	cmp	esi, esp		
00401337	. E8 14050000	call	_chkesp		
0040133C	. 8BF4	mov	esi, esp		
0040133E	. 8B4D FC	mov	ecx, dword ptr [ebp-4]		
00401341	. 51	push	ecx		hObject
00401342	. FF15 9001430	call	dword ptr [&KERNEL32.CloseHandle]		CloseHandle
00401348	. 3BF4	cmp	esi, esp		
0040134A	. E8 01050000	call	_chkesp		
0040134F	. 68 08004200	push	00428008		mode = "r"
00401354	. 68 0C004200	push	0042800C		path = "pwd.txt"
00401359	. E8 020A0000	call	fopen		fopen
0040135E	. 83C4 08	add	esp, 8		
00401361	. 8945 F0	mov	dword ptr [ebp-10], eax		

执行完返回 main 函数，发现是 cmp 指令，此时 esi 和 esp 相同，都为 0019FE8C，故，eax=1，

00401337	. E8 14050000	call	_chkesp		
0040133C	. 8BF4	mov	esi, esp		
0040133E	. 8B4D FC	mov	ecx, dword ptr [ebp-4]		
00401341	. 51	push	ecx		hObject
00401342	. FF15 9001430	call	dword ptr [&KERNEL32.CloseHandle]		CloseHandle
00401348	. 3BF4	cmp	esi, esp		
0040134A	. E8 01050000	call	_chkesp		
0040134F	. 68 08004200	push	00428008		mode = "r"
00401354	. 68 0C004200	push	0042800C		path = "pwd.txt"
00401359	. E8 020A0000	call	fopen		fopen
0040135E	. 83C4 08	add	esp, 8		
00401361	. 8945 F0	mov	dword ptr [ebp-10], eax		
00401364	. 837D F0 00	cmp	dword ptr [ebp-10], 0		
00401368	. 74 0F	je	short 00401379		
0040136A	. 68 08004200	push	00428008		format = "open success",LF,""
0040136F	. E8 7C000000	call	printf		printf
00401374	. 83C4 04	add	esp, 4		
00401377	. EB 0D	jmp	short 00401386		
00401379	. 68 08004200	push	00428008		format = "open fail",LF,""
0040137E	. E8 6D000000	call	printf		printf
00401383	. 83C4 04	add	esp, 4		
00401386	. 837D F0 00	cmp	dword ptr [ebp-10], 0		
0040138A	. 74 17	je	short 004013A3		
0040138C	. 8D55 D0	lea	edx, dword ptr [ebp-30]		
0040138F	. 52	push	edx		

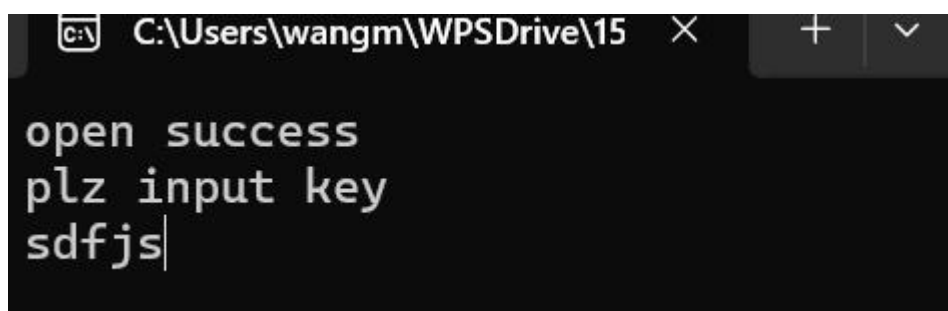
继续运行，命令行打印 “open success”





又打印 “plz input key” 的字样

004013B7	-	E8 30800000	call	printf	
004013BA	-	83C4 04	add	esp, 4	
004013BD	-	8D4D C8	lea	ecx, dword ptr [ebp-40]	
004013C0	-	51	push	ecx	
004013C1	-	68 24804200	push	00428094	
004013C6	-	E8 35070000	call	scanf	[format = "%s"
004013CB	-	83C4 08	add	esp, 8	scanf



运行搭配 scanf 函数处要求输入字符，随便输入 “sdfjs”

004013D4	-	8B55 F0	mov	edx, dword ptr [ebp-10]	
004013D7	-	52	push	edx	
004013D8	-	E8 43060000	call	fclose	[fclose
004013DD	-	83C4 04	add	esp, 4	
004013E0	>	8BF4	mov	esi, esp	
004013E2	-	8D45 D0	lea	eax, dword ptr [ebp-30]	
004013E5	-	50	push	eax	[String2
004013E6	-	8D4D C8	lea	ecx, dword ptr [ebp-40]	String1
004013E9	-	51	push	ecx	lstrcmpA
004013EA	-	FF15 8C014300	call	dword ptr [&KERNEL32.lstrcmpA]	
004013F0	-	3BF4	cmp	esi, esp	
004013F2	-	E8 59040000	call	_chkesp	
004013F7	-	8945 BC	mov	dword ptr [ebp-44], eax	
004013FA	-	837D BC 00	cmp	dword ptr [ebp-44], 0	
004013FE	~	75 0F	jnz	short 0040140F	
00401400	-	68 58042000	push	00428058	[format = "congratulations!",LF,""
00401405	-	E8 E6070000	call	printf	printf

继续步过，完成 lstrcmpA 函数比较

004013F2	-	E8 59040000	call	_chkesp	
004013F7	-	8945 BC	mov	dword ptr [ebp-44], eax	
004013FA	-	837D BC 00	cmp	dword ptr [ebp-44], 0	
004013FE	~	75 0F	jnz	short 0040140F	
00401400	-	68 58042000	push	00428058	[format = "congratulations!",LF,""
00401405	-	E8 E6070000	call	printf	printf
0040140A	-	83C4 04	add	esp, 4	
0040140D	~	E8 00	jmp	short 0040141C	
0040140F	>	68 48042000	push	00428048	[format = "try again!",LF,""
00401414	-	E8 D7070000	call	printf	printf
00401419	-	83C4 04	add	esp, 4	

由于 eax=1，不为 0，我们跳转，输出 “try again!”

```

C:\Users\wangm\WPSDrive\15
open success
plz input key
sdfjs
try again!
|

```

00401419	. 83C4 04	add esp, 4	
0040141C	> 68 40804200	push 00428040	
00401421	. E8 EA040000	call system	[command = "Pause"]
00401426	. 83C4 04	add esp, 4	system

最后调用 system 程序执行完成；

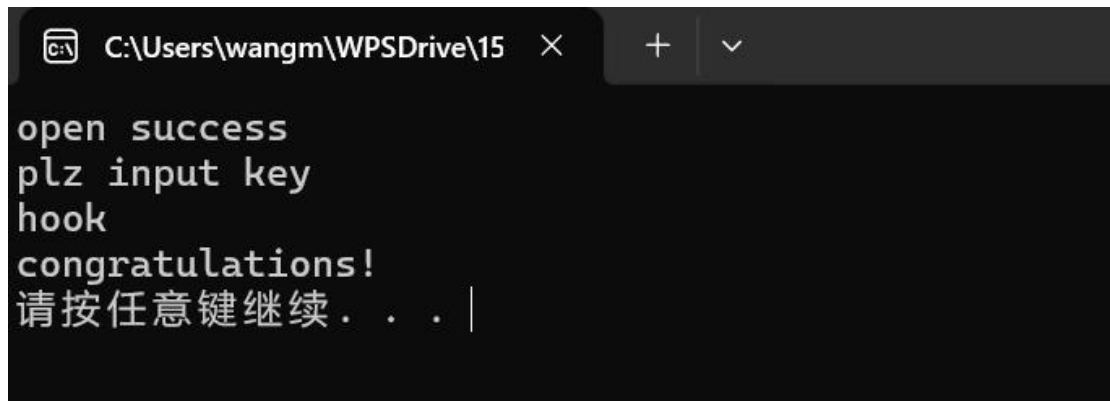
因为最后没有输出“congratulations！”，我们在输入函数 scanf 之前设置断点，运行到断点处，查看如何才能输出“congratulations！”

00401399	. E8 C2070000	call fscanf	[fscanf
0040139E	. 83C4 0C	add esp, 0C	
004013A1	. EB 0D	jmp short 004013B0	
004013A3	> 68 84804200	push 00428084	[format = "scan fail",LF,""
004013A8	. E8 43080000	call printf	printf
004013AD	. 83C4 04	add esp, 4	
004013B0	> 68 70804200	push 00428070	[format = "plz input key",LF,""
004013B5	. E8 36080000	call printf	printf
004013BA	. 83C4 04	add esp, 4	
004013BD	. 8D4D C0	lea ecx, dword ptr [ebp-40]	
004013C0	. 51	push ecx	
004013C1	. 68 94804200	push 00428094	
004013C6	. E8 35070000	call scanf	[format = "%s"
004013CB	. 83C4 08	add esp, 8	scanf
004013CE	. 837D F0 00	cmp dword ptr [ebp-10], 0	
004013D2	. 74 0C	je short 004013E0	
004013D4	. 8B55 F0	mov edx, dword ptr [ebp-10]	
004013D7	. 52	push edx	[stream
004013D8	. E8 43060000	call fclose	fclose
004013DD	. 83C4 04	add esp, 4	
004013E0	. 8BF4	mov esi, esp	
004013E2	. 8D45 D0	lea eax, dword ptr [ebp-30]	
004013E5	. 50	push eax	
004013E6	. 8D4D C0	lea ecx, dword ptr [ebp-40]	
004013E9	. 51	push ecx	
004013EA	. FF15 8C014300	call dword ptr [<&KERNEL32.1strcmpA>]	KERNEL32.1strcmpA
004013FA	. 3BF4	cmp esi, esp	

EAX	0019FF04	ASCII "hook"
ECX	0019FEF4	ASCII "mnbjg"
EDX	0042A800	set_pwd.0042A800
EBX	0025B000	
ESP	0019FE88	
EBP	0019FF34	
ESI	0019FE8C	
EDI	0019FF34	
EIP	004013E9	set_pwd.004013E9
C 0	ES	002B 32 0(FFFFFFFF)
P 0	CS	0023 32 0(FFFFFFFF)

随即输入“mnbjg”被复制到 ecx，“hook”被复制到 eax，因此，若我们输入“hook”，下面的 strcmpA 函数即可返回 ‘0’，eax 为 0，则输出“congratulations！”

验证：



```
C:\Users\wangm\WPSDrive\15 >
open success
plz input key
hook
congratulations!
请按任意键继续 . . . |
```

验证成功，即该程序的 HOOK 的作用为将 WriteFile 函数原本要写入的“real_pwd”改为“hook”。