

脱壳小记

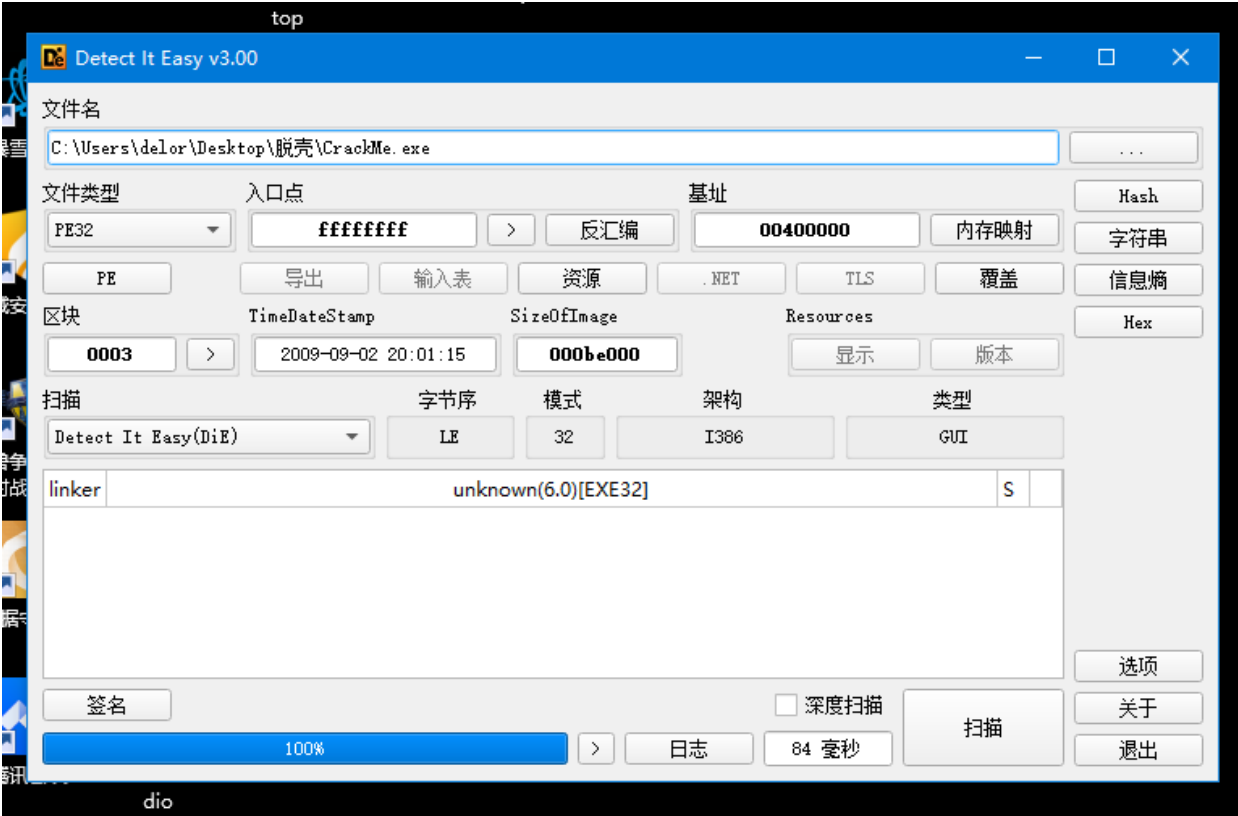
原题目链接: <https://bbs.pediy.com/thread-97892.htm>

原分析链接: <https://bbs.pediy.com/thread-101765.htm>

本文作者: Delort

本文是纯粹的脱壳, 无算法分析的内容。

1. 查壳



没有查到。

2. 寻找oep

单步步过pushad

地址	十六进制	汇编指令
0047A034	60	pushad
0047A035	9C	pushfd
0047A036	E8 8E000000	call crackme.47A0C9
0047A03B	C3	ret
0047A03C	EB 06	jmp crackme.47A044
0047A03E	> 44	inc esp
0047A03F	EB 0F	jmp crackme.47A050
0047A041	> 44	inc esp
0047A042	EB 0F	jmp crackme.47A053
0047A044	> 8B1C24	mov ebx,dword ptr ss:[esp]
0047A047	EB F5	jmp crackme.47A03E
0047A049	> FFE7	jmp edi
0047A04B	> 83EB 0A	sub ebx,A
0047A04E	EB 06	jmp crackme.47A056
0047A050	> 44	inc esp
0047A051	EB EE	jmp crackme.47A041
0047A053	> 44	inc esp

在此处esp处下硬件访问断点。

修改

二进制(B)

复制(C)

断点(K)

搜索匹配特征(E)... Ctrl+B

* 转到 ESP (S)

* 转到 EBP (B)

转到(G)

锁定堆栈

在内存布局中转到DWORD值

在反汇编中转到指定DWORD(E) Enter

在内存窗口中转到

在内存窗口中转到双字(D)

在内存窗口中转到双字(D)

监视 DWORD(W)

"`漏?"

"`漏?"

&"`漏?"

"`漏?"

"`漏?"

crackme.0047

00 (ERROR_SUCC

00 (STATUS_SUC

4 "`漏?"

0

4 ")鷗u"

00

34 "`漏?"

0019FF54 <&EntryPoint>

0019FF58 <&EntryPoint>

0019FF5C

0019FF60

0019FF64

0019FF68 <&EntryPoint>

0019FF6C

0047A034

0047A034

0019FF80

0019FF74

00383000

0047A034

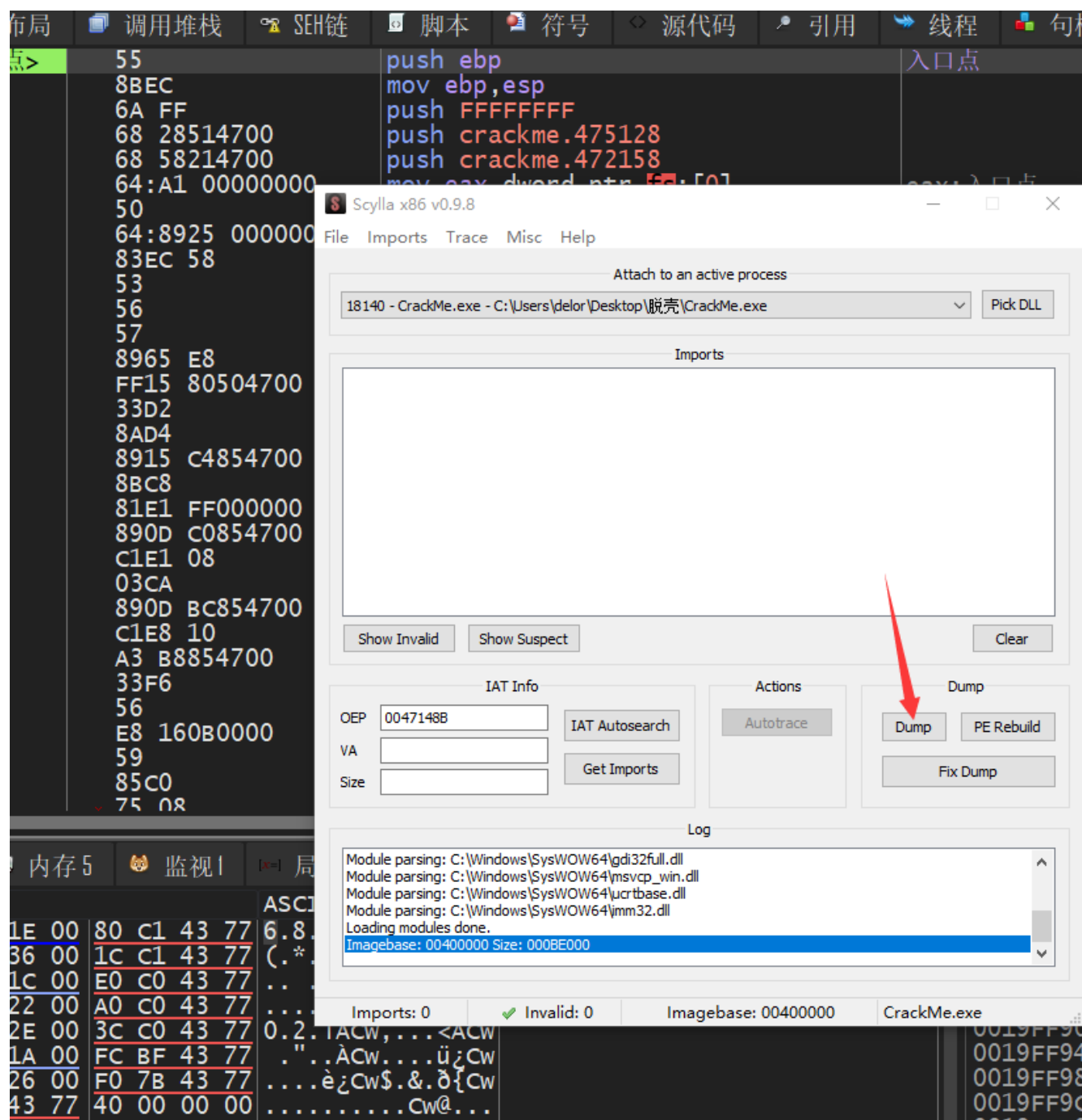
0047A034

单步步过此处为oep

地址	偏移	汇编
0047148B	<crackme.入口点>	55
0047148C		8BEC
0047148E		6A FF
00471490		68 28514700
00471495		68 58214700
0047149A		64:A1 00000000
004714A0		50
004714A1		64:8925 00000000
004714A8		83EC 58
004714AB		53
004714AC		56
004714AD		57
004714AE		8965 E8

```
push ebp
mov ebp,esp
push FFFFFFFF
push crackme.475128
push crackme.472158
mov eax,dword ptr fs:[0]
push eax
mov dword ptr fs:[0],esp
sub esp,58
push ebx
push esi
push edi
mov dword ptr ss:[ebp-18]
```

然后打开Scylla进行dump操作



3.手动重建导入表和IAT

3.1复制INT到IAT

此步骤其实可忽略，只需新建import table 并在数据目录填入RVA即可。

INT:

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	ANSI	ASCII
000B78E0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00		
000B78F0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00		
000B7900	00	00	00	00	00	00	00	00	00	00	00	00	12	58	07	00		X
000B7910	06	58	07	00	EE	57	07	00	E2	57	07	00	22	58	07	00	X	iW aW "X
000B7920	00	00	00	00	6E	5A	07	00	5C	5A	07	00	4E	5A	07	00		nZ \Z NZ
000B7930	3E	5A	07	00	32	5A	07	00	26	5A	07	00	1C	5A	07	00	>Z	2Z &Z Z
000B7940	10	5A	07	00	04	5A	07	00	7E	5A	07	00	EC	59	07	00	Z	Z ~Z iY
000B7950	DE	59	07	00	D0	59	07	00	C2	59	07	00	B2	59	07	00	BY	BY AY ^Y
000B7960	98	59	07	00	8A	59	07	00	7A	59	07	00	68	59	07	00	^Y	SY zY hY
000B7970	4E	59	07	00	36	59	07	00	94	5A	07	00	A4	5A	07	00	NY	6Y "Z wZ
000B7980	B4	5A	07	00	C6	5A	07	00	F8	59	07	00	76	58	07	00	^Z	EZ oY vX
000B7990	3E	58	07	00	52	58	07	00	64	58	07	00	84	58	07	00	>X	RX dX „X
000B79A0	92	58	07	00	A6	58	07	00	BA	58	07	00	D6	58	07	00	'X	IX °X OX
000B79B0	EC	58	07	00	06	59	07	00	20	59	07	00	00	00	00	00	iX	Y Y
000B79C0	C4	57	07	00	B8	57	07	00	A6	57	07	00	9E	57	07	00	AW	W W W zW
000B79D0	92	57	07	00	86	57	07	00	76	57	07	00	68	57	07	00	'W	tW vW hW
000B79E0	58	57	07	00	4C	57	07	00	40	57	07	00	2E	57	07	00	XW	LW @W .W
000B79F0	22	57	07	00	10	57	07	00	FE	56	07	00	EC	56	07	00	"W	W pV iV
000B7A00	DE	56	07	00	CE	56	07	00	C2	56	07	00	B4	56	07	00	EV	IV AV ^V
000B7A10	A0	56	07	00	92	56	07	00	7E	56	07	00	70	56	07	00	V	'V ~V pV
000B7A20	58	56	07	00	44	56	07	00	30	56	07	00	00	00	00	00	XV	DV oV
000B7A30	95	00	44	69	73	70	61	74	63	68	4D	65	73	73	61	67	• DispatchMessag	
000B7A40	65	41	00	00	82	02	54	72	61	6E	73	6C	61	74	65	4D	eA	, TranslateM

IAT:

000B73E0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00		
000B73F0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00		
000B7400	12	58	07	00	06	58	07	00	EE	57	07	00	E2	57	07	00	X	X iW aW
000B7410	22	58	07	00	00	00	00	00	6E	5A	07	00	5C	5A	07	00	"X	nZ \Z
000B7420	4E	5A	07	00	3E	5A	07	00	32	5A	07	00	26	5A	07	00	NZ	>Z 2Z &Z
000B7430	1C	5A	07	00	10	5A	07	00	04	5A	07	00	7E	5A	07	00	Z	Z ~Z
000B7440	EC	59	07	00	DE	59	07	00	D0	59	07	00	C2	59	07	00	iY	BY BY AY
000B7450	B2	59	07	00	98	59	07	00	8A	59	07	00	7A	59	07	00	^Y	^Y SY zY
000B7460	68	59	07	00	4E	59	07	00	36	59	07	00	94	5A	07	00	hY	NY 6Y "Z
000B7470	A4	5A	07	00	B4	5A	07	00	C6	5A	07	00	F8	59	07	00	wZ	^Z EZ oY
000B7480	76	58	07	00	3E	58	07	00	52	58	07	00	64	58	07	00	vX	>X RX dX
000B7490	84	58	07	00	92	58	07	00	A6	58	07	00	BA	58	07	00	„X	'X IX °X
000B74A0	D6	58	07	00	EC	58	07	00	06	59	07	00	20	59	07	00	OX	iX Y Y
000B74B0	00	00	00	00	C4	57	07	00	B8	57	07	00	A6	57	07	00	AW	W W W
000B74C0	9E	57	07	00	92	57	07	00	86	57	07	00	76	57	07	00	zW	'W tW vW
000B74D0	68	57	07	00	58	57	07	00	4C	57	07	00	40	57	07	00	hW	XW LW @W
000B74E0	2E	57	07	00	22	57	07	00	10	57	07	00	FE	56	07	00	.W	"W W pV
000B74F0	EC	56	07	00	DE	56	07	00	CE	56	07	00	C2	56	07	00	iV	EV IV AV
000B7500	B4	56	07	00	A0	56	07	00	92	56	07	00	7E	56	07	00	^V	V 'V ~V
000B7510	70	56	07	00	58	56	07	00	44	56	07	00	30	56	07	00	pV	XV DV oV
000B7520	00	00	00	00	00	00	00	00	FF	FF	FF	FF	62	15	47	00	yyyb	G

3.2新建导入表

找一个空白地址填入信息，我此处找的是下图地址：

000B71B0	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	
000B71C0	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	
000B71D0	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	
000B71E0	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	
000B71F0	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	
000B7200	CC CC CC CC CC CC CC CC	CC CC CC CC 34 58 07 00	iiiiiiiiiiii4X
000B7210	00 50 07 00 CC CC CC CC	CC CC CC CC CC CC CC CC	P iiiiiiiiiiiii
000B7220	D8 5A 07 00 18 50 07 00	CC CC CC CC CC CC CC CC	0Z P iiiiiiiiii
000B7230	CC CC CC CC D6 57 07 00	B4 50 07 00 00 00 00 00	iiii0w 'P
000B7240	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	
000B7250	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	
000B7260	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	
000B7270	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	
000B7280	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	

INT和IAT在文件中都是函数地址信息表，内容一致都指向函数编号和函数名。

举例：

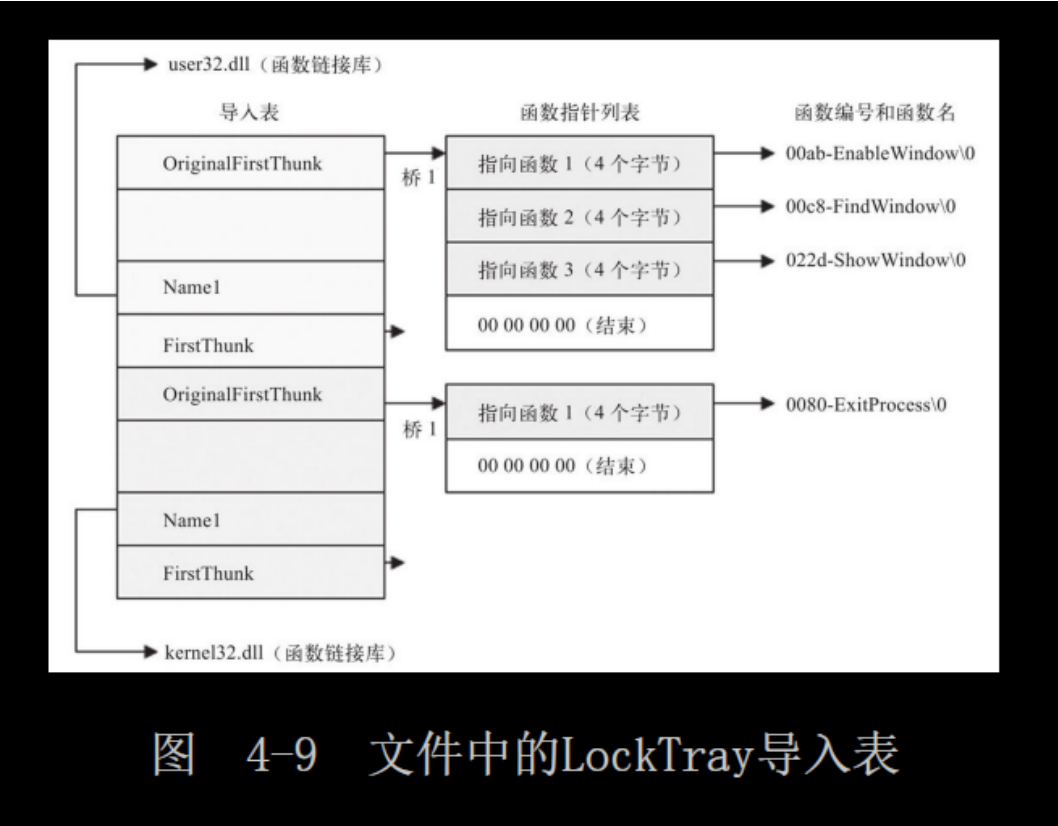


图 4-9 文件中的LockTray导入表

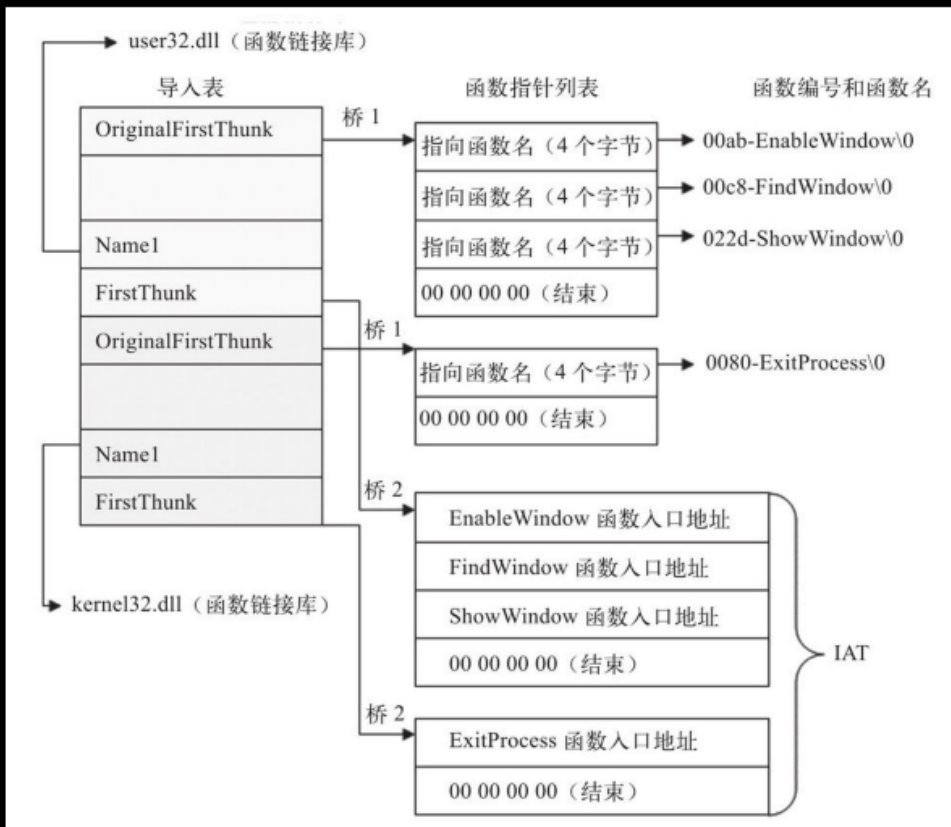


图 4-10 内存中的LockTray导入表

只不过IAT加载内存里，存的数值变成函数VA了。

3.3导入表数据结构-复习

由多个导入表描述符构成：

```

IMAGE_IMPORT_DESCRIPTOR STRUCT
    union
        Characteristics          dd    ?
        OriginalFirstThunk       dd    ? ; 0000h - 桥 1
    ends
    TimeDateStamp               dd    ? ; 0004h - 时间戳
    ForwarderChain              dd    ? ; 0008h - 链表的前一个结构
    Name1                       dd    ? ; 000ch - 指向链接库名字的指针
    FirstThunk                  dd    ? ; 0010h - 桥 2
IMAGE_IMPORT_DESCRIPTOR ENDS

```

结尾全0。

桥1或者桥2在文件中指向IMAGE_THUNK_DATA，该结构为：

```
IMAGE_THUNK_DATA STRUCT
union u1
ForwarderString dd ?
Function dd ?
Ordinal dd ?
AddressOfData dd ?
ends
IMAGE_THUNK_DATA ENDS
```

结束标志全0.

该结构又指向编号-函数名的结构。

```
IMAGE_IMPORT_BY_NAME STRUCT
Hint dw ? ;0000h-函数编号
Name1 db ? ;0004h-表示函数名的字符串
IMAGE_IMPORT_BY_NAME ENDS
```