

分析程序

找到响应事件

添加签名 导出map文件

分析算法

写出注册机

校验结果

【软件名称】： DaNiEl-RJ.1.exe

【软件大小】： 216KB

【下载地址】： <https://github.com/TonyChen56/160-Crackme>

【加壳方式】： 无壳

【保护方式】： Name/Serial

【编译语言】： Delphi

【调试环境】： W10 64

【使用工具】： OD+IDA+Darkde

【破解日期】： 2019-05-04

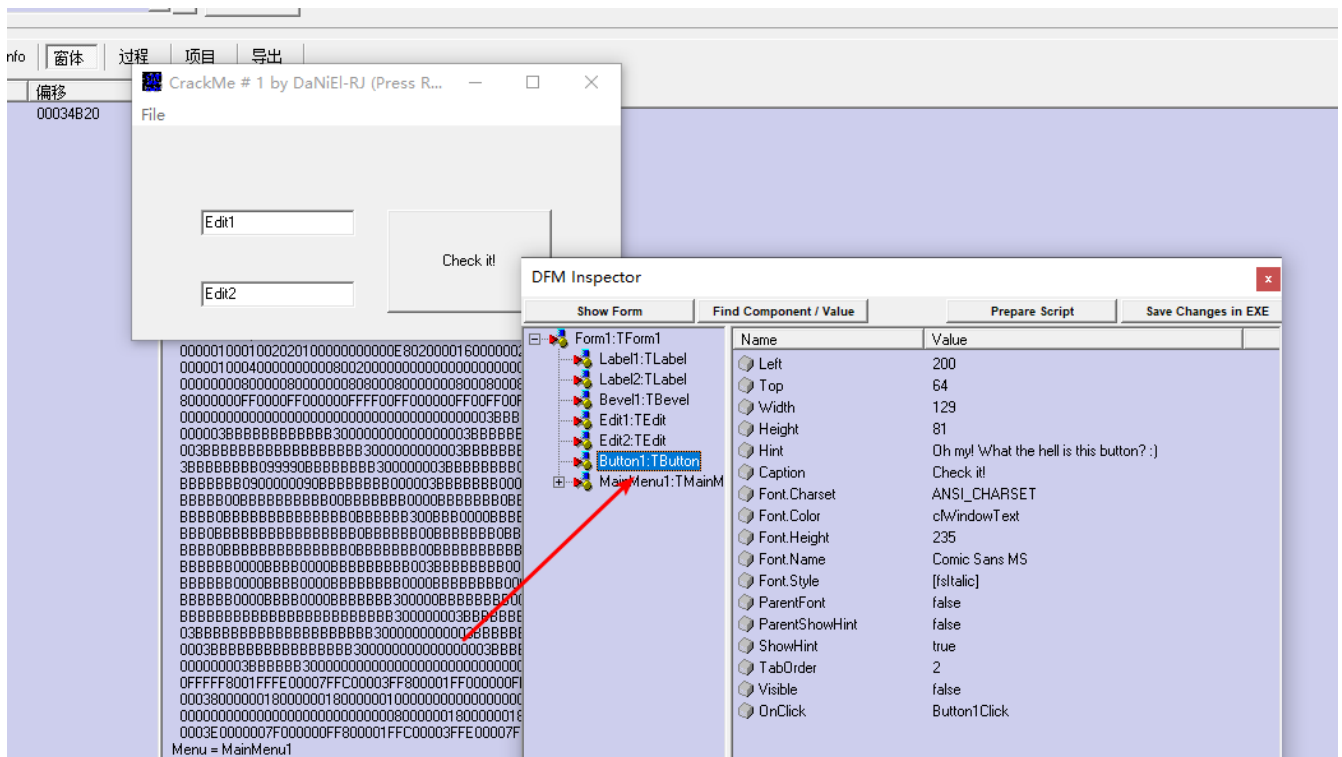
【破解目的】： 纯属兴趣

分析程序

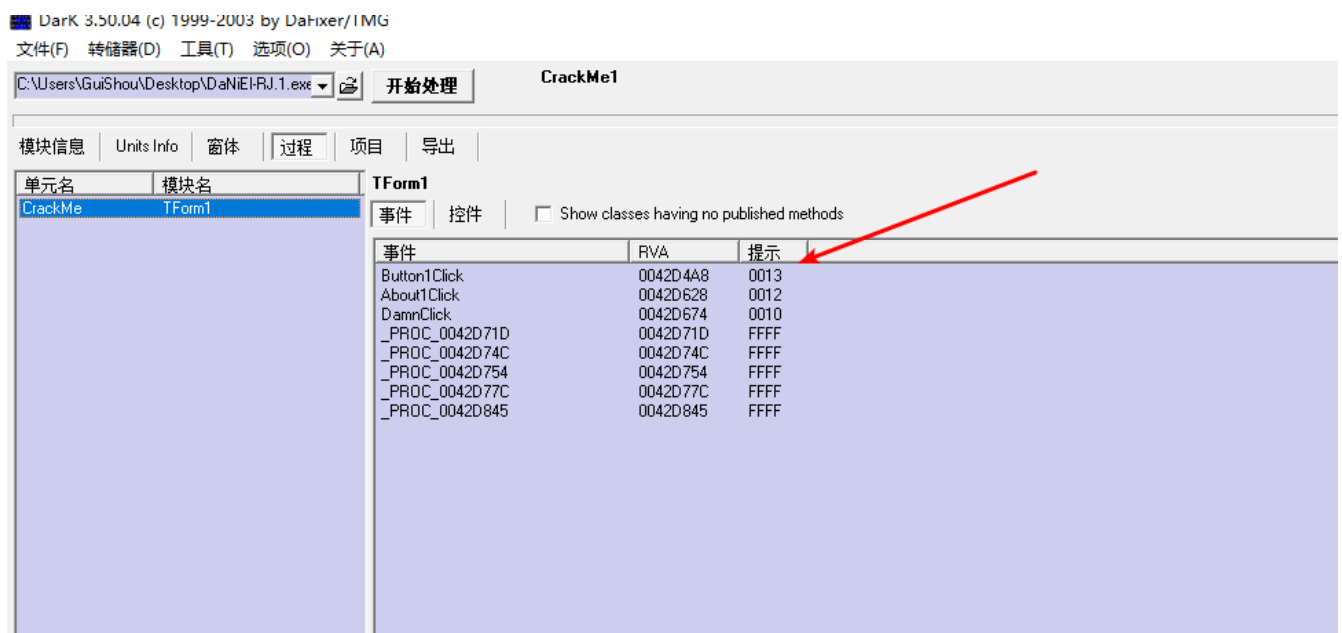
Delphi的程序利用OD+IDA+Darkde的黄金组合破解起来可以说是相当简单了。因为程序是用户名和序列号的保护方式，所以直接找到按钮事件分析算法。

找到响应事件

首先打开Darkde

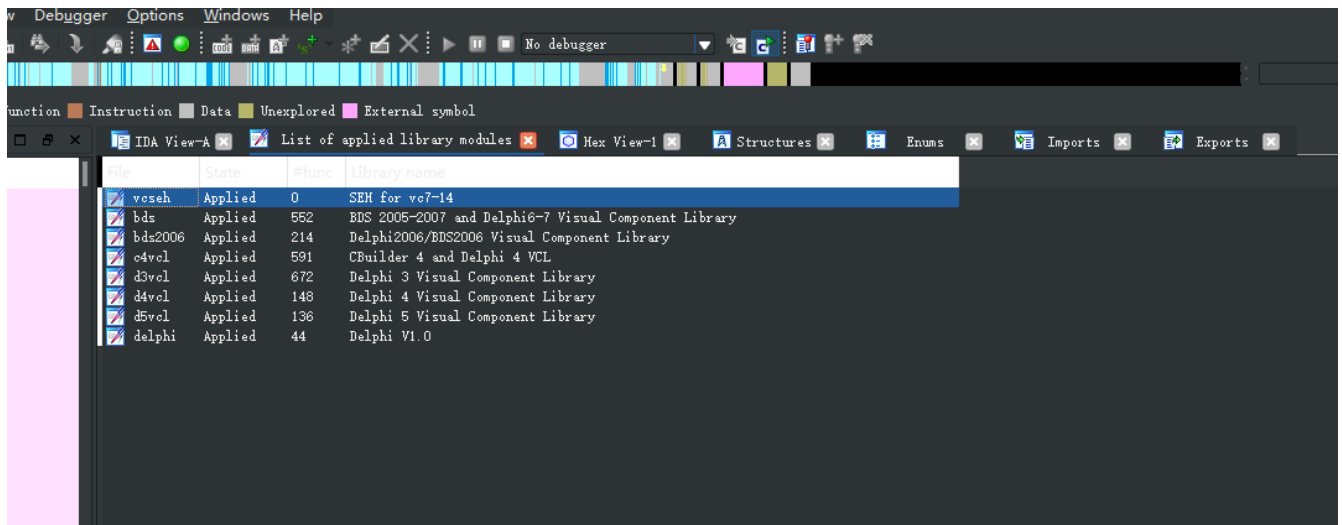


通过窗体找到按钮名，

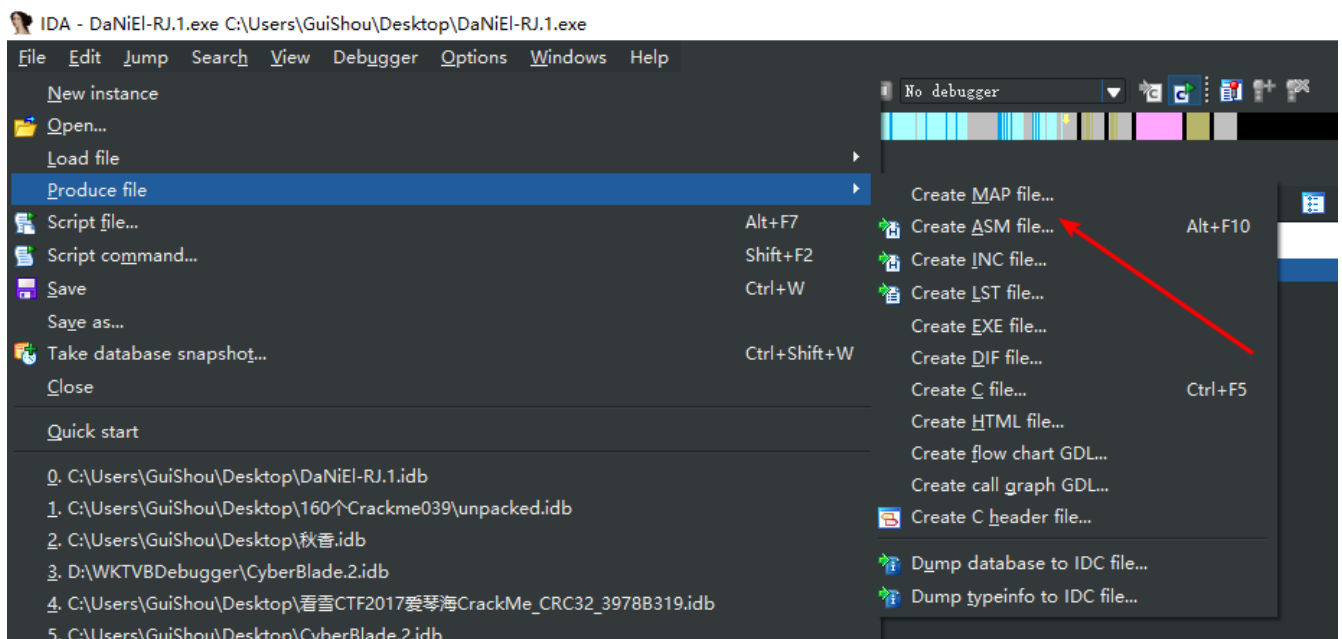


再到过程窗口根据按钮名找到响应事件

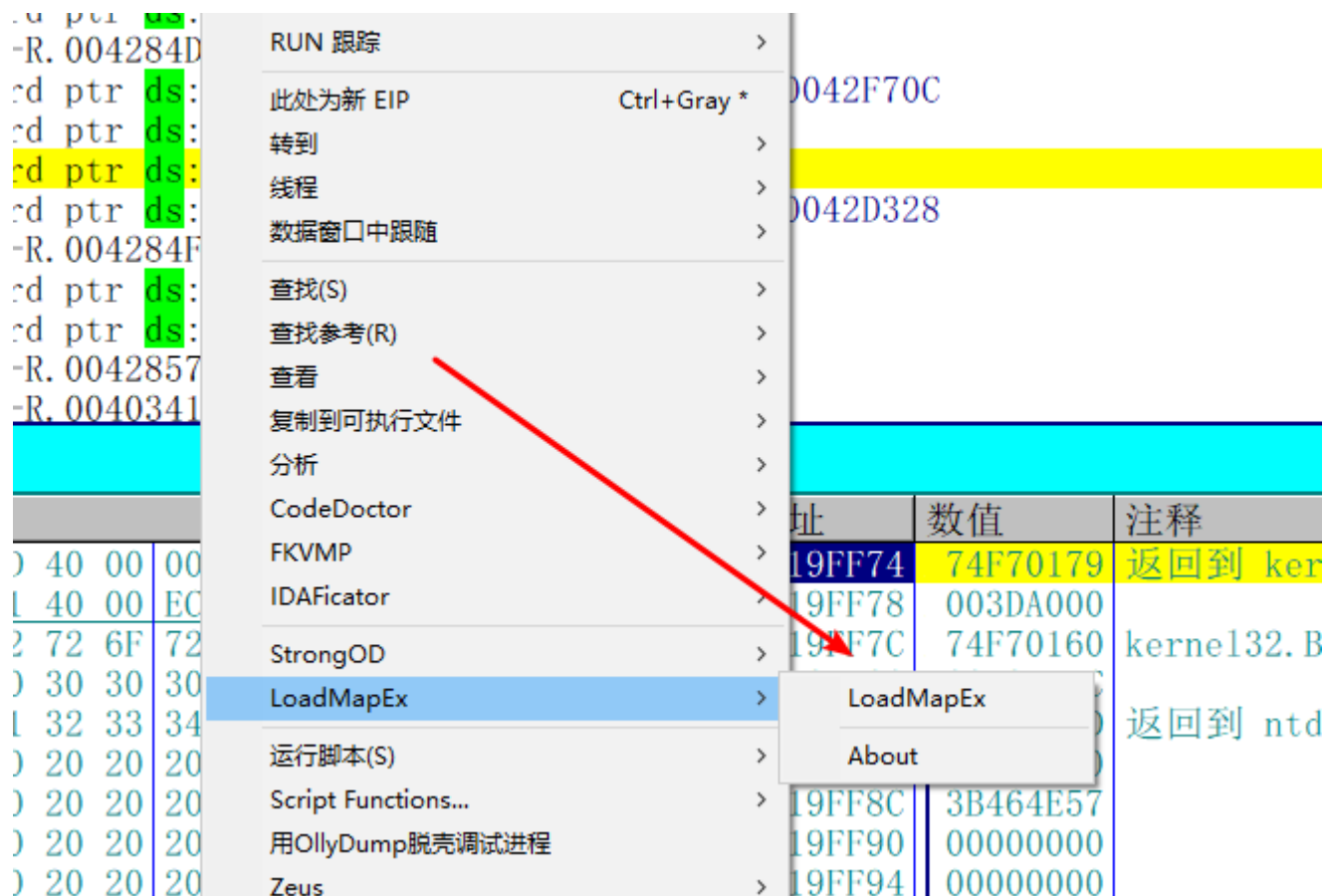
添加签名 导出map文件



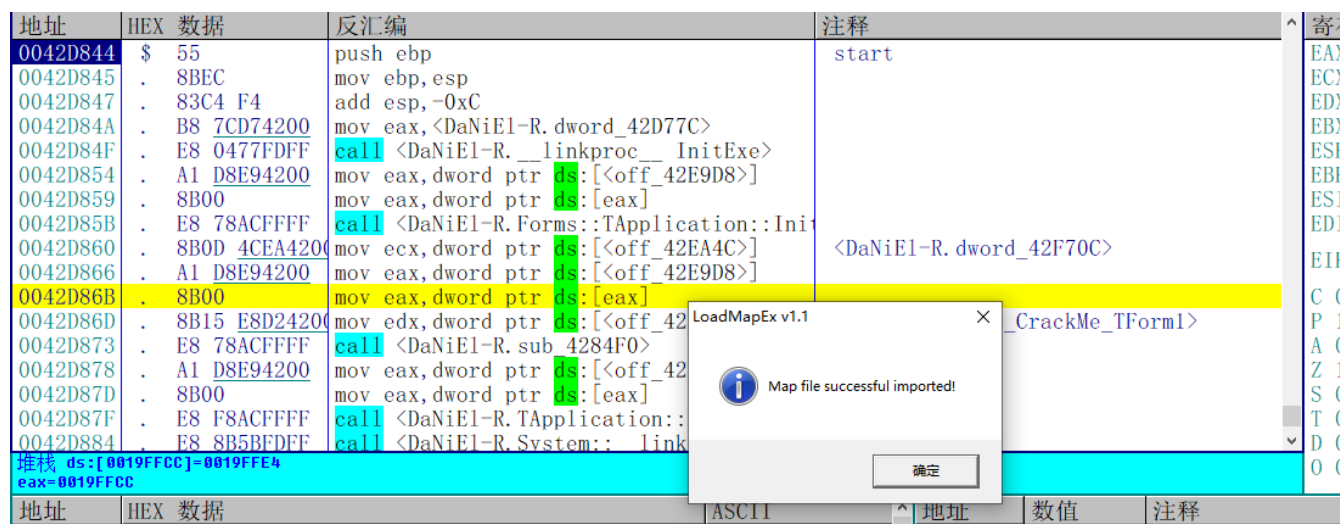
将文件拖入IDA，添加所有的Delphi签名



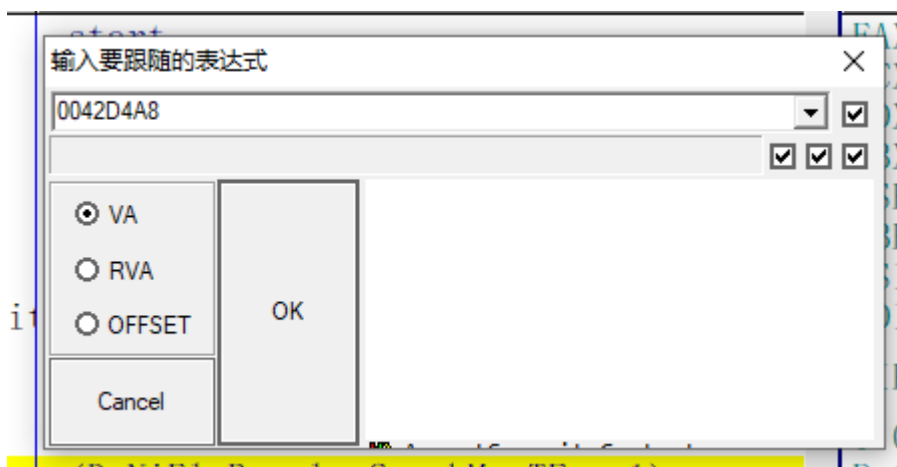
然后导出map文件



接着将map文件导入到OD

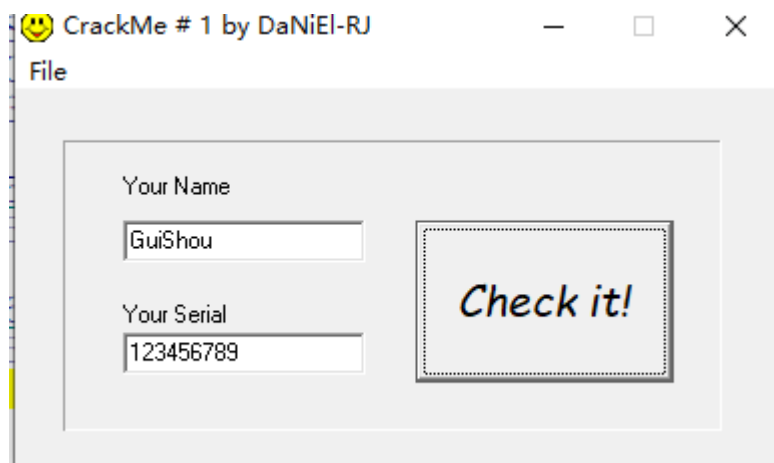


可以很清楚的看到已经IDA的符号已经被导入进来了。



然后直接去到按钮响应事件，分析注册算法

分析算法



随便输入一个用户名和序列号，开始分析算法，在IDA中算法逻辑如下

```

27  v9 = __readfsdword(0);
28  __writefsdword(0, (unsigned int)&v9);
29  Controls::TControl::GetText(*(Controls::TControl **)(a1 + 476)); // 获取用户名 检测是否为空
30  if ( username && (Controls::TControl::GetText(*(Controls::TControl **)(v4 + 480)), v16) ) // 获取序列号 检测是否为空
31  {
32      i = 1;
33      Controls::TControl::GetText(*(Controls::TControl **)(v4 + 476));
34      usernameLen = __linkproc__ LStrLen(username); // 获取用户名长度
35      System::_linkproc__ LStrClr(&v18);
36      if ( usernameLen >= 1 )
37      {
38          do
39          {
40              Controls::TControl::GetText(*(Controls::TControl **)(v4 + 476));
41              unknown_libname_886(&v15, *(unsigned __int8 *)(username + i - 1) + 5); // 将用户名的每一位ASCII+5之后转为字符
42              System::_linkproc__ LStrCat(&v18, v15); // 拼接转换后的每一个字符
43              ++i;
44          }
45          while ( usernameLen >= i );
46      }
47      Controls::TControl::GetText(*(Controls::TControl **)(v4 + 480)); // 获取序列号
48      System::_linkproc__ LStrCmp(username, v18); // 将序列号和用户名拼接的字符进行比较
49      if ( v7 )
50          ShowMessage(&str_Congratz_cracke[1]);
51      else

```

下面进行逐步讲解

1. 获取用户名 检测是否为空

地址	HEX 数据	反汇编	注释	寄存器 (FPU)	
0042D4B3	. 57	push edi		EAX 00000007	
0042D4B4	. 8BF0	mov esi, eax		ECX 4223F198	
0042D4B6	. 33C0	xor eax, eax		EDX 00000000	
0042D4B8	. 55	push ebp		EBX 021A6E00	
0042D4B9	. 68 B2D54200	push <DaNiEl-R. loc_42D5B2>		ESP 0019F678	
0042D4BE	. 64:FF30	push dword ptr fs:[eax]		EBP 0019F6A0	
0042D4C1	. 64:8920	mov dword ptr fs:[eax], esp		ESI 021A4AAC	
0042D4C4	. 8D55 F8	lea edx, [local.2]		EDI 021A6E00	
0042D4C7	. 8B86 DC010000	mov eax, dword ptr ds:[esi+0x1DC]		EIP 0042D4D2 Da	
0042D4CD	. E8 8EC9FEFF	call <DaNiEl-R. Controls::TControl::GetText(void)>	获取用户名		
0042D4D2	. 837D F8 00	cmp [local.2], 0x0	检测是否为空	C 0 ES 002B 32	
0042D4D6	. 74 14	je short <DaNiEl-R. loc_42D4EC>		P 0 CS 0023 32	
0042D4D8	. 8D55 F4	lea edx, [local.3]		A 0 SS 002B 32	
0042D4DB	. 8B86 E0010000	mov eax, dword ptr ds:[esi+0x1E0]		Z 0 DS 002B 32	
0042D4E1	. E8 7AC9FEFF	call <DaNiEl-R. Controls::TControl::GetText(void)>	获取序列号	S 0 FS 0053 32	
0042D4E6	. 837D F4 00	cmp [local.3], 0x0	检测是否为空	T 0 GS 002B 32	
0042D4EA	. 75 0F	jnz short <DaNiEl-R. loc_42D4FB>		D 0	
堆栈 ss:[0019F698]=021A6614, (ASCII "GuiShou")				O 0 LastErr ER	
地址	数值	注释	地址	数值	注释
0019F698	021A6614	ASCII "GuiShou"	0019F678	0019F788	指向下一个 SEH 记录的指针
0019F69C	00000000		0019F67C	0042D5B2	SE处理程序
0019F6A0	0019F710		0019F680	0019F6A0	
0019F6A4	0041A857	返回到 DaNiEl-R. Db::TDataSet::DoAfterOpen(void)+1B	0019F684	021A6E00	

2. 获取序列号 检测是否为空

吾爱破解 - DaNiEl-RJ.1.exe - [LCG - 主线程 模块 - DaNiEl-RJ]					
文件(F) 查看(V) 调试(D) 插件(P) 选项(T) 窗口(W) 帮助(H) [+] 快捷菜单 Tools BreakPoint->					
暂停					
l e m t w h c p k b r ... s					
寄存器 (FPU)					
地址	HEX 数据	反汇编	注释	EAX 00000009 ECX 4223F198 EDX 00000000 EBX 021A6E00 ESP 0019F678 EBP 0019F6A0 ESI 021A4AAC EDI 021A6E00 EIP 0042D4E6 DaNi	
0042D4B6	. 33C0	xor eax, eax		C 0 ES 002B 32位	
0042D4B8	. 55	push ebp		P 0 CS 0023 32位	
0042D4B9	. 68 B2D54200	push <DaNiEl-R. loc_42D5B2>		A 0 SS 002B 32位	
0042D4BE	. 64:FF30	push dword ptr fs:[eax]		Z 0 DS 002B 32位	
0042D4C1	. 64:8920	mov dword ptr fs:[eax], esp		S 0 FS 0053 32位	
0042D4C4	. 8D55 F8	lea edx, [local.2]		T 0 GS 002B 32位	
0042D4C7	. 8B86 DC010000	mov eax, dword ptr ds:[esi+0x1DC]		D 0	
0042D4CD	. E8 8EC9FEFF	call <DaNiEl-R. Controls::TControl::GetText(void)>	获取用户名	O 0 LastErr ERRO	
0042D4D2	. 837D F8 00	cmp [local.2], 0x0	检测是否为空		
0042D4D6	. 74 14	je short <DaNiEl-R. loc_42D4EC>			
0042D4D8	. 8D55 F4	lea edx, [local.3]			
0042D4DB	. 8B86 E0010000	mov eax, dword ptr ds:[esi+0x1E0]			
0042D4E1	. E8 7AC9FEFF	call <DaNiEl-R. Controls::TControl::GetText(void)>	获取序列号		
0042D4E6	. 837D F4 00	cmp [local.3], 0x0	检测是否为空		
0042D4EA	. 75 0F	jnz short <DaNiEl-R. loc_42D4FB>			
0042D4EC	. B8 C8D54200	mov eax, DaNiEl-R. 0042D5C8	loc_42D4EC		
0042D4F1	. E8 02FCFEFF	call <DaNiEl-R. @ShowMessage>			
堆栈 ss:[0019F694]=021A4ED4, (ASCII "123456789")					
地址	数值	注释	地址	数值	注释
0019F694	021A4ED4	ASCII "123456789"	0019F678	0019F788	指向下一个 SEH 记录的指针

3. 初始化循环 并获取用户名长度

文件(F) 查看(V) 调试(D) 插件(P) 选项(O) 窗口(W) 帮助(H) 快捷菜单 Tools BreakPoint->				
暂停				
l e m t w h c p k b r f ... s				
C 0				

4. 获取用户名每一位的ASCII值+5

DaNiEl-RJ.1.exe - [LCG - 主线程 模块 - DaNiEl-R]					
文件(F) 查看(V) 调试(D) 插件(P) 选项(T) 窗口(W) 帮助(H) [+] 快速菜单 Tools BreakPoint->					
暂停					
1 e m t w h c p k b r ... s					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					
[Icons]					

地址	HEX 数据	反汇编	注释	寄存器 (FPU)
042D516	. 8BF8	mov edi,eax		EAX 0019F690
042D518	. 8D45 FC	lea eax,[local.1]		ECX 4223F198
042D51B	. E8 2460FDFE	call <DaNiEl-R.System::__linkproc__ LStrClr(System::>		EDX 0000004C
042D520	. 3BF8	cmp edi,ebx		EIP 00000001
042D522	. 7C 32	jl short <DaNiEl-R.loc_42D556>		ESP 0019F678
042D524	. 8D55 F8	lea edx,[local.2]	loc_42D524	EBP 0019F6A0
042D527	. 8B86 DC010000	mov eax,dword ptr ds:[esi+0x1DC]		ESI 021A4AAC
042D52D	. E8 2EC9FEFF	call <DaNiEl-R.Controls::TControl::GetText(void)>		EDI 00000007
042D532	. 8B45 F8	mov eax,[local.2]	eax=username	EIP 0042D541 DaNi
042D535	. 33D2	xor edx,edx		C 0 ES 002B 32位
042D537	. 8A5418 FF	mov dl,byte ptr ds:[eax+ebx-0x1]	dl=username[i]	P 0 CS 0023 32位
042D53B	. 83C2 05	add edx,0x5	edx=dl=username[i]+0x5	A 0 SS 002B 32位
042D53E	. 8D45 F0	lea eax,[local.4]		

5. 将用户名的ASCII值+5后转为字符

0042D53B	. 83C2 05	add edx,0x5	edx=dl=username[i]+0:
0042D53E	. 8D45 F0	lea eax,[local.4]	
0042D541	. E8 A261FDFE	call <DaNiEl-R.unknown_libname_886>	将edi转为字符
0042D546	. 8B55 F0	mov edx,[local.4]	
0042D549	. 8D45 FC	lea eax,[local.1]	
0042D54C	. E8 7762FDFE	call <DaNiEl-R.System::__linkproc__ LStrCat(void)>	拼接转换后的字符
0042D551	. 43	inc ebx	i++
堆栈 ss:[0019F698]=021A0FC4, (UNICODE "L")			
edx=0000004C			

地址	HEX 数据	ASCII	地址	数值	注释
021A0FC4	4C 00 00 00	16 00 00 00	01 00 00 00	06 00 00 00	L... □... □...
021A0FD4	53 63 72 69	70 74 00 00	16 00 00 00	01 00 00 00	Script... □... □
021A0FE4	09 00 00 00	43 6F 75 72	69 65 72 00	0A 00 00 00	□... Courier. □
021A0FF4	01 00 00 00	08 00 00 00	4D 53 20 53	65 72 69 66	□... □... MS Se
021A1004	00 00 00 00	1E 00 00 00	01 00 00 00	0D 00 00 00 □.....
021A1014	4D 53 20 53	61 6E 73 20	53 65 72 69	66 00 00 00	MS Sans Serif..
021A1024	6E 00 00 00	18 F0 41 00	AC 4A 1A 02	C0 6B 1A 02	n... □餉. 培□ 絲
021A1034	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
021A1044	00 00 00 00	00 01 00 00	C4 10 1A 02	00 00 00 00 □..?□ ..
021A1054	00 01 00 00	01 00 00 00	02 00 00 00	00 00 00 00	□ □

6. 拼接转换后的字符 并开始新一轮循环

0042D541	. E8 A261FDFE	call <DaNiEl-R.unknown_110name_886>	将edi转为字符
0042D546	. 8B55 F0	mov edx,[local.4]	
0042D549	. 8D45 FC	lea eax,[local.1]	
0042D54C	. E8 7762FDFE	call <DaNiEl-R.System::__linkproc__ LStrCat(void)>	拼接转换后的字符
0042D551	. 43	inc ebx	i++
0042D552	. 3BF8	cmp edi,ebx	
0042D554	. 7D CE	jge short <DaNiEl-R.loc_42D524>	
0042D556	> 8D55 F8	lea edx,[local.2]	loc_42D556
0042D559	. 8B86 E0010000	mov eax,dword ptr ds:[esi+0x1E0]	
0042D55F	. E8 FCC8FEFF	call <DaNiEl-R.Controls::TControl::GetText(void)>	获取序列号
0042D564	. 8B45 F8	mov eax,[local.2]	
跳转已实现			

直接在42D556循环结束后下断点，可以看到最后拼接的结果

0042D54C	. E8 7762FDFF	call <DaNiEl-R.System::__linkproc__ LStrCat(void)>	拼接转换后的字符		
0042D551	. 43	inc ebx	i++		
0042D552	. 3BFB	cmp edi,ebx			
0042D554	. ^ 7D CE	jge short <DaNiEl-R. loc_42D524>			
0042D556	> 8D55 F8	lea edx,[local.2]	loc_42D556		
0042D559	. 8B86 E0010000	mov eax,dword ptr ds:[esi+0x1E0]			
0042D55F	. E8 FCC8FEFF	call <DaNiEl-R.Controls::TControl::GetText(void)>	获取序列号		
0042D564	. 8B45 F8	mov eax,[local.2]			
堆栈地址=0019F698 edx=021A661A, (UNICODE "z&")					
地址	数值	注释	地址	数值	注释
0019F698	021A4F38	ASCII "GuiShou"	0019F678	0019F788	指向下一个 SEH 记录的指针
0019F69C	021A6614	ASCII "LznXmtz"	0019F67C	0042D5B2	SE处理程序
0019F6A0	0019F710		0019F680	0019F6A0	
0019F6A4	0041A857	返回到 DaNiEl-R.Db::TDataSet::DoAfterOpen(void)+1B	0019F684	021A6E00	
0019F6A8	021A6E00		0019F688	021A6E00	
0019F6AC	0042B795	返回到 DaNiEl-R.0042B795 来自 <DaNiEl-R.Db::TDataSet::DoAfterOpen(void)>	0019F68C	021A6E00	
0019F6B0	0041A857	返回到 DaNiEl-R.Db::TDataSet::DoAfterOpen(void)+1B	0019F690	021A6E00	

7. 获取序列号 将序列号和拼接的字符进行比较

文件(F) 查看(V) 调试(D) 插件(P) 选项(T) 窗口(W) 帮助(H) [+] 快捷菜单 Tools BreakPoint->					
暂停					
地址	HEX	数据	反汇编	注释	寄存器 (FPU)
0042D53E	. 8D45 F0		lea eax,[local.4]		EAX 021A144C ASCII "123456789"
0042D541	. E8 A261FDFF		call <DaNiEl-R.unknown_libname_886>	将edi转为字符	ECX 4223F198
0042D546	. 8B55 F0		mov edx,[local.4]		EDX 021A6614 ASCII "LznXmtz"
0042D549	. 8D45 FC		lea eax,[local.1]		EBX 00000008
0042D54C	. E8 7762FDFF		call <DaNiEl-R.System::__linkproc__ LStrCat(void)>	拼接转换后的字符	ESP 0019F678
0042D551	. 43		inc ebx	i++	EBP 0019F6A0
0042D552	. 3BFB		cmp edi,ebx		ESI 021A4AAC
0042D554	. ^ 7D CE		jge short <DaNiEl-R. loc_42D524>		EDI 00000007
0042D556	> 8D55 F8		lea edx,[local.2]	loc_42D556	EIP 0042D56A DaNiEl-R.0042D56A
0042D559	. 8B86 E0010000		mov eax,dword ptr ds:[esi+0x1E0]		C 0 ES 002B 32位 0(FFFFFFFF)
0042D55F	. E8 FCC8FEFF		call <DaNiEl-R.Controls::TControl::GetText(void)>	获取序列号	P 0 CS 0023 32位 0(FFFFFFFF)
0042D564	. 8B45 F8		mov eax,[local.2]		A 0 SS 002B 32位 0(FFFFFFFF)
0042D567	. 8B55 FC		mov edx,[local.1]		Z 0 DS 002B 32位 0(FFFFFFFF)
0042D56A	. E8 6163FDFF		call <DaNiEl-R.System::__linkproc__ LStrCmp(void)>	将序列号和拼接的字符进行比较	S 0 FS 0053 32位 3DD000(FFF)
0042D56F	. 75 0C		jnz short <DaNiEl-R. loc_42D57D>		T 0 GS 002B 32位 0(FFFFFFFF)
0042D571	. B8 ECD54200		mov eax,DaNiEl-R.0042D5EC		D 0
0042D576	. E8 7DEBFEFF		call <DaNiEl-R.@ShowMessage>		O 0 LastErr ERROR_SUCCESS (000)
004038D0<DaNiEl-R.System::__linkproc__ LStrCmp(void)>					
地址	数值	注释	地址	数值	注释
0019F698	021A144C	ASCII "123456789"	0019F678	0019F788	指向下一个 SEH 记录的指针
0019F69C	021A6614	ASCII "LznXmtz"	0019F67C	0042D5B2	SE处理程序
0019F6A0	0019F710		0019F680	0019F6A0	
0019F6A4	0041A857	返回到 DaNiEl-R.Db::TDataSet::DoAfterOpen(void)+1B	0019F684	021A6E00	

0042D564	. 8B45 F8	mov eax,[local.2]	
0042D567	. 8B55 FC	mov edx,[local.1]	
0042D56A	. E8 6163FDFD	call <DaNiEl-R.System::__linkproc__ LStrCmp(void)>	
0042D56F	. 75 0C	jnz short <DaNiEl-R.loc_42D57D>	
0042D571	. B8 ECD54200	mov eax,DaNiEl-R.0042D5EC	
0042D576	. E8 7DFBFFFF	call <DaNiEl-R.@ShowMessage>	
0042D57B	. EB 0A	jmp short <DaNiEl-R.loc_42D587>	
0042D57D	> B8 10D64200	mov eax,DaNiEl-R.0042D610	1
0042D582	. E8 71FBFFFF	call <DaNiEl-R.@ShowMessage>	
0042D587	> 33C0	xor eax,eax	1
0042D589	. 5A	pop edx	C
0042D58A	. 59	pop ecx	C
0042D58B	. 59	pop ecx	C
0042D58C	. 64:8910	mov dword ptr fs:[eax],edx	
0042D5F8=<DaNiEl-R.@ShowMessage>			

地址	HEX 数据	ASCII	地址
0042D5EC	43 6F 6E 67 72 61 74 7A 20 63 72 61 63 6B 6E	Congratz cracker	0019F6
0042D5FC	21 20 68 65 68 65 68 65 00 00 00 00 FF FF FF	! hehehe...	0019F6
0042D60C	17 00 00 00 4E 6F 20 6E 6F 20 6E 6F 21 20 3E	□...No no no! :	0019F6
0042D61C	20 54 72 79 20 61 67 61 69 6E 21 00 53 8B D8	Try again!.S嬢?	0019F6
0042D62C	4C D6 42 00 E8 C2 FA FF EF E2 01 8B 83 E8 01	1 辞 本? 2 娘?	0019F6

根据比较的结果提示是否注册成功

写出注册机

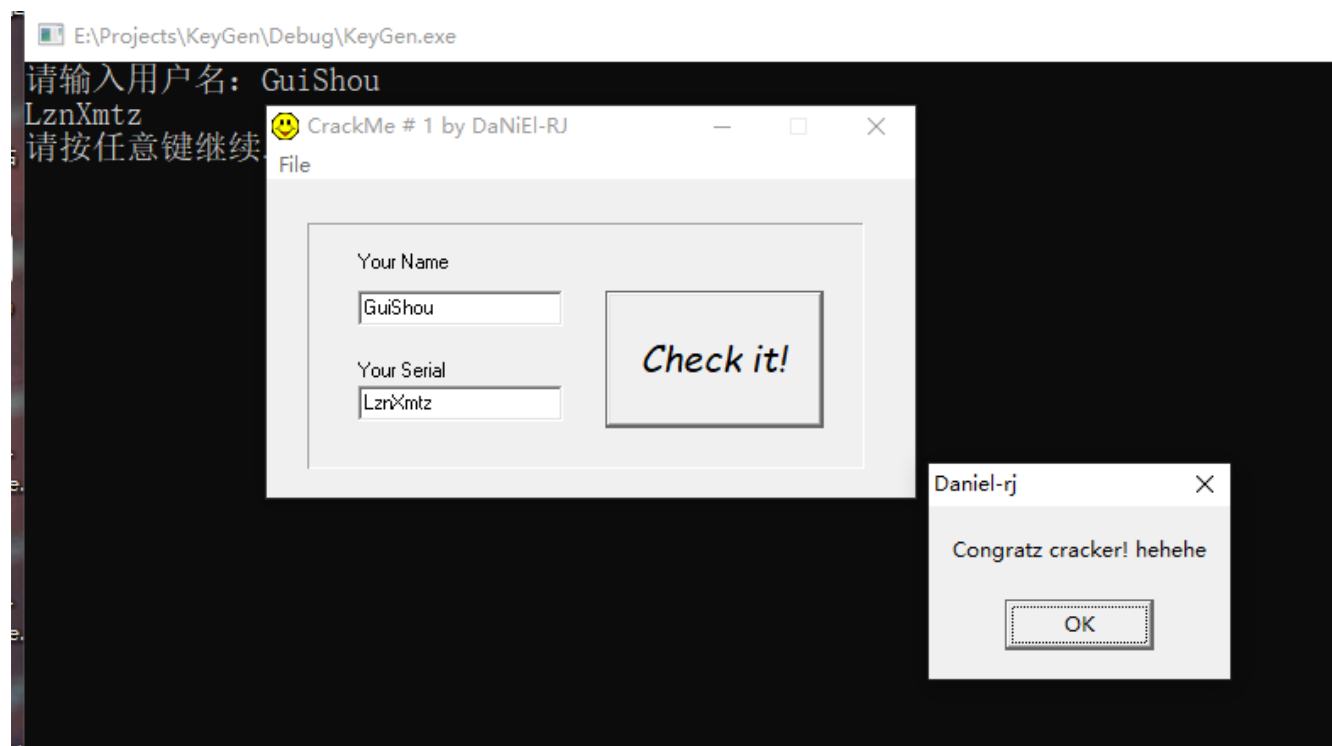
这个算法还是相对比较简单，直接写出注册机，代码如下

```
#include "pch.h"
#define _CRT_SECURE_NO_WARNINGS
#include <windows.h>
#include <stdio.h>

int main()
{
    char username[20] = { 0 };
    char serial[20] = { 0 };
    printf("请输入用户名: ");
    scanf_s("%s", username, 20);
    for (int i = 0; i < strlen(username); i++)
    {
        serial[i] = username[i] + 5;
    }
    printf("%s\n", serial);
    system("pause");
    return 0;
}
```

校验结果

输入用户名和计算的序列号，提示注册成功 破解完成



需要注册机和相关文件可以到我的Github下载: <https://github.com/TonyChen56/160-Crackme>