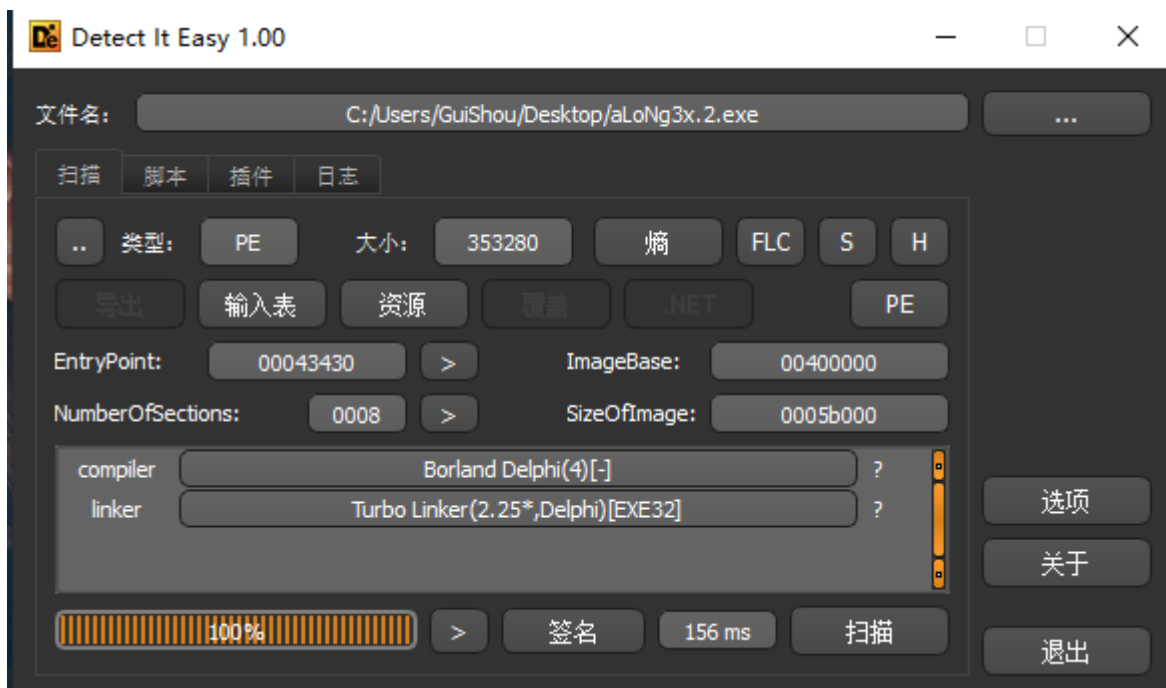


- 查壳
- Darkde分析程序
- 导入符号
- 分析Register按钮事件
 - 分析核心算法
 - 写出注册机
 - 校验结果
- 分析again按钮事件
- 校验步骤

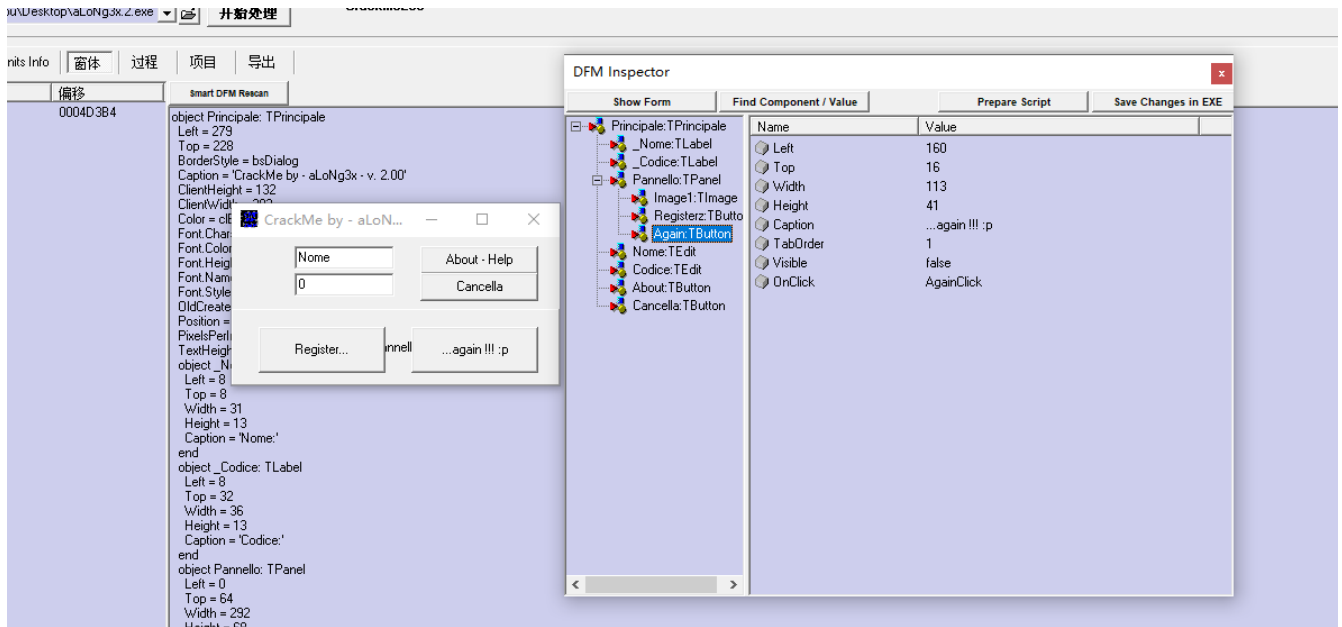
007这个crackme跟006是同一个作者，只不过难度上升了一颗星。先来查一下壳吧

查壳

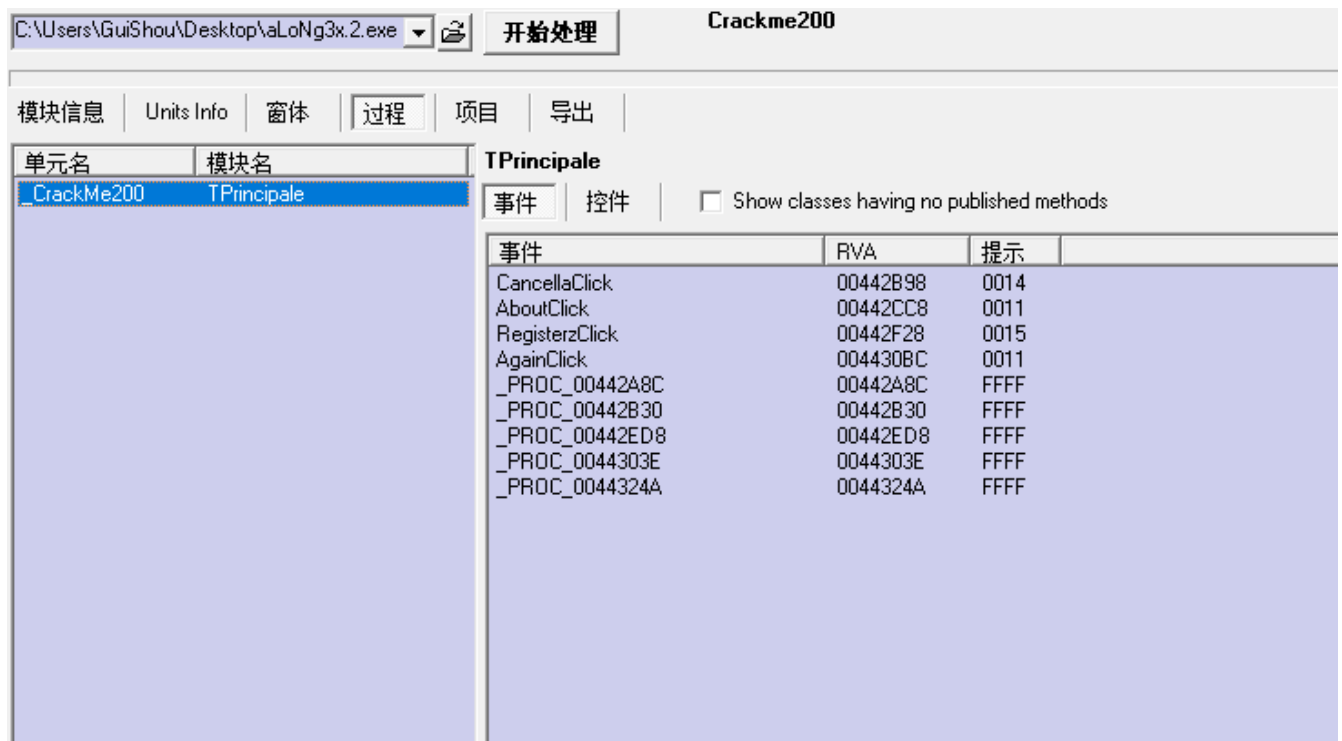


还是Delphi写的程序，没有壳。接下来用Darkde分析一下程序。

Darkde分析程序



右边这个again按钮是被隐藏的。然后再来看事件

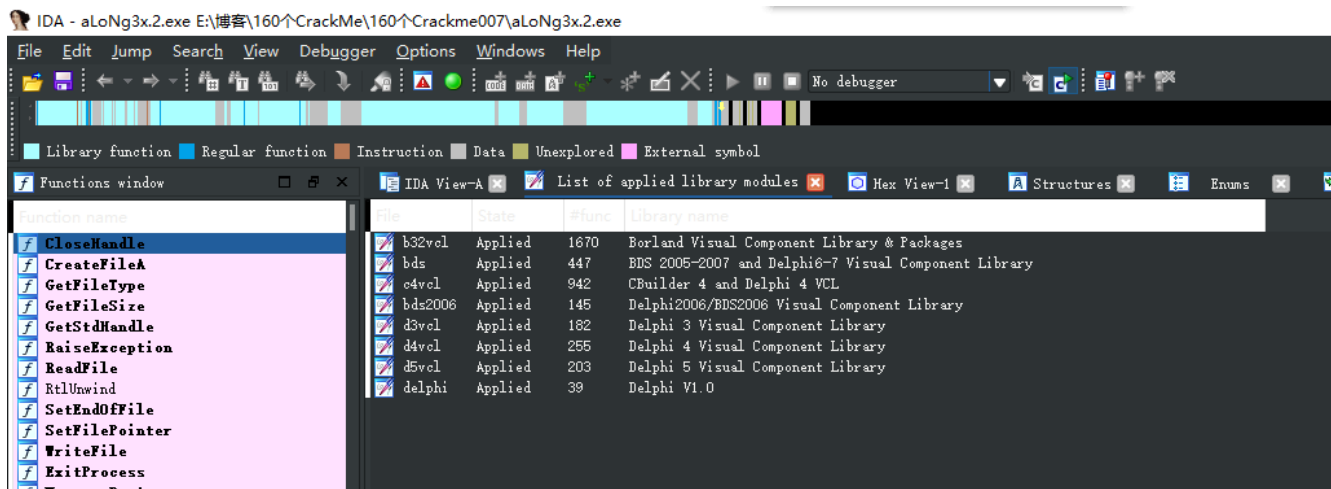


这里有以下几个事件:

- Cancella按钮的点击事件
- About按钮的点击事件
- Registerz按钮的点击事件
- Again按钮的点击事件

总共四个按钮事件。我们就从Registerz按钮的点击事件开始分析。

导入符号



接下来拖到IDA里，添加所有的Delphi签名。导出Map文件，导入到OD里，方便接下来的分析。

分析Register按钮事件

找到0x442F28的位置，下断点分析。

地址	HEX	数据	反汇编	注释
00442F48	. 8B83 DC020000		mov eax, dword ptr ds:[ebx+0x2DC]	
00442F4E	. E8 ED02FEFF		call <aLoNg3x_.TControl::GetText(void)>	获取密码
00442F53	. 8B45 F8		mov eax, [local.2]	
00442F56	. 8D55 FC		lea edx, [local.1]	
00442F59	. E8 FAF9FBFF		call <aLoNg3x_.System::linkproc Val	检测是否为纯数字
00442F5E	. 8BF0		mov esi, eax	
00442F60	. 837D FC 00		cmp [local.1], 0x0	
00442F64	. 74 37		je short <aLoNg3x_.loc_442F9D>	
00442F66	. B8 38304400		mov eax, aLoNg3x_.00443038	ASCII 59, "ou MUST insert a valid Lor
00442F6B	. E8 00F6FFFF		call <aLoNg3x_.Dialogs::ShowMessage(Sys	
00442F70	. 8D55 F8		lea edx, [local.2]	edx=密码
00442F73	. 8B83 DC020000		mov eax, dword ptr ds:[ebx+0x2DC]	
00442F79	. E8 C202FEFF		call <aLoNg3x_.TControl::GetText(void)>	
00442F7E	. 8B45 F8		mov eax, [local.2]	eax=密码
00442F81	. E8 06FBFFFF		call <aLoNg3x_.Libmain::TWindowDesigner	根据密码生成一个值
00443038=aLoNg3x_.00443038 (ASCII 59, "ou MUST insert a valid Long Integer Value in the Code Editor... Thank you :)")				

函数首先获取密码，然后检测密码是否为纯数字，不是则报错。但是我们必须让他报一次错。因为当密码不为纯数字的时候，他会根据密码生成一个值，这个值在后面必须要用到。如果密码为纯数字，则这个必须的值恒为零，注册永远不会成功。

地址	HEX	数据	反汇编	注释
00442F70	. 8D55 F8		lea edx, [local.2]	edx=密码
00442F73	. 8B83 DC020000		mov eax, dword ptr ds:[ebx+0x2DC]	
00442F79	. E8 C202FEFF		call <aLoNg3x_.TControl::GetText(void)>	
00442F7E	. 8B45 F8		mov eax, [local.2]	eax=密码
00442F81	. E8 06FBFFFF		call <aLoNg3x_.Libmain::TWindowDesigner	根据密码生成一个值
00442F86	. A3 30584400		mov dword ptr ds:[<dword_445830>], eax	
00442F8B	. BA 90304400		mov edx, aLoNg3x_.00443090	UNICODE "0"
00442F90	. 8B83 DC020000		mov eax, dword ptr ds:[ebx+0x2DC]	
00442F96	. E8 D502FEFF		call <aLoNg3x_.Controls::TControl::SetTe	
00442F9B	. EB 6F		jmp short <aLoNg3x_.loc_44300C>	
00442F9D	. 85F6		test esi, esi	loc_442F9D
00442F9F	. 7E 5A		jle short <aLoNg3x_.loc_442FFB>	
00442FA1	. 8D55 F8		lea edx, [local.2]	

接着如果输入的密码不是纯数字，那么则会根据输入的密码生成一个值，这个值在后面的核心算法会用到，至于是怎么算的我就不知道了。所以我将这个值固定一下，把第一次输入的密码固定为GuiShou，。

地址	HEX 数据	反汇编	注释
00442F9D	> 85F6	test esi,esi	loc_442F9D
00442F9F	. 7E 5A	jle short <aLoNg3x_.loc_442FFB>	
00442FA1	. 8D55 F8	lea edx,[local.2]	
00442FA4	. 8B83 D8020000	mov eax,dword ptr ds:[ebx+0x2D8]	
00442FAA	. E8 9102FEFF	call <aLoNg3x_.TControl::GetText(void)>	获取用户名
00442FAF	. 8B4D F8	mov ecx,[local.2]	ecx=username
00442FB2	. 8BD6	mov edx,esi	
00442FB4	. A1 30584400	mov eax,dword ptr ds:[<dword_445830>]	
00442FB9	. E8 EAF9FFFF	call <aLoNg3x_.sub_4429A8>	核心算法函数
00442FBE	. 84C0	test al,al	返回值为1则控件消失
00442FC0	. 74 30	jz short <aLoNg3x_.loc_442FF2>	
00442FC2	. 33D2	xor edx,edx	
00442FC4	. 8B83 CC020000	mov eax,dword ptr ds:[ebx+0x2CC]	
00442FCA	. E8 6101FEFF	call <aLoNg3x_.Controls::TControl::SetV	
00442FCE	. B2 01	mov dl,0x1	
al=00			

接下来获取用户名，然后有一个核心的算法，算法通过返回值为1则控件消失，否则不通过。接下来分析核心算法。

分析核心算法

地址	HEX 数据	反汇编	注释
004429CF	. 8B45 F8	mov eax,[local.2]	
004429D2	. E8 5D10FCFF	call <aLoNg3x_.__linkproc__ LStrLen>	获取用户名长度
004429D7	. 83F8 04	cmp eax,0x4	比较是否小于等于4
004429DA	. 0F8E 82000000	jle <aLoNg3x_.loc_442A62>	
004429E0	. 33DB	xor ebx,ebx	
004429E2	. 8B45 F8	mov eax,[local.2]	
004429E5	. E8 4A10FCFF	call <aLoNg3x_.__linkproc__ LStrLen>	再次获取用户名长度
004429EA	. 85C0	test eax,ebx	
004429EC	. 7E 38	jle short <aLoNg3x_.loc_442A26>	
004429EE	. 8945 F4	mov [local.3],eax	local3=usernameLen
004429F1	. BE 01000000	mov esi,0x1	esi=1
004429F6	> 8B45 F8	mov eax,[local.2]	eax=username
004429F9	. E8 3610FCFF	call <aLoNg3x_.__linkproc__ LStrLen>	再次获取用户名长度
004429FE	. 83F8 01	cmp eax,0x1	比较用户名长度是否为1
00442A01	. 7C 1D	jz short <aLoNg3x_.loc_442A20>	

首先获取用户名的长度，判断用户名长度是否小于等于4。是则报错。

地址	HEX 数据	反汇编	注释
00442A03	> 8B55 F8	mov edx,[local.2]	edx=username
00442A06	. 0FB65432 FF	movzx edx,byte ptr ds:[edx+esi-0x1]	edx=用户名的第i位
00442A0B	. 8B4D F8	mov ecx,[local.2]	ecx=username
00442A0E	. 0FB64C01 FF	movzx ecx,byte ptr ds:[ecx+eax-0x1]	ecx=用户名的最后一位(每次递减一位)
00442A13	. 0FAFD1	imul edx,ecx	edx=用户名的第一位乘以最后一位
00442A16	. 0FAFD7	imul edx,edi	edx=edx*edi (0xE9F)
00442A19	. 03DA	add ebx,edx	结果累加存放在ebx里
00442A1B	. 48	dec eax	用户名长度减一
00442A1C	. 85C0	test eax,ebx	检测用户名长度是否为零
00442A1E	. 75 E3	jnz short <aLoNg3x_.loc_442A03>	
00442A20	> 46	inc esi	esi++
00442A21	. FF4D F4	dec [local.3]	usernameLen--
00442A24	. 75 D0	jnz short <aLoNg3x_.loc_4429F6>	
00442A26	> 8BC3	mov eax,ebx	eax保存两轮循环结果
00442A28	. 99	cdb	
ebx=022848E4 eax=00000000			

接下来是个两层循环，计算的是用户名的第一位和最后一位的乘积，然后再乘以刚刚用用户名计算出来的被我固定的值。外层循环变换用户名最后一位，每次往前移动一位。内层循环变换用户名第一位，每次往后移动一位。接着将结果保存到eax。

地址	HEX 数据	反汇编	注释
00442A24	. ^ 75 D0	jmp short <aLoNg3x_.loc_4429F6>	
00442A26	> 8BC3	mov eax,ebx	eax保存两轮循环结果
00442A28	. 99	cdq	
00442A29	. 33C2	xor eax,edx	
00442A2B	. 2BC2	sub eax,edx	
00442A2D	. B9 2A2C0A00	mov ecx,0xA2C2A	ecx=0xA2C2A
00442A32	. 99	cdq	
00442A33	. F7F9	idiv ecx	
00442A35	. 8BDA	mov ebx,edx	ebx=eax%0xA2C2A
00442A37	. 8B45 FC	mov eax,[local.1]	eax=password
00442A3A	. B9 59000000	mov ecx,0x59	ecx=0x59
00442A3F	. 99	cdq	
00442A40	. F7F9	idiv ecx	
00442A42	. 8BC8	mov ecx,eax	ecx=password/0x59
00442A44	. 8B45 FC	mov eax,[local.1]	eax=password

地址	HEX 数据	反汇编	注释
00442A40	. F7F9	idiv ecx	
00442A42	. 8BC8	mov ecx,eax	ecx=password/0x59
00442A44	. 8B45 FC	mov eax,[local.1]	eax=password
00442A47	. BE 50000000	mov esi,0x50	esi=0x50
00442A4C	. 99	cdq	
00442A4D	. F7FE	idiv esi	eax=password/0x50
00442A4F	. 03CA	add ecx,edx	ecx=password/0x59+password%0x50
00442A51	. 41	inc ecx	ecx+1
00442A52	. 894D FC	mov [local.1],ecx	
00442A55	. 3B5D FC	cmp ebx,[local.1]	关键比较
00442A58	. 75 04	jmp short <aLoNg3x_.loc_442A5E>	
00442A5A	. B3 01	mov bl,0x1	
00442A5C	. EB 06	jmp short <aLoNg3x_.loc_442A64>	
00442A5E	> 33DB	xor ebx,ebx	loc_442A5E

接着将eax对0xA2C2A取模，记为结果1，然后将输入的密码除以0x59加上密码模以0x50再加1的值，记为结果2。然后比较结果1和结果2是否相等。相等则返回1，消失按钮。不相等则返回0。

写出注册机

为了防止最后计算的结果产生溢出，所以我将用户名固定为5位。代码如下。

```
int CalcKey1()
{
    int nKey = 0x1C48;
    char szName[10] = { 0 };
    int nCode = 0;
    int nTemp = 0;

    printf("请输入用户名:");
    scanf_s("%s", szName, 10);
    if (strlen(szName)!=5)
    {
        printf("请输入五位用户名\n");
        return 0;
    }

    //根据name字符串计算
    for (int i = 1; i <= 5; i++)
    {
        for (int j = 5; j >= 1; j--)
        {
            nTemp += szName[i - 1] * szName[j - 1] * nKey;
        }
    }
}
```

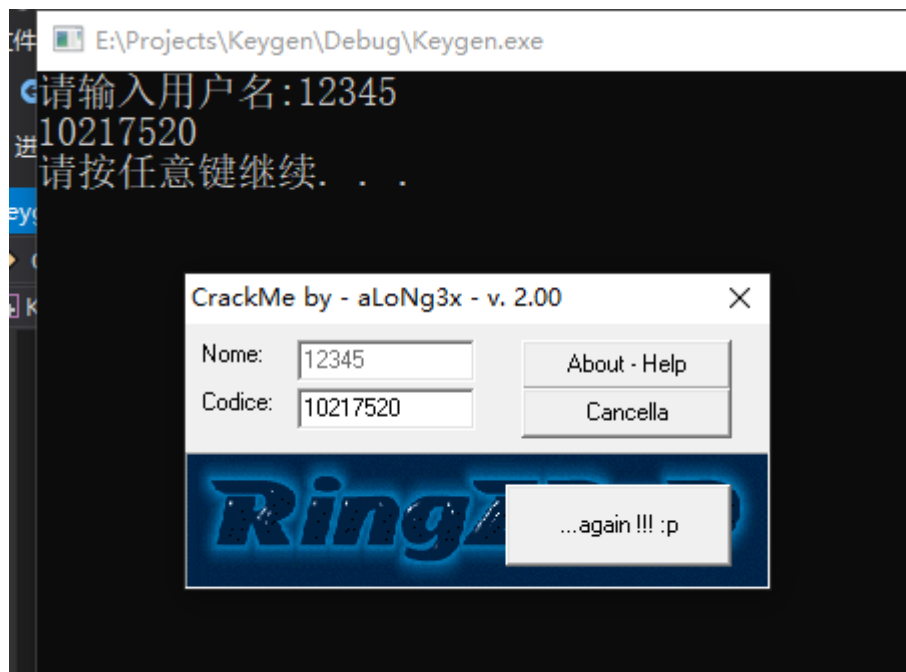
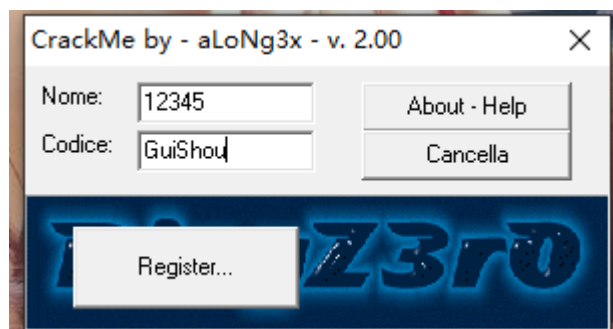
```

//取模
nTemp = nTemp % 0xA2C2A;
//反推code
nCode = (0x50 - ((nTemp - 1) * 0x59 % 0x50)) + (nTemp - 1) * 0x59;
printf("%d\n", nCode);
return 0;
}

```

校验结果

首先输入用户名为12345(不固定 但限制长度必须为5)，密码输入GuiShou(密码已固定)。然后点击，提示报错信息直接点击确定。



接着将用户名输入到注册机中，然后复制生成的序列号，点击注册，可以看到注册按钮消失。

分析again按钮事件

接下来来到0x4430BC这个位置，分析整个again按钮的点击事件。这个按钮事件跟Register完全一样。

地址	HEX 数据	反汇编	注释
004430E0	. E8 5B01FEFF	call <aLoNg3x_.TControl::GetText(void)>	获取密码
004430E5	. 8B45 F4	mov eax,[local.3]	
004430E8	. 8D55 FC	lea edx,[local.1]	
004430EB	. E8 68F8FBFF	call <aLoNg3x_.System::__linkproc__ ValLong(vo	检测是否为纯数字
004430F0	. 8BF0	mov esi,eax	
004430F2	. 837D FC 00	cmp [local.1],0x0	
004430F6	. 74 3A	je short <aLoNg3x_.loc_443132>	
004430F8	. B8 44324400	mov eax,aLoNg3x_.00443244	ASCII 59,"ou MUST insert a vali
004430FD	. E8 6EF4FFFF	call <aLoNg3x_.Dialogs::ShowMessage(System::An	
00443102	. 8D55 F4	lea edx,[local.3]	
00443105	. 8B83 DC020000	mov eax,dword ptr ds:[ebx+0x2DC]	
0044310B	. E8 3001FEFF	call <aLoNg3x_.TControl::GetText(void)>	获取密码
00443110	. 8B45 F4	mov eax,[local.3]	
00443113	. E8 74F9FFFF	call <aLoNg3x_.Libmain::TWindowDesigner::Selec	根据密码计算出来一个值->0x14C8
00443118	. A3 30584400	mov dword ptr ds:[<dword_445830>],eax	

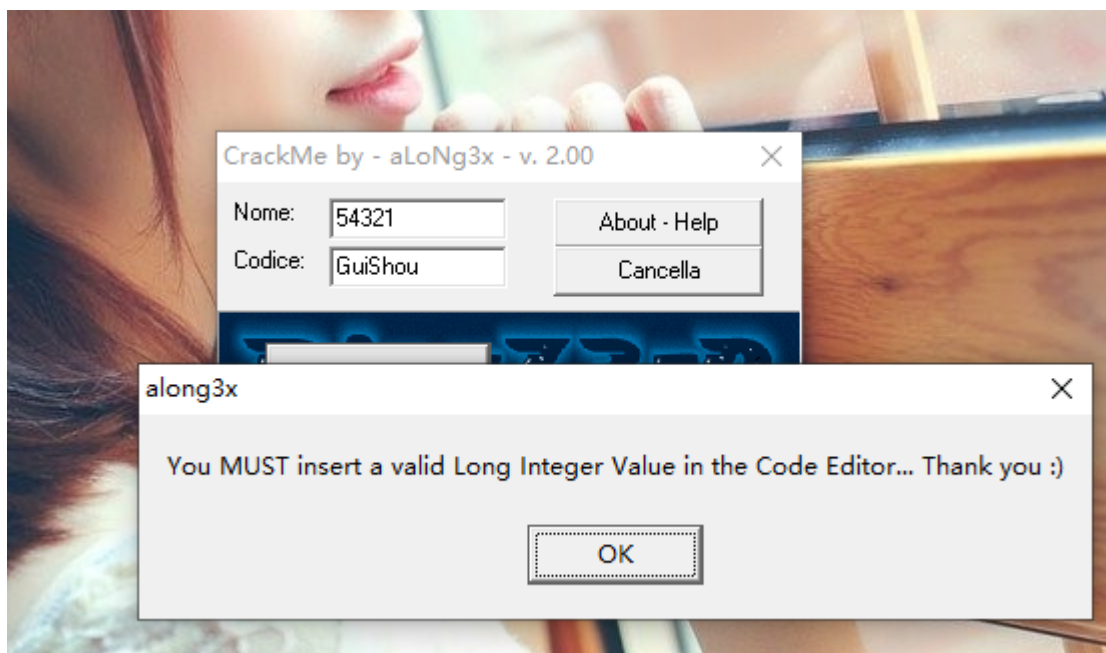
首先是获取密码，然后检测密码是否为纯数字，不是则报错，然后根据非纯数字的密码计算出一个值。

地址	HEX 数据	反汇编	注释
0044312D	. E9 DD000000	jmp <aLoNg3x_.loc_44320F>	
00443132	. 85F6	test esi,esi	esi=密码的十进制数
00443134	. 0F8E C4000000	jle <aLoNg3x_.loc_4431FE>	
0044313A	. 8D55 F4	lea edx,[local.3]	
0044313D	. 8B83 D8020000	mov eax,dword ptr ds:[ebx+0x2D8]	
00443143	. E8 F800FEFF	call <aLoNg3x_.TControl::GetText(void)>	获取密码的前五位
00443148	. 8B4D F4	mov ecx,[local.3]	ecx=获取密码的前五位
0044314B	. 8BD6	mov edx,esi	edx=密码的十进制数
0044314D	. A1 30584400	mov eax,dword ptr ds:[<dword_445830>]	eax=根据第一次输入的用户名算出来
00443152	. E8 51F8FFFF	call <aLoNg3x_.sub_4429A8>	核心算法函数
00443157	. 84C0	test al,al	
00443159	. 74 73	je short <aLoNg3x_.loc_4431CE>	
0044315B	. 33D2	xor edx,edx	

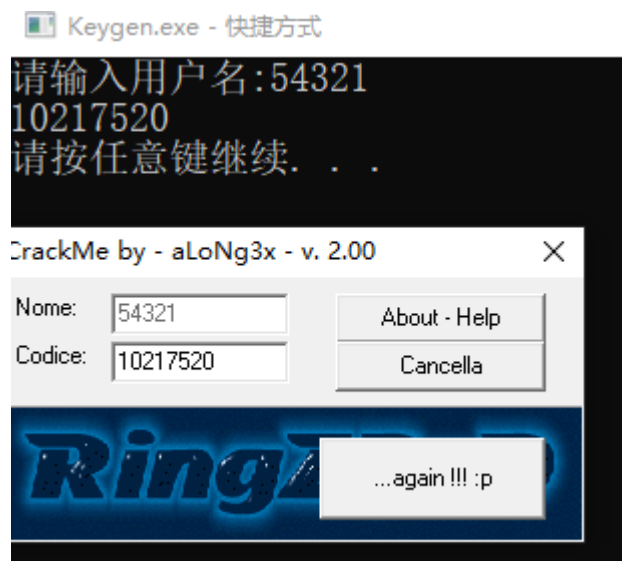
接着传入参数，又到了核心的算法函数，这个函数跟之前分析的一模一样。所以这个crackme到这里就算完全结束了。最后做一下总结。

校验步骤

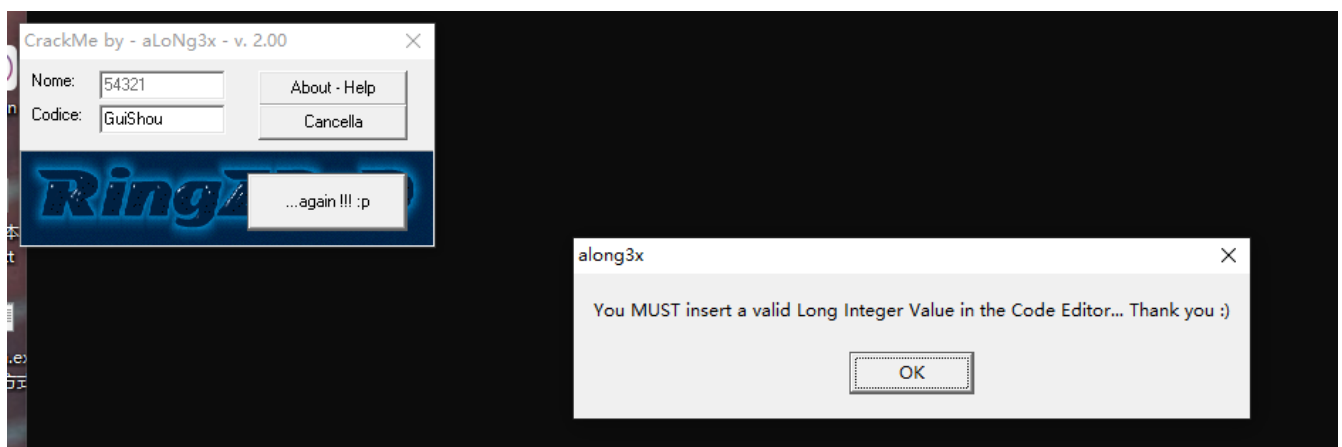
1. 首先输入用户名(内容随意，长度必须的五位数，为了防止最后的结果溢出)，然后输入固定的密码GuiShou，点击注册，弹框报错直接点击确定即可



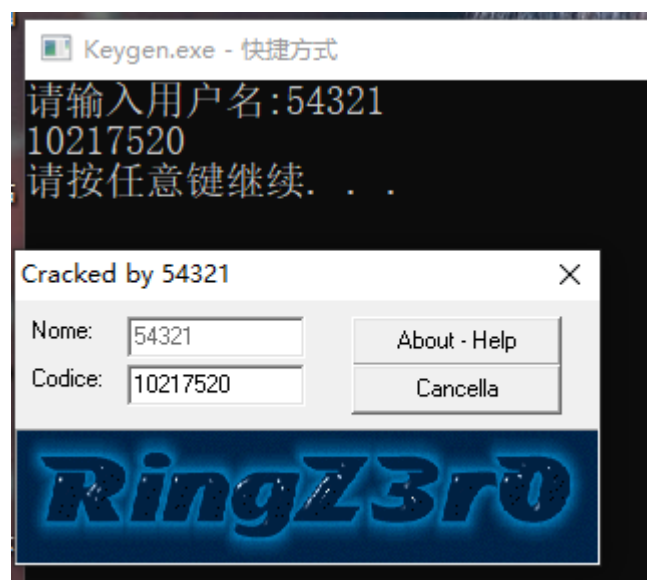
2. 接着将用户名输入到注册机，会算出一个序列号，再把这个序列号粘贴到Codice处，按钮消失。如图：



3. 接着再次再密码框输入GuiShou，报错后点击确定



4. 接着还是输入刚才注册机计算出来的序列号，点击again按钮，按钮消失，破解完成



需要相关文件的可以到我的Github下载: <https://github.com/TonyChen56/160-Crackme>