

## 校验步骤

条件一

条件二

条件三

条件四

条件五

注册机的编写

【软件名称】：Dope2112.2.exe

【软件大小】：12.0 KB

【下载地址】：自行搜索下载

【加壳方式】：无壳

【保护方式】：Keyfile

【编译语言】：MASM

【调试环境】：W7 32

【使用工具】：OD

【破解日期】：2019-6-15

【破解目的】：纯属兴趣

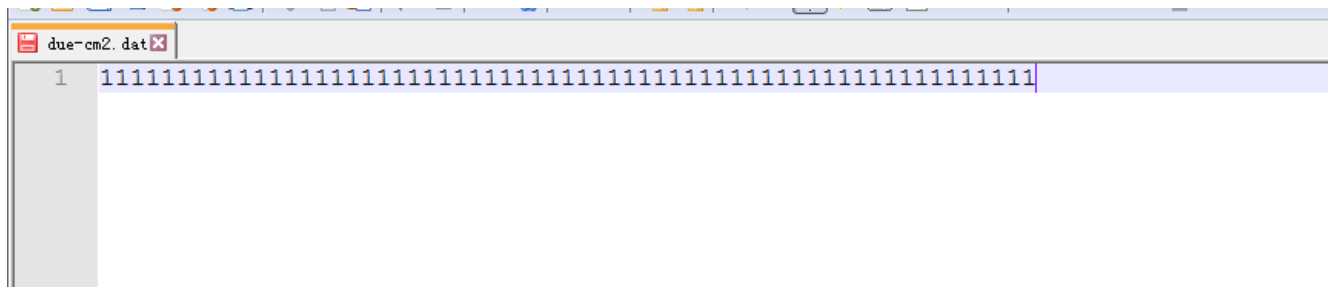
## 校验步骤

### 条件一

用OD载入后直接往下拉就能看到CreateFile函数，我们在这下个断点

00401067	. 6A 03	push 0x3	ShareMode = FILE_SHARE_READ FILE_SHARE_WRITE	EDI 00000000	
00401069	. 68 000000C0	push 0xC0000000	Access = GENERIC_READ GENERIC_WRITE	EIP 00401073 DueList_.00401073	
0040106E	. 68 79204000	push DueList_.00402079	FileName = "due-cm2.dat"	C 0 ES 0023 32位 0(FFFFFFFF)	
00401072	. E8 0B020000	call <jmp.&KERNEL32.CreateFileA>	CreateFileA	P 1 CS 001B 32位 0(FFFFFFFF)	
00401078	. 83F8 FF	cmp eax,-0x1		A 0 SS 0023 32位 0(FFFFFFFF)	
0040107B	. 75 1D	jnz short DueList_.0040109A		Z 1 DS 0023 32位 0(FFFFFFFF)	
0040107D	. 6A 00	push 0x0	Style = MB_OK MB_APPLMODAL	S 0 FS 003B 32位 7FFDE000(FFF)	
0040107F	. 68 01204000	push DueList_.00402001	Title = "Duelist's Crackme #2"	T 0 GS 0000 NULL	
00401084	. 68 17204000	push DueList_.00402017	Text = "Your time-trial has ended... Please register	D 0	
00401089	. 6A 00	push 0x0	hOwner = NULL	O 0 LastErr ERROR_SUCCESS (00000000)	
0040108B	. E8 D7020000	call <jmp.&USER32.MessageBoxA>	MessageBoxA	EFL 00000246 (NO,NB,E,BE,NS,PE,GE,LE)	
00401090	. E8 24020000	call <jmp.&KERNEL32.ExitProcess>	ExitProcess	ST0 empty 0.0	
00401095	. E9 28010000	jmp DueList_.004011C2		ST1 empty 0.0	
0040109A	. 6A 00	push 0x0	pOverlapped = NULL	ST2 empty 0.0	
00401283=<jmp.&KERNEL32.CreateFileA>					
地址	HEX 数据	ASCII	地址	数值 注释	
00402000	20 44 75 65 6C 69 73 74 27 73 20 43 72 61 63 68	Duelist's Crack	0012FF70	00402079	FileName = "due-cm2.dat"
00402010	6D 65 20 23 32 00 20 59 6F 75 72 20 74 69 6D 65	me #2. Your time	0012FF74	C0000000	Access = GENERIC_READ GENERIC_WRITE
00402020	2D 74 72 69 61 6C 20 68 61 73 20 65 6E 64 65 64	-trial has ended	0012FF78	00000003	ShareMode = FILE_SHARE_READ FILE_SHARE_WRITE
00402030	2E 2E 2E 20 50 6C 65 61 73 65 20 72 65 67 69 73	... Please regis	0012FF7C	00000000	pSecurity = NULL
00402040	74 65 72 20 61 6E 64 20 63 6F 70 79 20 74 68 65	ter and copy the	0012FF80	00000003	Mode = OPEN_EXISTING
00402050	20 6B 65 79 66 69 6C 65 20 73 65 6E 74 20 74 6F	keyfile sent to	0012FF84	0040216F	Attributes = READONLY HIDDEN SYSTEM ARCHIVE TEMPORARY 402048
00402060	20 79 6F 75 20 74 6F 20 74 68 69 73 20 64 69 72	you to this dir	0012FF88	00000000	hTemplateFile = NULL

该函数以读写的方式打开due-cm2.dat文件，为了通过校验，我们需要在同路径下创建一个名为due-cm2.dat的文件，并在文件内填充任意内容



接着读取0x46个字节的文件内容到缓冲区

004010A8	. 50	push eax	hFile = 0000006C (window)	D 0
004010A9	. E8 2F020000	call <jmp.&KERNEL32.ReadFile>	ReadFile	0 0 LastErr ERROR_SUCCESS (00000000)
004010AE	. 85C0	test eax, eax		EFL 00000213 (NO, B, NE, BE, NS, PO, C)
004010B0	. 75 02	jnz short DueList_.004010B4		ST0 empty 0.0
004010B2	. EB 43	jmp short DueList_.004010F7		ST1 empty 0.0
004012D0=<jmp.&KERNEL32.ReadFile>				
地址	HEX 数据	ASCII	地址	数值 注释
00402173	00 00 00 00 00 00 40 00 00 00 00 00 00 00 00 00	.....@.....	0012FF78	0000006C hFile = 0000006C (window)
00402183	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....	0012FF7C	0040211A Buffer = DueList_.0040211A
00402193	00 00 00 00 03 40 00 00 A6 11 40 00 00 00 00 00	...@...?@...	0012FF80	00000046 BytesToRead = 46 (70...)
004021A3	00 00 00 00 00 00 40 00 D1 07 1C 00 03 00 01 00	...@...?..	0012FF84	00402173 pBytesRead = DueList_.00402173
004021B3	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....	0012FF88	00000000 pOverlapped = NULL
004021C3	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....	0012FF8C	75503C45 返回到 kernel32.75503C45
004021D3	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....	0012FF90	7FFDF000
004021E3	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....	0012FF94	0012FFD4
004021F3	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....	0012FF98	76F137F5 返回到 ntdll.76F137F5
00402203	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....	0012FF9C	7FFDF000
00402213	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....	0012FFA0	76B3C073 shell132.76B3C073
00402223	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....	0012FFA4	00000000
00402233	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....	0012FFA8	00000000
00402243	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....	0012FFAC	7FFDF000

这里重点关注pBytesRead实际读取到的字节数

004010A8	. 50	push eax	hFile = 00000001
004010A9	. E8 2F020000	call <jmp.&KERNEL32.ReadFile>	ReadFile
004010AE	. 85C0	test eax, eax	
004010B0	. 75 02	jnz short DueList_.004010B4	
004010B2	. EB 43	jmp short DueList_.004010F7	
004010B4	. 33DB	xor ebx, ebx	
004010B6	. 33F6	xor esi, esi	
004010B8	. 833D 73214000	cmp dword ptr ds:[0x402173], 0x12	
004010BF	. 7C 36	jl short DueList_.004010F7	
004010C1	. 8A83 1A214000	mov al, byte ptr ds:[ebx+0x40211A]	
004010C7	. 3C 00	cmp al, 0x0	
004010C9	. 74 08	je short DueList_.004010D3	
004010CB	. 3C 01	cmp al, 0x1	
ds:[00402173]=0000003D			
地址	HEX 数据	ASCII	
00402173	3D 00 00 00 00 00 40 00 00 00 00 00 00 00 00 00	.....@.....	
00402183	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....	

然后会比较[0x402173]的位置的值是否小于0x12，这个地址就是实际读取的字节数，所以我们得出

条件1: due-cm2.dat文件内容不能小于0x12个字节

条件二

004010BF	< 7C 30	jl short DueList_.004010F7	
004010C1	> 8A83 1A214000	mov al,byte ptr ds:[ebx+0x40211A]	循环读取文件内容到al
004010C7	< 3C 00	cmp al,0x0	比较al是否为0 这里是在判断是否到了文件末尾
004010C9	< 74 08	je short DueList_.004010D3	
004010CB	< 3C 01	cmp al,0x1	比较al是否为1
004010CD	< 75 01	jnz short DueList_.004010D0	
004010CF	< 46	inc esi	文件内容为1 则esi++
004010D0	< 43	inc ebx	ebx++
004010D1	< EB EE	jmp short DueList_.004010C1	跳转到循环开始处
004010D3	< 83FE 02	cmp esi,0x2	
004010D6	< 7C 1F	jl short DueList_.004010F7	跳转提示错误
004010D8	< 33F6	xor esi,esi	
004010DA	< 33DB	xor ebx,ebx	
004010DC	> 8A83 1A214000	mov al,byte ptr ds:[ebx+0x40211A]	
004010E2	< 3C 00	cmp al,0x0	
跳转未实现 004010F7=DueList_.004010F7			
地址	HEX 数据	ASCII	地址 数值 注释
0040211A	31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31	1111111111111111	0012FF8C 75503C45 返回到
0040212A	31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31	1111111111111111	0012FF90 7FFDF000
0040213A	31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31	1111111111111111	0012FF94 0012FFD4
0040214A	31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31	1111111111111111	0012FF98 76F137F5 返回到
0040215A	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....	0012FF9C 7FFDF000
0040216A	00 00 00 00 00 80 00 00 00 3D 00 00 00 00 00 40	.....e...=....@	0012FFA0 76B3C073 shell:
0040217A	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....	0012FFA4 00000000
0040218A	00 00 00 00 00 00 00 00 00 00 00 00 00 03 40 00	.....L@.	0012FFA8 00000000
0040219A	00 A6 11 40 00 00 00 00 00 00 00 00 00 00 00 40	.?@.....@	0012FFAC 7FFDF000
004021AA	00 D1 07 1C 00 03 00 01 00 00 00 00 00 00 00 00	.?.L.....	0012FFB0 00000000
004021BA	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....	0012FFB4 00000000
004021CA	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....	0012FFB8 00000000
004021DA	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....	0012FFBC 0012FFA0 ASCII
004021EA	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....	0012FFC0 00000000
004021FA	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....	0012FFC4 FFFFFFFF SEH 链
0040220A	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....	0012FFC8 76ECE0ED SE处理
0040221A	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....	0012FFCC 00512DDF
0040222A	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....	0012FFD0 00000000
0040223A	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....	0012FFD4 0012FFEC

接着到了一个循环，这个循环会读取文件的每一个字节，并且和1比较，如果内容为1，则esi++。然后再循环结束之后，会比较esi是否小于2。所以我们得出：

条件二：文件内容的ASCII必须有至少两个0x1

### 条件三

继续往下，我们暂时先将esi的值修改为2

寄存器 (FPU)
EAX 00000000
ECX 754F9754 ke
EDX 76EF70B4 nt
EBX 0000003D
ESP 0012FF8C
EBP 0012FF94
ESI 00000002
EDI 00000000

004010D8	. 33F6	xor esi,esi	esi清零
004010DA	. 33DB	xor ebx,ebx	ebx清零
004010DC	> 8A83 1A214000	mov al,byte ptr ds:[ebx+0x40211A]	循环读取文件内容到al
004010E2	. 3C 00	cmp al,0x0	比较al是否为0 这里是在判断是否到了文件末尾
004010E4	. 74 09	je short DueList_.004010EF	跳转到循环结束
004010E6	. 3C 01	cmp al,0x1	比较al是否为1
004010E8	. 74 05	je short DueList_.004010EF	
004010EA	. 03F0	add esi,eax	将文件内容的ASCII值和esi相加
004010EC	. 43	inc ebx	ebx++
004010ED	. EB ED	jmp short DueList_.004010DC	
004010EF	> 81FE D5010000	cmp esi,0x1D5	如果al为1则比较esi是否为0x1D5
004010F5	. 74 1D	je short DueList_.00401114	如果相等则跳过错误提示
004010F7	> 6A 00	push 0x0	Style = MB_OK MB_APPLMODAL
004010F9	. 68 01204000	push DueList_.00402001	Title = "Duelist's Crackme #2"
004010FE	. 68 86204000	push DueList_.00402086	Text = "Your current keyfile is invalid... Please obt
00401103	. 6A 00	push 0x0	hOwner = NULL
00401105	. E8 5D020000	call <jmp.&USER32.MessageBoxA>	MessageBoxA
0040110A	. E8 AA010000	call <jmp.&KERNEL32.ExitProcess>	ExitProcess
0040110F	. E9 AE000000	jmp DueList_.004011C2	
00401114	> 33F6	xor esi,esi	
00401116	. 43	inc ebx	

接着又是一轮循环，这一次将上轮循环的esi和ebx清零之后，依旧是循环读取文件内容到al，如果al的值为1则比较esi是否为0x1D5，接着会根据比较的结果决定是否跳转。所以我们得出：

**条件三：在第一个0x1之前的文件内容的ASCII值之和必须为0x1D5**

## 条件四

继续跟踪，这里先暂时通过修改零标志位让程序通过条件三的验证

0040110F	. E9 AE000000	jmp DueList_.004011C2	
00401114	> 33F6	xor esi,esi	esi清零
00401116	. 43	inc ebx	ebx++
00401117	. 8A83 1A214000	mov al,byte ptr ds:[ebx+0x40211A]	循环读取文件内容到al
0040111D	. 3C 00	cmp al,0x0	比较al是否为0 这里是在判断是否到了文件末尾
0040111F	. 74 18	je short DueList_.00401139	跳转到循环结束
00401121	. 3C 01	cmp al,0x1	比较al是否为1
00401123	. 74 14	je short DueList_.00401139	
00401125	. 83FE 0F	cmp esi,0xF	if (esi==0xF)
00401128	. 73 0F	jnb short DueList_.00401139	不小于则跳转到循环结束位置
0040112A	. 3286 1A214000	xor al,byte ptr ds:[esi+0x40211A]	进行异或
00401130	. 8986 60214000	mov dword ptr ds:[esi+0x402160],eax	保存异或结果
00401136	. 46	inc esi	esi++
00401137	. EB DD	jmp short DueList_.00401116	跳到循环开始处

这里还是循环读取文件内容，不同的是这一次ebx的值没有被清零，ebx的值是条件三的循环次数。这里将进行循环异或然后将异或的结果保存，保存的结果最终将显示为成功注册的用户名

## 条件五

地址	HEX 数据	反汇编	注释
0040113A	. 33F6	xor esi,esi	esi清零
0040113C	> 8A83 1A214000	mov al,byte ptr ds:[ebx+0x40211A]	循环读取文件内容到al
00401142	. 3C 00	cmp al,0x0	比较al是否为0 这里是在判断是否到了文件末尾
00401144	. 74 09	je short DueList_.0040114F	跳转到循环结束
00401146	. 3C 01	cmp al,0x1	比较al是否为1
00401148	. 74 F2	je short DueList_.0040113C	如果al为1则跳过此次循环
0040114A	. 03F0	add esi,eax	将文件内容和esi相加
0040114C	. 43	inc ebx	ebx++
0040114D	. EB ED	jmp short DueList_.0040113C	
0040114F	> 81FE B2010000	cmp esi,0x1B2	
00401155	. 75 A0	jnz short DueList_.004010F7	
00401157	. 6A 00	push 0x0	lParam = NULL
00401159	. 68 C9114000	push DueList_.004011C9	DlgProc = DueList_.004011C9
0040115E	. 6A 00	push 0x0	hOwner = NULL
00401160	. 6A 05	push 0x5	pTemplate = 0x5
00401162	. FF35 77214000	push dword ptr ds:[0x402177]	hInst = 00400000
00401168	. E8 42020000	call <jmp.&USER32.DialogBoxParamA>	DialogBoxParamA
0040116D	. EB 53	jmp short DueList_.004011C2	
0040116F	> 6A 00	push 0x0	MsgFilterMax = 0x0
00401171	. 6A 00	push 0x0	MsgFilterMin = 0x0
00401173	. 6A 00	push 0x0	hWnd = NULL

这里同样没有把ebx清零，也是读取文件内容，将文件内容的ASCII值累加到esi，如果遇到文件内容为1则跳过循环。最后循环结束之后比较esi的值是否为0x1B2，如果相等则校验通过，所以我们得出

条件四：文件内容的ASCII值(除去0x1)累加必须等于0x1B2

## 注册机的编写

看到这么繁琐的校验我就脑阔疼。这个程序分析出了注册条件还得自己去推注册机！直接从吾爱拷了一个过来。。。代码如下：

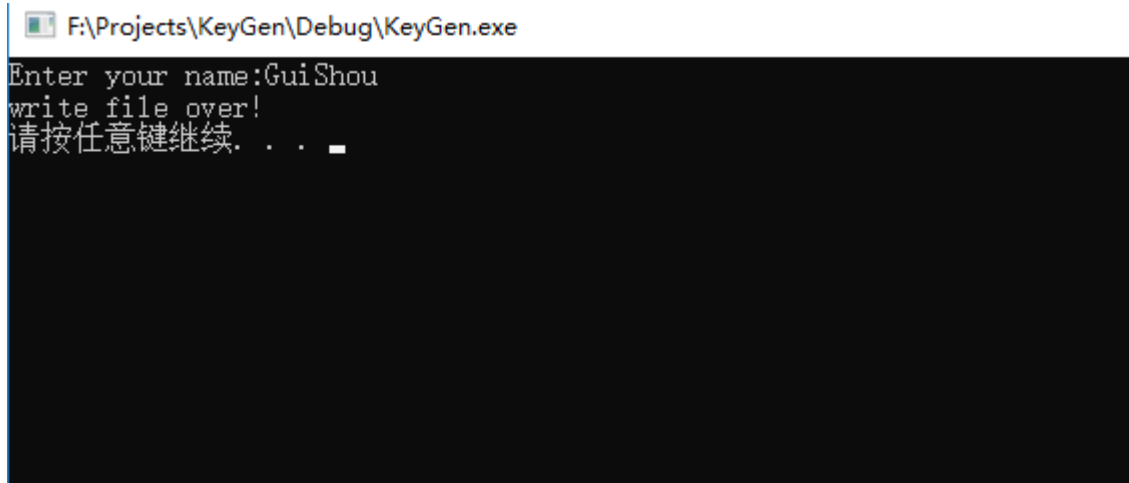
```
#define _CRT_SECURE_NO_WARNINGS
#include<iostream>
#include <windows.h>
using namespace std;
int main()
{
    const char * path = "E:\\\\due-cm2.dat";
    FILE * keyfile;
    char * youkey = new char[40];           //输入用户名
    memset(youkey, 0, 40);
    unsigned char * writekey = new unsigned char[40];
    memset(writekey, 0, 40);               //存入文件的内容
    cout << "Enter your name:";
    scanf_s("%s",youkey,40);
    if (strlen(youkey) >= 8)
        cout << "用户名最多支持13位，其余部分将被截断!" << endl;
    int now = 0;

    writekey[0] = 0xEA;
    writekey[1] = 0xEB;
    writekey[2] = 0x01;
    for (unsigned int x = 0; x < 13 && x < strlen(youkey); x++)
    {
        writekey[x + 3] = youkey[x] ^ writekey[x];
    }
    if (strlen(youkey) >= 13)
        now = 16;
    else
        now = strlen(youkey) + 3;

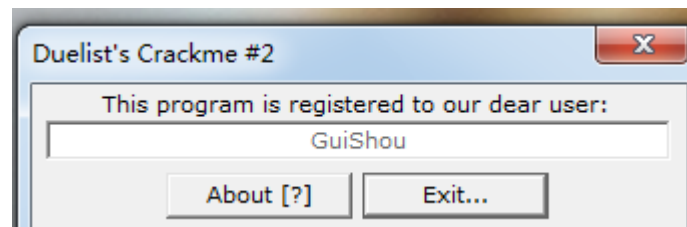
    writekey[now] = 0x01;
    writekey[now + 1] = 0xD9;
    writekey[now + 2] = 0xD9;
    errno_t err = fopen_s(&keyfile, path, "w+b");
    if (err != 0)
    {
        cout << "file open or create failed!" << endl;
        system("pause");
        return -1;
    }
    rewind(keyfile);
    fwrite(writekey, sizeof(byte), 40, keyfile);
    fclose(keyfile);
    cout << "write file over!" << endl;
    delete[] youkey;
```

```
delete[] writekey;  
system("pause");  
return 1;  
}
```

设置好文件路径，然后输入用户名



就会将用户名和密码写入到文件，然后再次打开程序



显示注册成功 KO!

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