# CrackMe破解



## 前置知识介绍：

* 160 CrackMe 是比较适合新手学习逆向破解的CrackMe的一个集合，一共160个待逆向破解的程序
* CrackMe:一些公开给别人尝试破解的小程序，制作crackme的人可能是程序员，想测试一下自己的软件保护技术，也可能是一位cracker,想挑战一下其它cracker的破解实力，也可能是一些正在学习破解的人，自己编一些小程序给自己破。
* CrackMe简称CM，cracker是以破解各种加密或有限制的商业软件为乐趣的人。

# CrackMe001

## 程序信息：

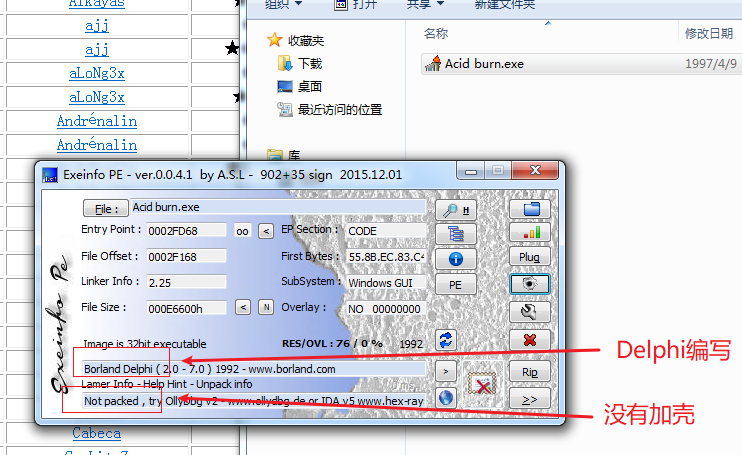
| **编号** | **作者** | **保护方式** |
| --- | --- | --- |
| 001 | Acid burn | Name/Serial,Serial |

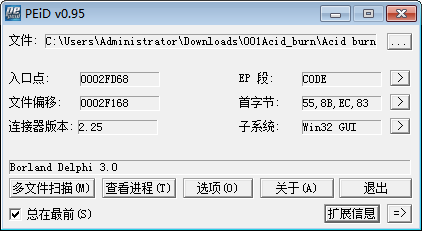
## 工具：

* ollyDbg（简称OD）
* Exeinfo PE/PEiD

## 开始破解：

1.先用ExeinfoPE/PEiD工具查看下001程序（打开工具，将程序拖拽进去即可查看），得到信息如下图，可知该程序没有加壳，而且是用Delphi语言编写的。

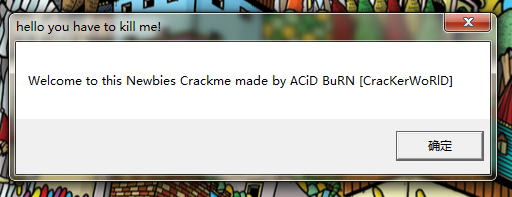


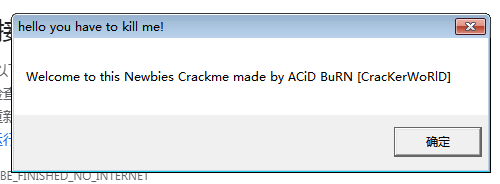


2.Delphi所编写的代码与VC++所编写的代码有所不同，最明显的两点区别如下：

* 函数调用时传参的方式主要用寄存器，一般将参数1放入eax寄存器，参数2放入edx寄存器，参数3放入ecx寄存器，多余的也是通过压栈的方式传入
* 字符串的处理，Delphi字符串处理函数与VC++是不同的，其字符串的首地址减去4后，所取出的4个字节就是此字符串的长度。分析的时候要注意。

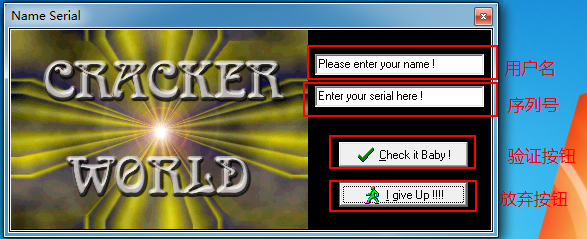
3.接下来我们运行程序，首先会弹出一个提示窗口，大概意思是欢迎你破解他，这个软件由ACiD BuRN制作





4.现在我们知道这个软件制作者是谁呢，不想每次运行都弹出这个对话框来提醒我制作者是谁，那么我们有了第一个目标，让提示窗口不显示。点击《确定》按钮，发现有两种验证机制，**一种是用户名与序列号验证，另一种只需要序列号**。所以我们又获得**两个新任务**，由易到难，我们第二个目标是只需要序列号验证的，最后攻克用户名与序列号验证。





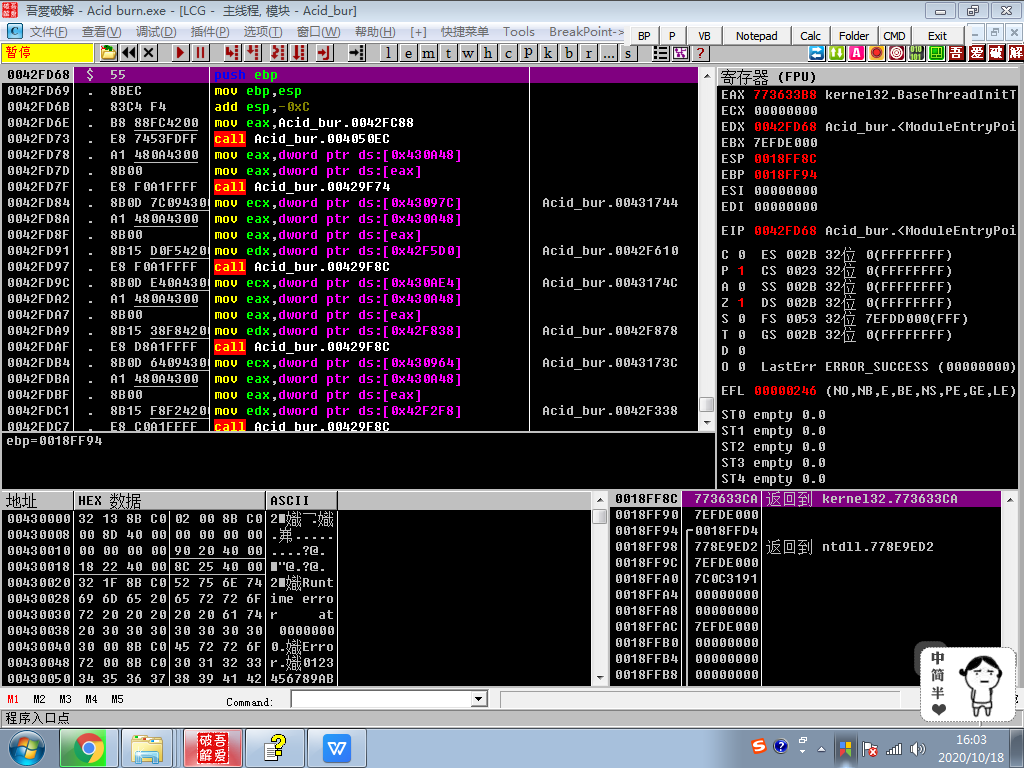


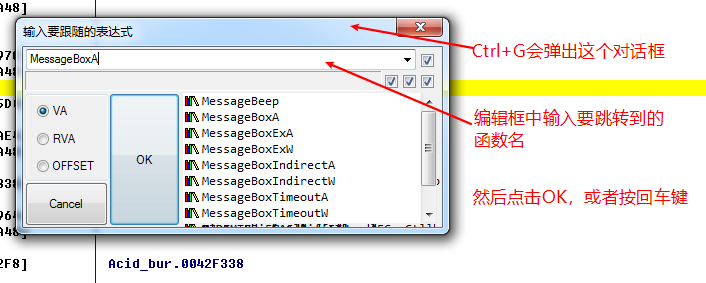
## 目标一：去掉烦人的提示对话框（常用函数下断法）

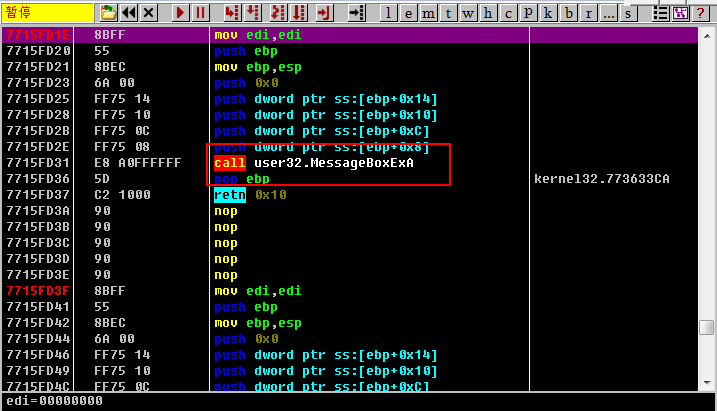
1.目标一：去掉烦人的提示对话框（一般软件设计者用来提醒用户购买正式版本的窗口），要完全去除提示窗口，只需要找到创建该窗口的代码并将其跳过。显示窗口的常用函数有：

* MessageBoxA(W)
* MessageBoxExA(W)
* DialogBoxParamA(W)
* ShowWindow
* CreateWindowExA(W)

2.我们用OD打开,然后用Ctrl+G跳转到上面常用函数处，然后按F2键下断点



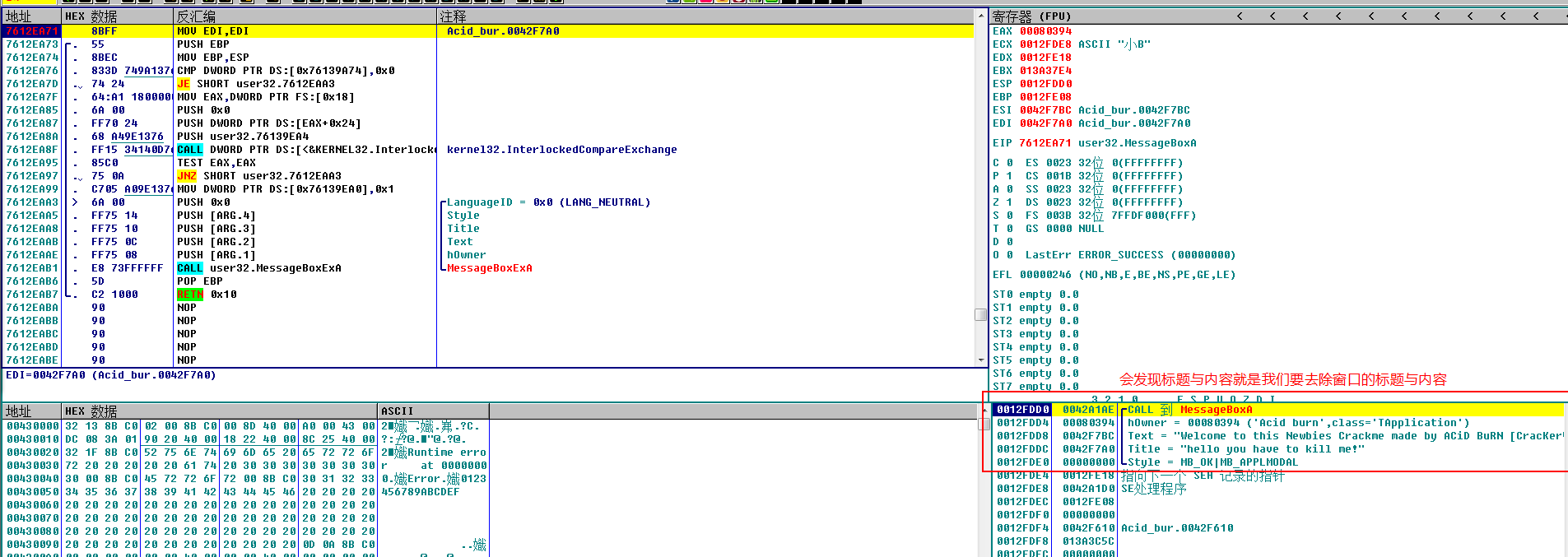


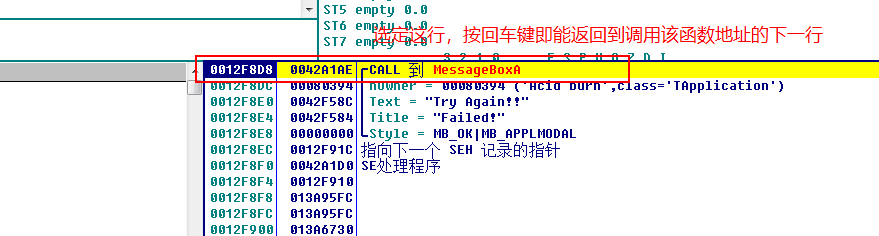


3.或者用命令bp MessageBoxA在指定函数处下断，bp表示下断点，后面跟着要下断点的函数名，（bp与函数名之间有个空格）



4.下完断点后，按F9让程序运行起来，发现程序会在函数MessageBoxA处断下，观察栈区，会发现就是我们要去除的窗口信息，即此时MessageBoxA函数执行完，就会弹出提示窗口，我们转到调用这个函数的位置（方法见下图2）

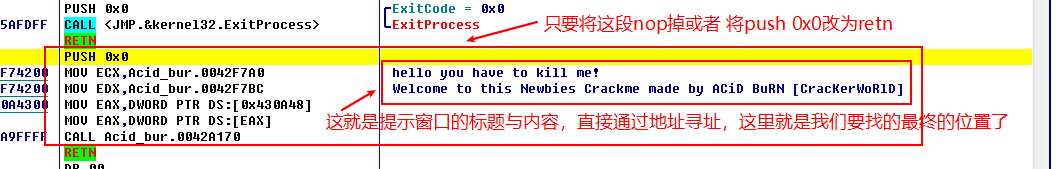




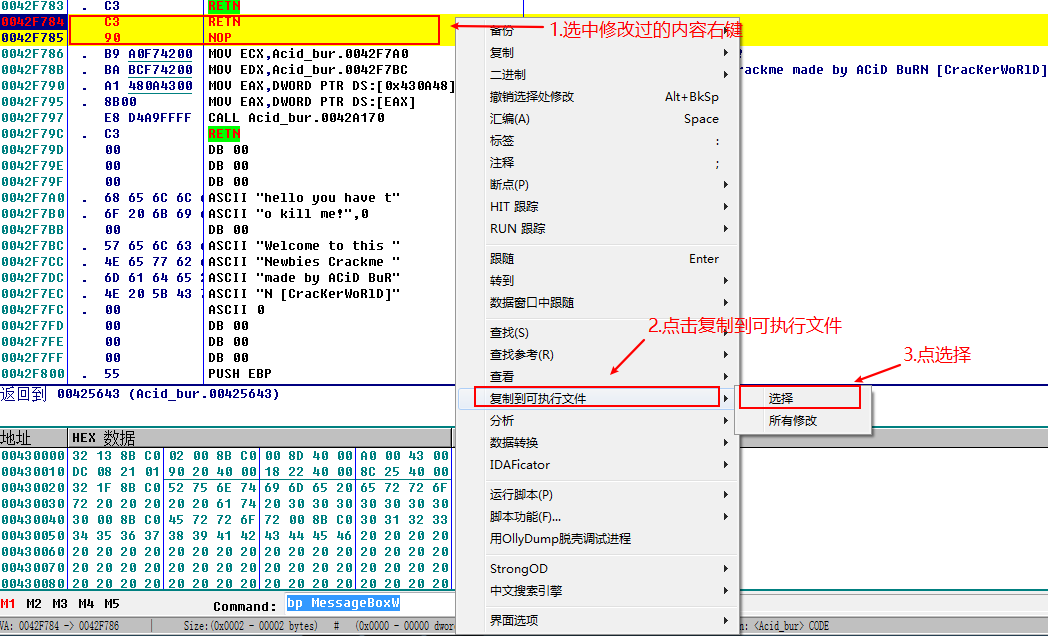
5.本来我们只要nop掉调用MessageBoxA的代码就可以实现目标一，不过后面验证后的结果提示窗口也会调用此处，所以我们还要继续向上找

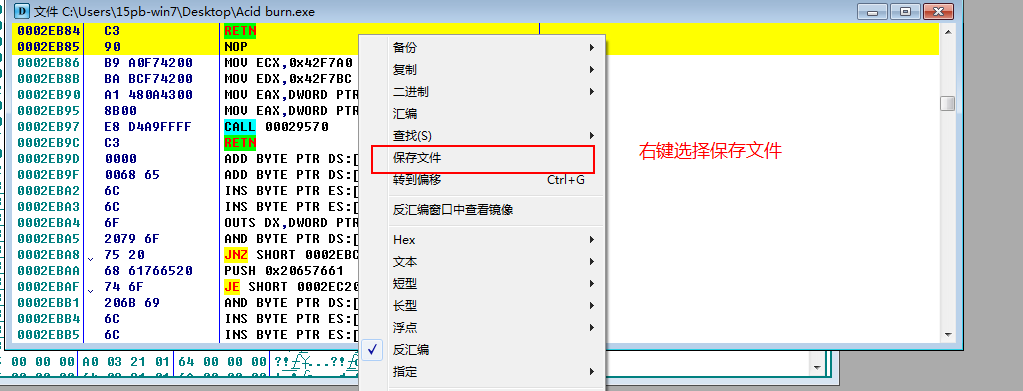


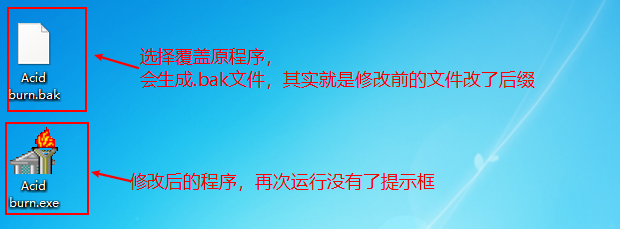
6.终于来到最终的关键处，此时我们采用第二种，直接将push 0x0改为retn(有兴趣的两种都可以试试)



7.直接选择要修改的那行，按空格键，输入改后的指令，然后保存修改好后的程序，选择覆盖原程序，会在原程序目录下生成个.bak文件，其实就是修改前的文件改了后缀。





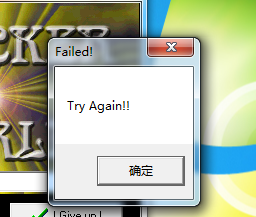


## 目标二：只需要序列号验证的（字符串定位法）

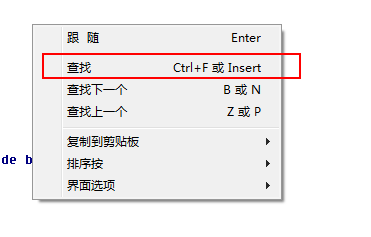
1.第二个目标，只需要序列号的验证机制，打开上篇完成修改后的程序，没有了烦人的提示窗口，是不是心情也愉快的很多。接下来我们点击Serial按钮来到序列号验证窗口，随意输入一个序列号，然后点击验证按钮。



2.显然我们随意输入的序列号是正确的可能性是极低的，一般友好的程序会在你验证过后给用户一个提示，告诉用户序列号是否正确，果然，程序以窗口的形式给了用户提示，又是窗口，如果你还记得昨天的方法，你应该可以找到创建该窗口的关键代码（不妨试试）

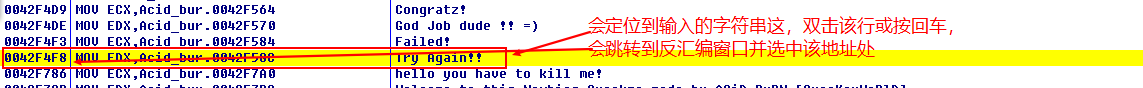


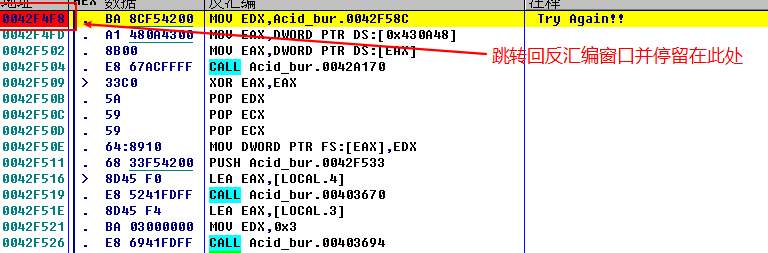
3.今天我们用一种新的方法，**字符串定位法**，上面验证错误后，会弹出一个窗口，显示Try Again(再来一次)，我们用OD打开程序，右键->中文搜索引擎->智能搜索（这里需要插件，会在文末给大家），在弹出的界面中右键->查找，输入刚刚提示的信息Try Again,然后回车



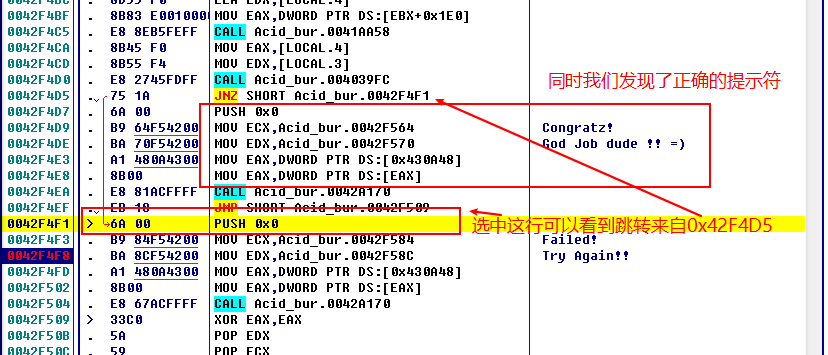


4.现在就会定位到输入的字符串那行（如果存在），按N或B键查找下一个，发现一共有三个，还行（如果太多，这个方法就不好用了），我们在三处都下断点（双击或按回车就会跳回到反汇编窗口并停留在选中的地址处，然后**按F2键下断点**），然后按程序运行，输入一个序列号，进行验证

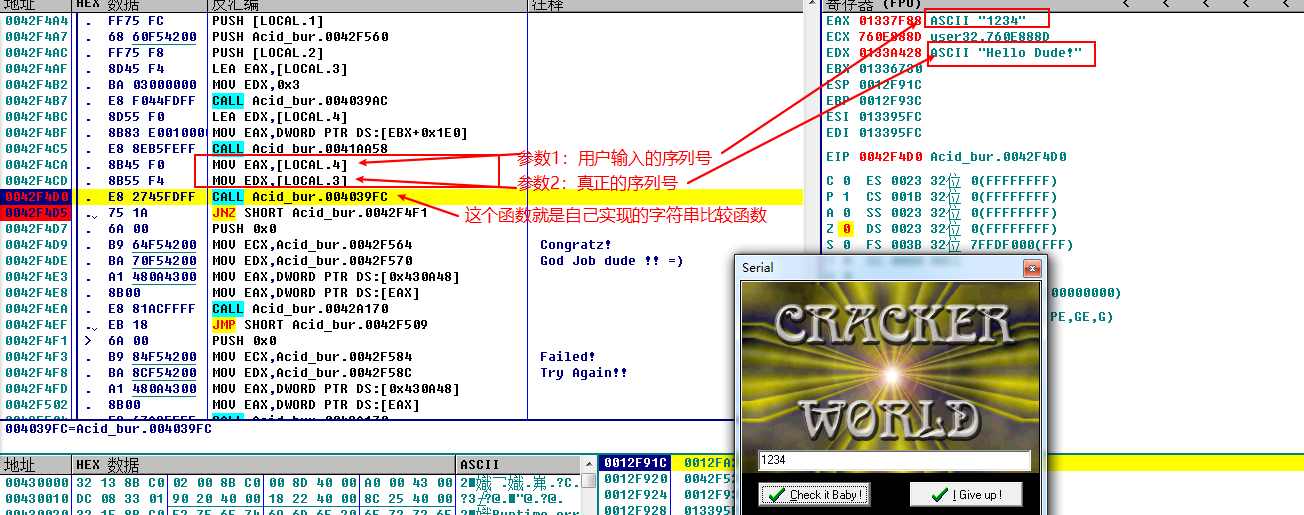




5.会发现程序停在0x42F4F8处，我们向上看，很容易就发现了**关键跳转**，同时发现了序列号正确的提示信息，所以**只要0x42F4D5处的跳转指令不执行，验证就一定会通过**，我们可以**将jnz改为jmp,然后保存修改后的内容就行了**，这样无论你输入什么都能正确（这就是爆破，可以参考上一篇试试）。



6.但是我们今天要找到正确的序列号，jnz指令的上一条指令是call一个函数，也就是当函数返回值是1就会实现跳转，我们在call指令所在地址0x42F4D0下断点，然后F9运行程序，再次输入一个序列号（1234，方便观察），点击验证按钮，会断在0x42F4D0处，同时我们会发现**函数两个参数一个是我们输入的序列号，另一个也是一串字符串**，所以这个函数的功能可能是**比较两个字符串，相同就返回0，不相同就返回1**。经过测试，的确如此。那么Hello Dude!就是真正的序列号



7.我们重新运行程序，输入序列号Hello Dude!，验证通过。

