VProtect 共享软件加密授权系统

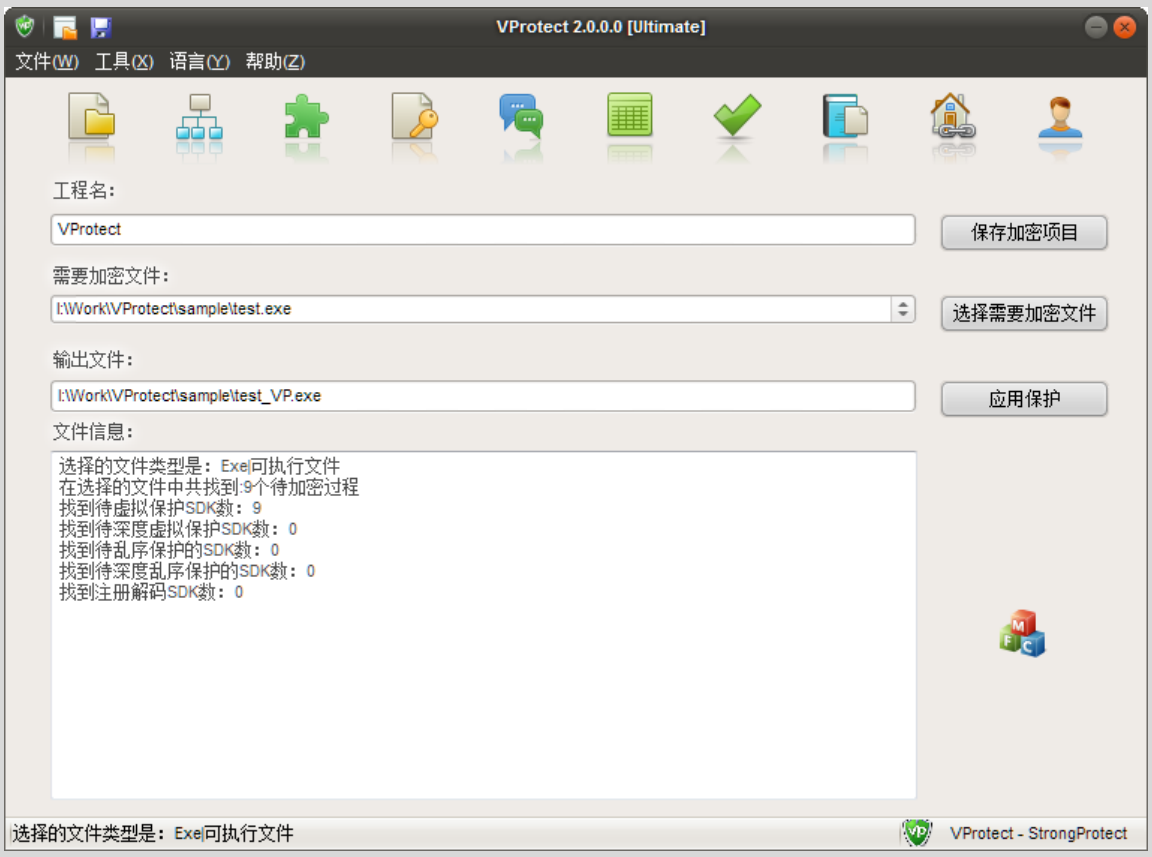
homepage:www.vprotect.net
Email:support@vprotect.net
Qq:1360505490

2011/2/2

目录

VPortect 产品简介

欢迎使用 VProtect 软件保护系统。这是软件的帮助文档。



VProtect 支持多种类型的文件格式：

- Win32 可执行文件 (*.exe);
- Windows 屏幕保护程序 (*.scr);
- 动态链接库 (*.dll);
- 32 位 ActiveX 控件 (*.ocx);
- 32 位驱动程序(*.sys)
- 其他 32 位可执行程序;
- 目前不支持原生 64 位可执行程序;

支持的编译器：

- Assembly language:
  MASM, FASM, POASM, TASM
- Basic:
  Visual Basic, Pure Basic, Power Basic
- C/C++:
  Visual C/C++, Borland C++ builder, Intel C++, Dev C++, Digital Mars C++, MinGW
- Pascal:
  Delphi, Free Pascal
- D Programming language:
  DMD
- 其他编译器

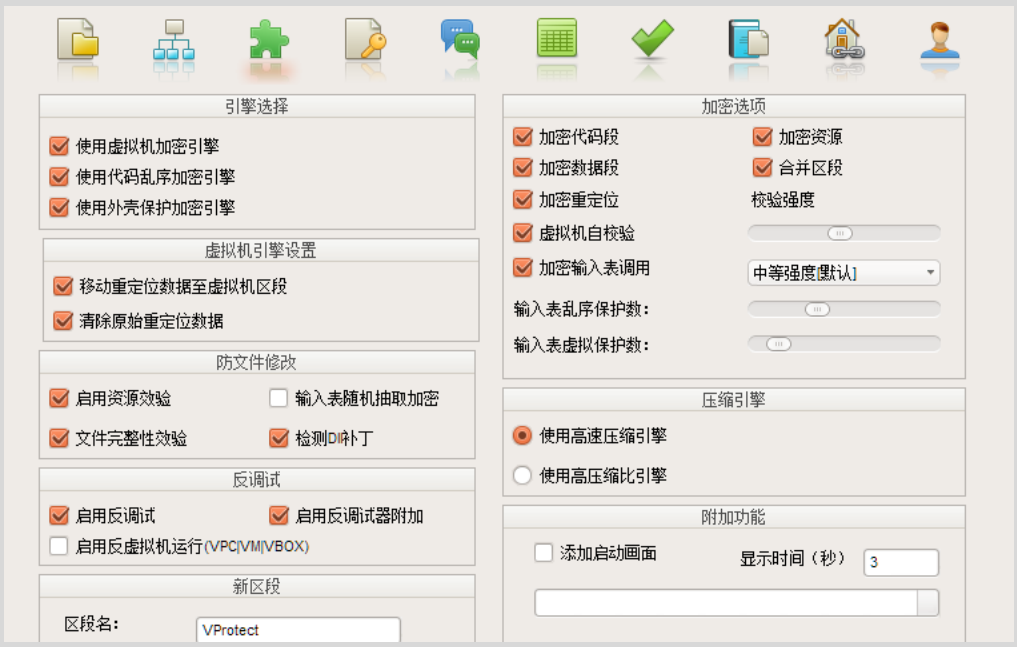
支持的操作系统：

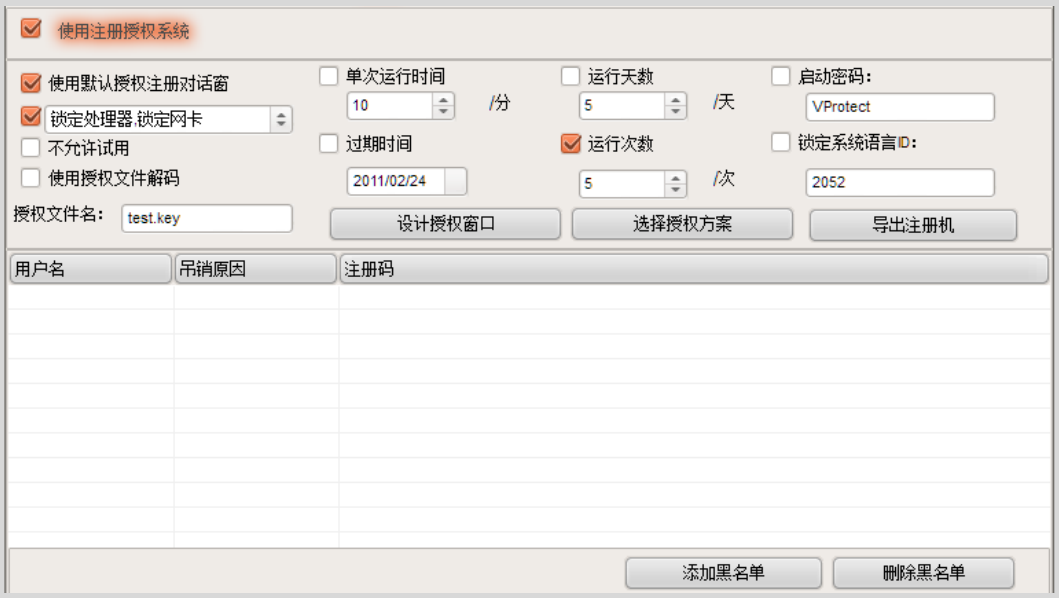
32 位 NT/2000/XP/2003/Vista/Win7/**win8**/2008 Server 及其对应 64 位版本。

Vprotect 功能

- 代码虚拟化保护
- 代码乱序保护
- 注册授权加密引擎
- 输入表加密
- 文件压缩
- 文件校验
- 区段合并
- 反内存转储存
- 资源防修改
- 资源加密
- 反调试
- 反调试器附加
- 反虚拟机执行

等实用功能





Http://www.Vprotect.Net

加密原理

保护时

Vprotect 加密方式是基于虚拟机解释引擎工作的。

原 始 的

```
00401000    6A 00                push    0
00401002    68 00304000          push    00403000
00401007    68 05304000          push    00403005
0040100C    6A 00                push    0
0040100E    A1 08204000          mov     eax, dword ptr
ds:[402008]
00401013    FFD0                 Call    eax
00401015    E8 02000000          Call    0040101C
0040101A    C3                   Retn
```

虚拟机指令翻译引擎

虚拟机指令

```
00430D7F    E8 0083C404          call    05079084
00430D84    74 0E                je      short
1_VP.00430D94
00430D86    EB 0E                jmp     short
1_VP.00430D96
00430D88    23E4                 and     esp, esp
00430D8A    2E:D975 0E           fstenv  (28-byte) ptr
cs:[ebp+E]
00430D8E    74 0C                je      short
1_VP.00430D9C
```

执行时

虚拟机指令

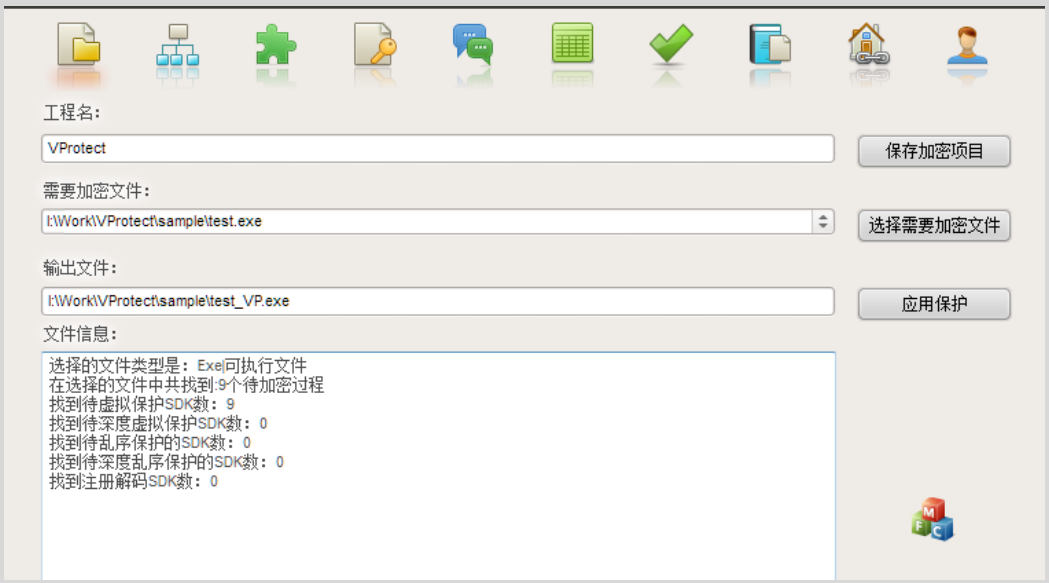
```
00430D7F      E8 0083C404              call    05079084
00430D84      74 0E                     je              short
1_VP.00430D94
00430D86      EB 0E                     jmp             short
1_VP.00430D96
00430D88      23E4                      and     esp, esp
00430D8A      2E:D975 0E                fstenv  (28-byte) ptr
cs:[ebp+E]
00430D8E      74 0C                     je              short
1_VP.00430D9C
```

虚拟机指令解释执行引擎

正确的结果。

如何使用代码保护

一：选择需要保护的文件



二：选择需要保护的流程

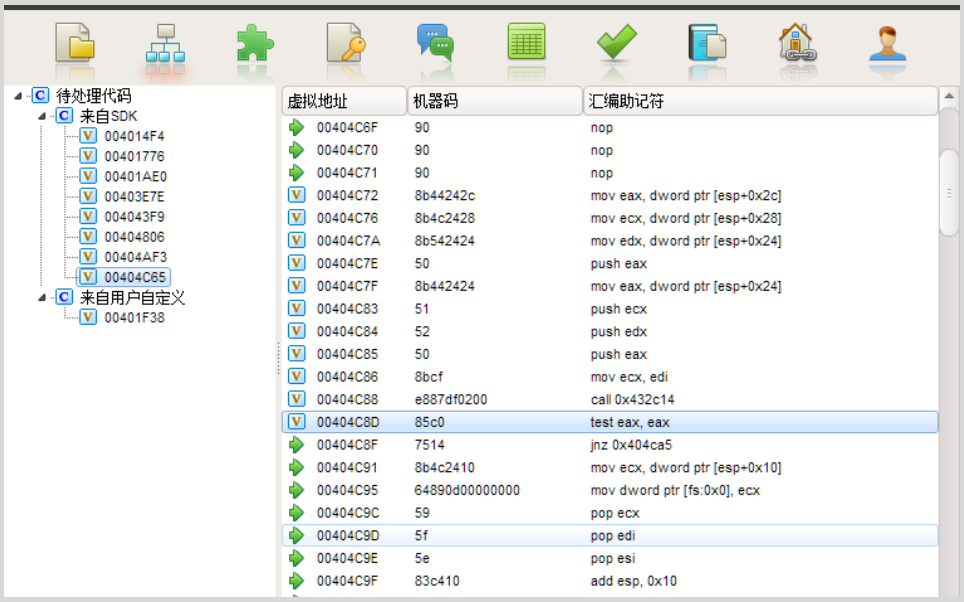
1：使用 SDK 添加流程

参考 Sdk 部分。

2：自定义流程

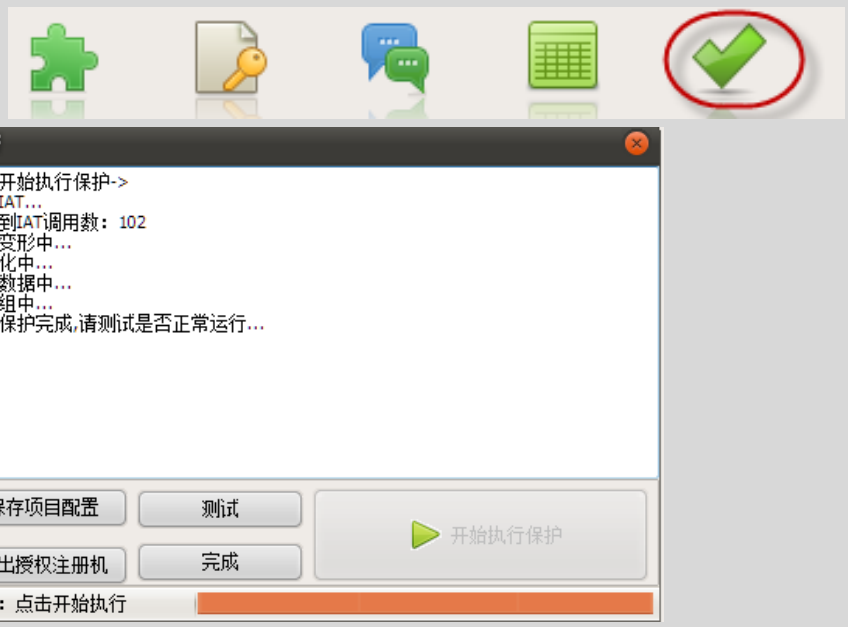
可以自定义流程处理方式。

3：查看流程



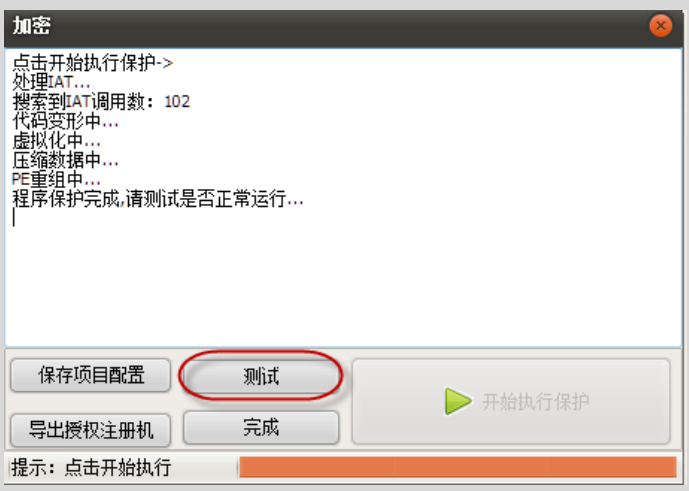
三：开始保护

1:



耐心等待保护完成。

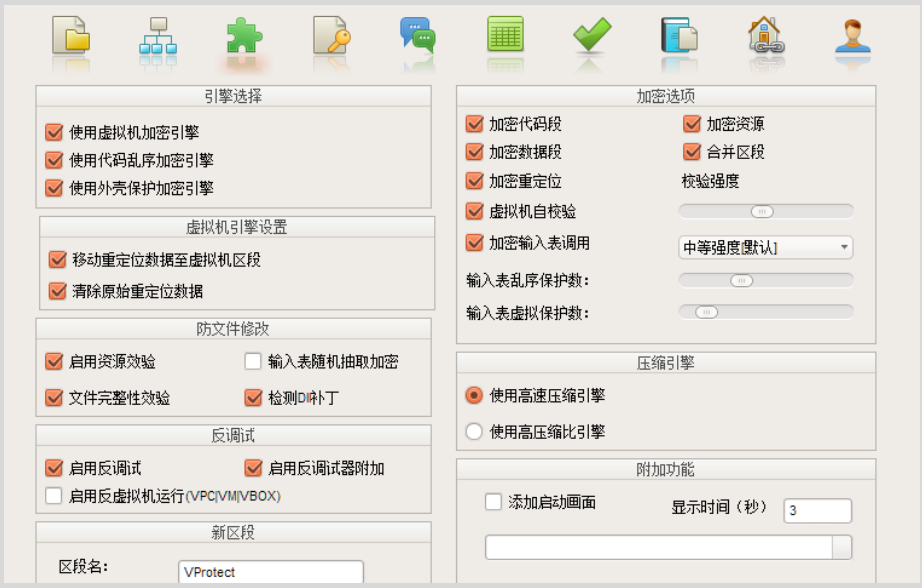
四：测试保护是否成功



五：完成保护：

测试功能是否正常。

如何使用外壳保护



一：什么是外壳保护

Vprotect 将代码保护引擎（虚拟机，乱序）和外壳保护引擎进行了区分。

你可以只使用代码保护引擎，也可以同时使用代码和外壳保护（推荐）。

外壳保护不同于代码保护，代码保护是针对一段指定代码（通长是关键算法），也就是局部的。

而外壳保护则是针对整个需要保护的程序，他是整体的。

二：加密选项介绍

加密需要保护程序的原始数据，减小源程序体积，增加保护强度。

1：Iat 加密等级选择根据需要

- GUI 建议选择中等强度
- 无 GUI 程序可以选择高强度。
- 对效率要求很高的可以使用最大速度

输入表乱序保护数：

是指使用乱序引擎保护 IAT 调用过程，效率很高。

输入表虚拟保护数：

是指用虚拟机保护引擎保护 IAT 调用过程，强度很高，但数量越大效率越低。

请酌情选择数值(建议在 20 左右具体可以根据程序保护后运行速度选择)。

2：反调试

反调试功能可以阻止您程序被逆向人员分析，可有效阻止逆向破解。

3：反虚拟机执行

可以设置是否允许程序在虚拟机(VPC|VM|VBOX)中执行。

4：压缩引擎

压缩可以使得保护后程序体积更小，更容易发布。

5：阻止修改

阻止非法修改文件内容，可以有效阻止被非法修改软件版权等重要信息。

6:资源加密

将程序使用到的资源数据加密起来，可以阻止非法修改。

7：添加启动画面

如果您的程序启动需要花费比较长的时间，我们建议您添加一个启动界面，提升用户的体验。

如何使用注册授权系统

什么是用户授权系统

我们知道共享软件一般是先提供试用，用户满意然后购买。

所以我们需要一个控制系统来方便分辨注册用户和未注册用户。

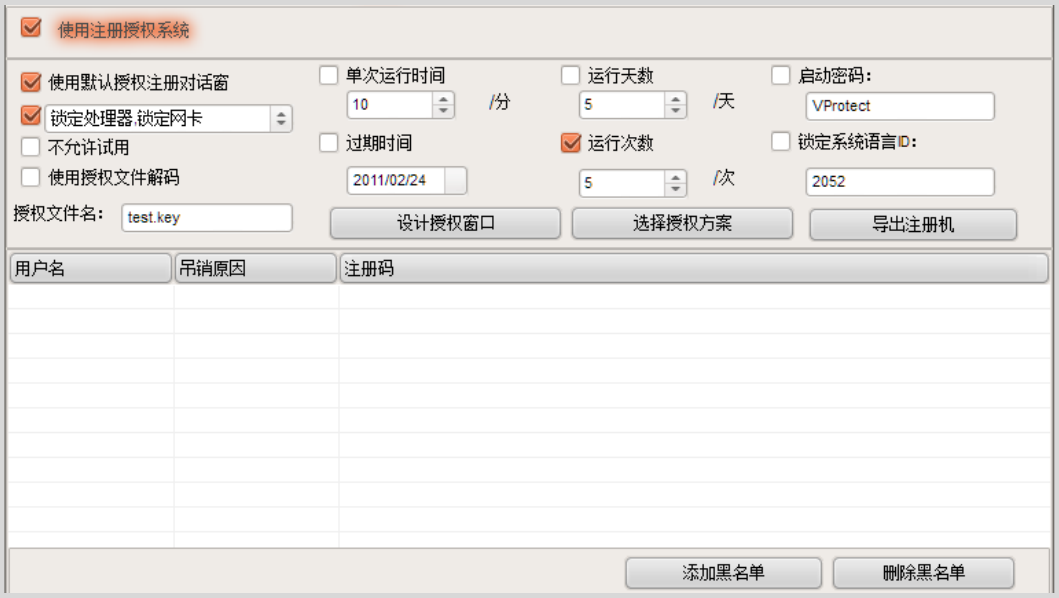
用户授权系统帮助软件开发人员完成了这些功能，您甚至不需要写一代码。

注意：只有专业版以上版本 Vprotect 才拥有授权功能。

使用注册授权系统可以方便的给你的软件加上高强度的注册授权功能，而你不需要写一行代码。

Vprotect 使用了 Rsa2048 位的公钥算法，可以有效的保护您的软件。

并且自动授权用户管理，注册机，黑名单等实用功能。



授权系统使用第一步（新建授权方案）

为什么要建授权方案

因为我们软件可能有很多版本。也可能我们软件每月都需要更新。但不能每次更新都发新授权文件给用户，我们需要让之前的有效授权可以正常使用。

这样我们就要为每个软件建一个授权方案，使我们更新软件的同时不影响之前的注册用户使用。

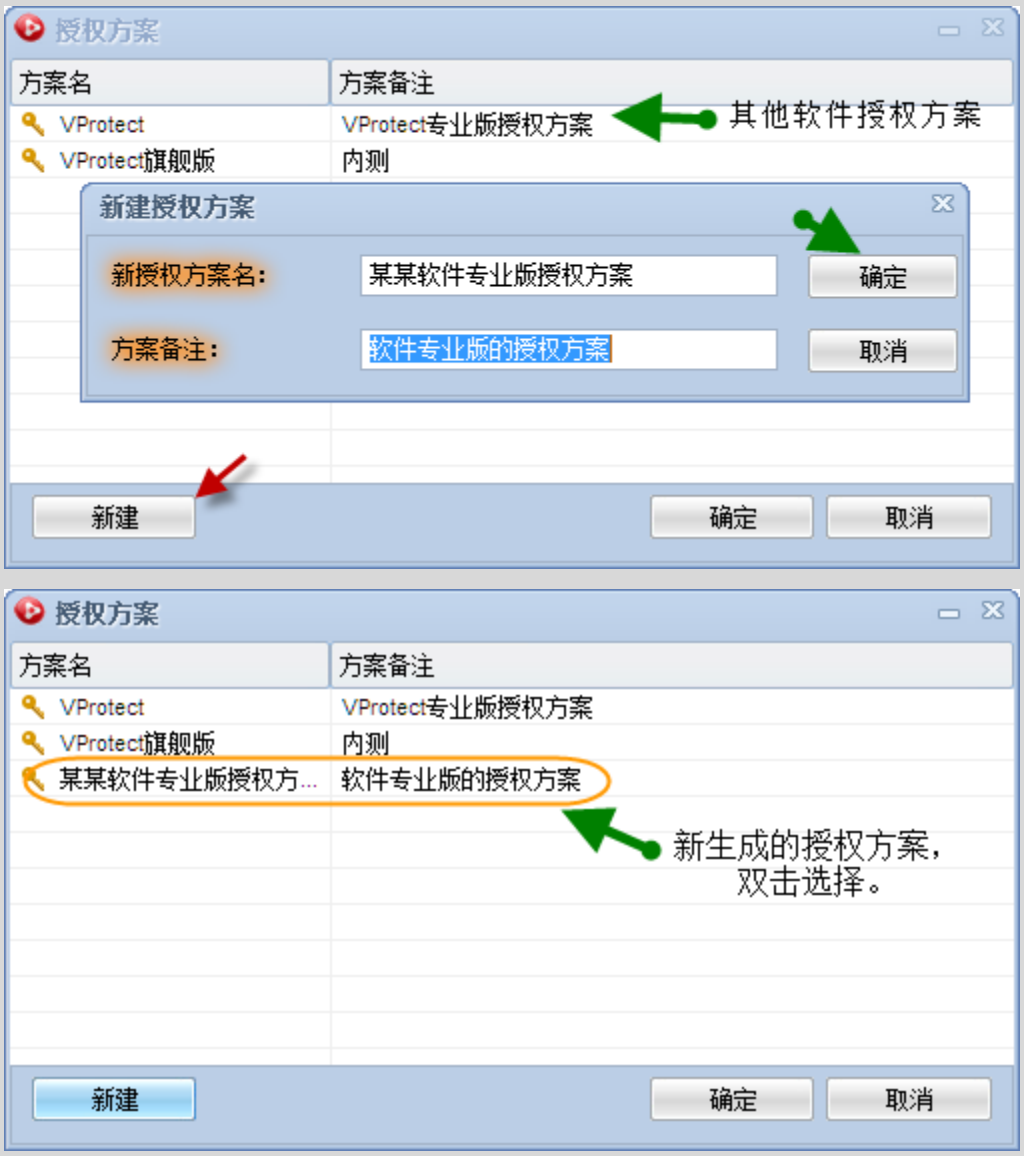
有了授权方案后，我们只需要每次加密使用同一个方案，之前授权文件将不需要更新。

新建授权方案

选择需要保护的文件。

根据提示新建一个授权工程，如果授权工程已经存在可以直接选择。

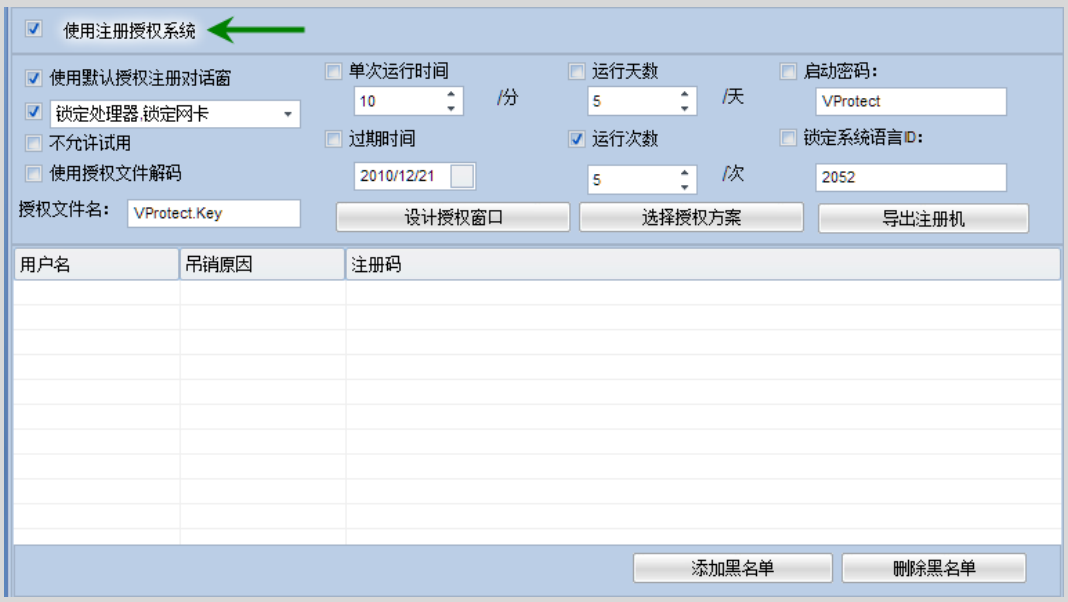
操作方法如图所示：



授权系统使用第二步（授权选项解释）

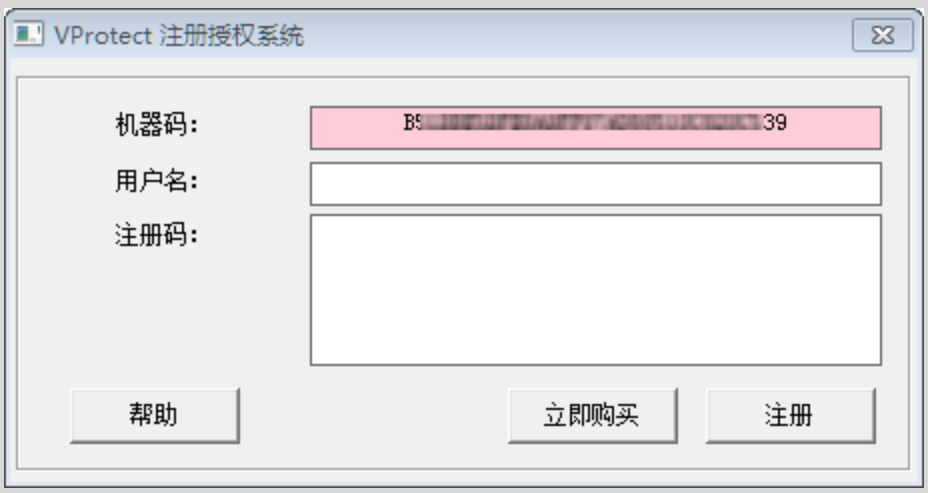
选择好虚拟机和外壳加密方案（使用方法参考帮助文档相应节）后切换到授权保护方案选项卡。

授权选项卡如图：



使用授权系统自带注册窗口功能，加密后的程序会显示一个授权窗口，方便用户输入注册码。

授权提示窗口如图：



试用限制可以提供给用户试用软件，当试用期满之后软件将不能试用。

锁定机器码功能可以将注册码和指定硬件绑定起来，也就是一个授权只能在一台电脑上使用。

授权文件解码功能是指程序数据将提供授权文件来解码，在没有授权文件或者授权文件无效的情况下，程序将不能运行。

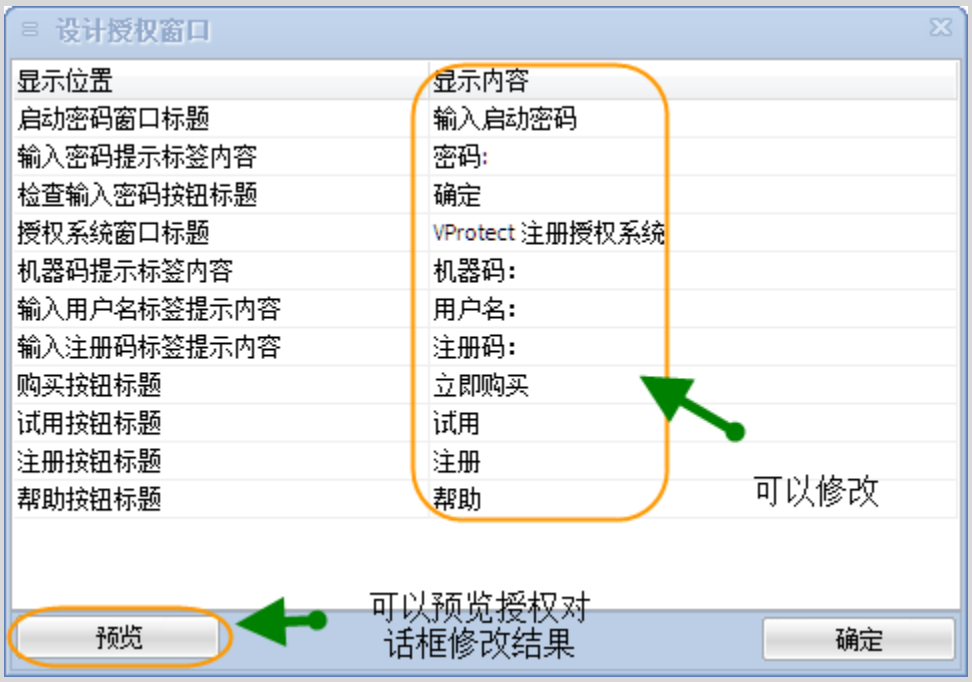
注意：使用授权解码功能后，软件将不能试用。

授权系统使用第三步（自定义授权提示和黑名单使用）

自定义授权窗口提示信息

Vprotect 可以自定义授权系统对话框。

在授权选项卡中，点击设计授权窗口，打开自定义编辑器。


设计授权窗口
显示位置
显示内容
启动密码窗口标题
输入启动密码
输入密码提示标签内容
密码:
检查输入密码按钮标题
确定
授权系统窗口标题
VProtect 注册授权系统
机器码提示标签内容
机器码:
输入用户名标签提示内容
用户名:
输入注册码标签提示内容
注册码:
购买按钮标题
立即购买
试用按钮标题
试用
注册按钮标题
注册
帮助按钮标题
帮助
可以修改
预览
可以预览授权对
话框修改结果
确定

黑名单功能使用

在上面步骤中完成了授权对话框自定义后。我们来学习黑名单功能使用。

什么情况下需要使用黑名单功能

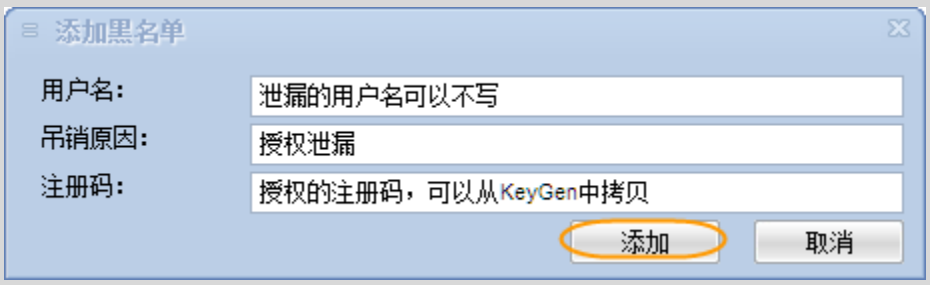
比如某个用户自己购买的授权泄漏了出去，而我们又不能因为这一个用户泄漏了授权而更换加密方案。因为更换加密方案会影响到其他正版用户。这时我们就需要将这个授权加入黑名单中。

使其失效。

如何使用黑名单

点击添加黑名单，在添加窗口中将泄漏用户的用户名，和注册码填写进去，点击确定。

这样泄漏的授权就无法使用了。

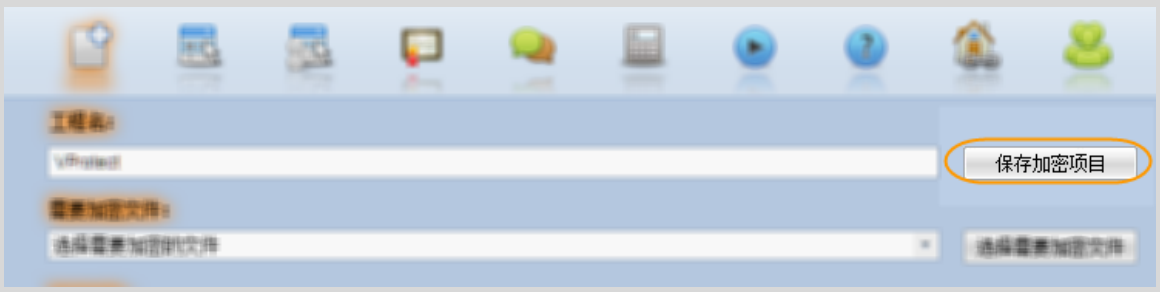


授权系统使用第四步（打开保存授权工程）

上面我们完成了授权选项，授权窗口自定义，和黑名单功能使用。

下面我们就要保存这些数据，这样就不需要每次加密都修改这些信息。

保存加密工程



将加密项目保存到加密程序目录或者自定义的目录（只要下次加密可以找到即可）。

注意：建议在每次修改授权数据，如添加黑名单，自定义授权窗口提示后都重新保存加密工程。

这样下次直接打开加密工程，就不需要重新修改了。

打开加密工程

上面说了怎么保存加密工程，只保存不会打开那不是没用么 o(∩_∩)o 。

我们这就来说怎么打开已经存在的加密工程。

在菜单栏-》文件-》打开项目

中选择上次保存的加密项目即可。

这样授权信息，授权窗口字串，黑名单就都不需要重新修改了。

授权系统使用第五步（导出注册机）

前面完成了授权配置，授权工程导出。现在我们要来导出注册机了。

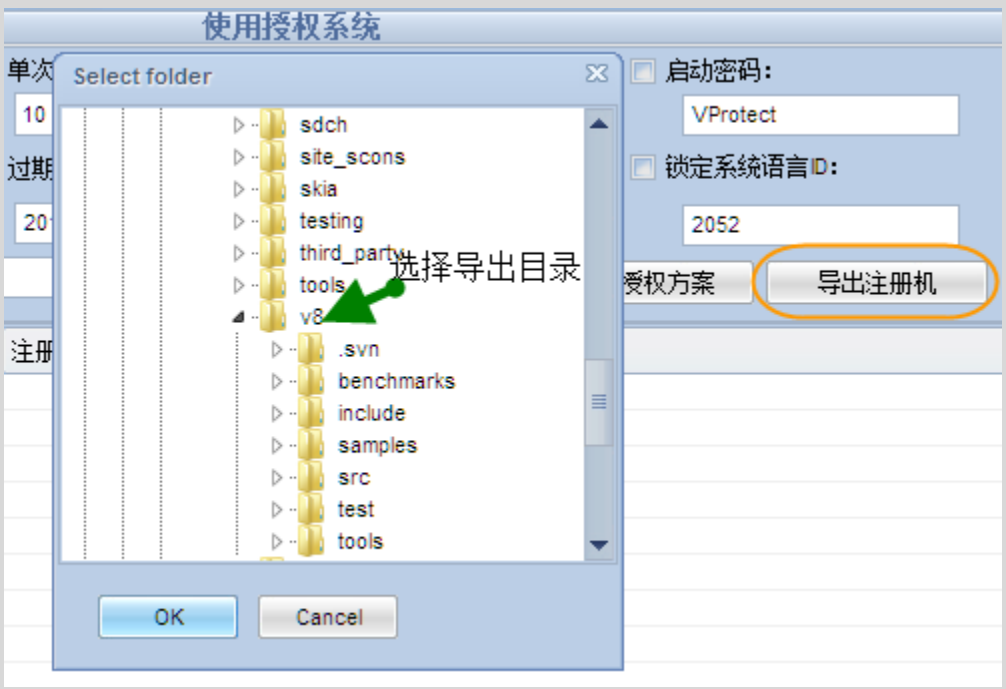
这步很重要，不然加密后的程序我们自己也无法授权了。

注意：在导出注册机之前，确定选择了授权方案。否则导出的注册机无法使用。

在授权方案和授权配置都完成后，我们就可以导出注册机了。

在授权选项卡中，点击导出注册机，选择一个导出目录，即可完成导出。

以后同一授权方案就不需要重复导出注册机了。



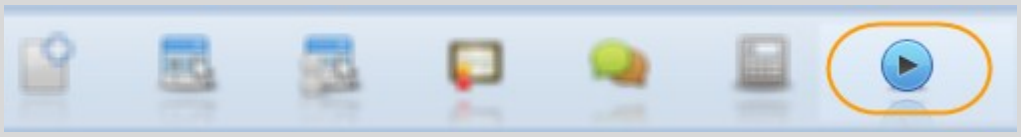
授权系统使用最后一步（完成加密）

上面我们完成了授权系统配置，授权工程保存,注册机导出，但最重要的加密还没说呢。

加密这么重要当然作为压轴出场。

压轴出场使用一定很难吧。

其实很简单只是点一个按钮而已。。



剩下只想呀耐心等待即可。。

授权使用这就说完了，如果你有不明白可以和我们联系。

使用注册机

软件发布后，用户注册，我们需要给用户一个授权许可。

授权就要使用我们在加密时导出的注册机来生成。

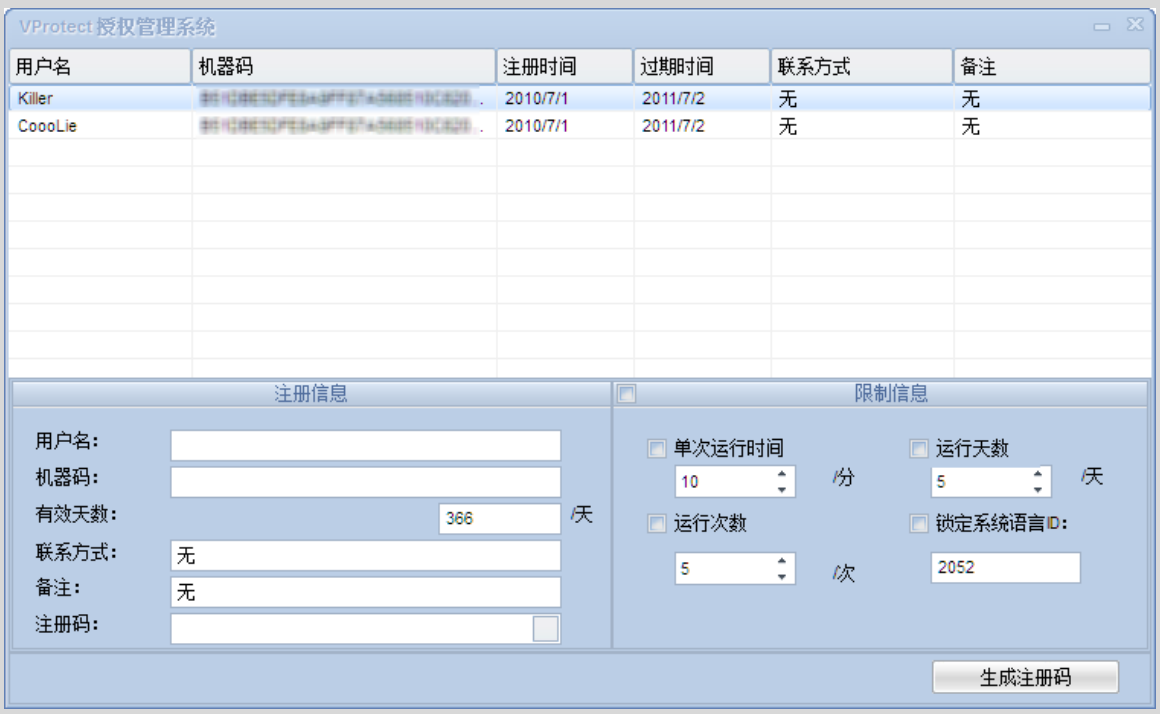
首先找到加密程序时注册机导出目录，找到注册机并运行。

根据用于提供的信息，用户名，机器码（加密时可选）来生成授权。

生成的授权文件将保存在以用户名为目录的目录下。

你可以直接发送授权文件给用户，也可以直接复制注册码字符串发送给用户来完成注册。

注册机运行界面如图：



用户名：填写用户的注册名。

机器码：填写用户机器的机器码，根据加密时设置不同。如果加密时选择了不锁定硬件则不需要填写机器码。（机器码可以提供授权窗口显示，或者可以提供 API 获得）。

有效天数：许可授权的有效天数，过期后授权将失效，用户需要程序购买许可授权。

联系方式：授权管理系统为了方便用户管理所提供的，可以不填写。

备注：同上可以不填写

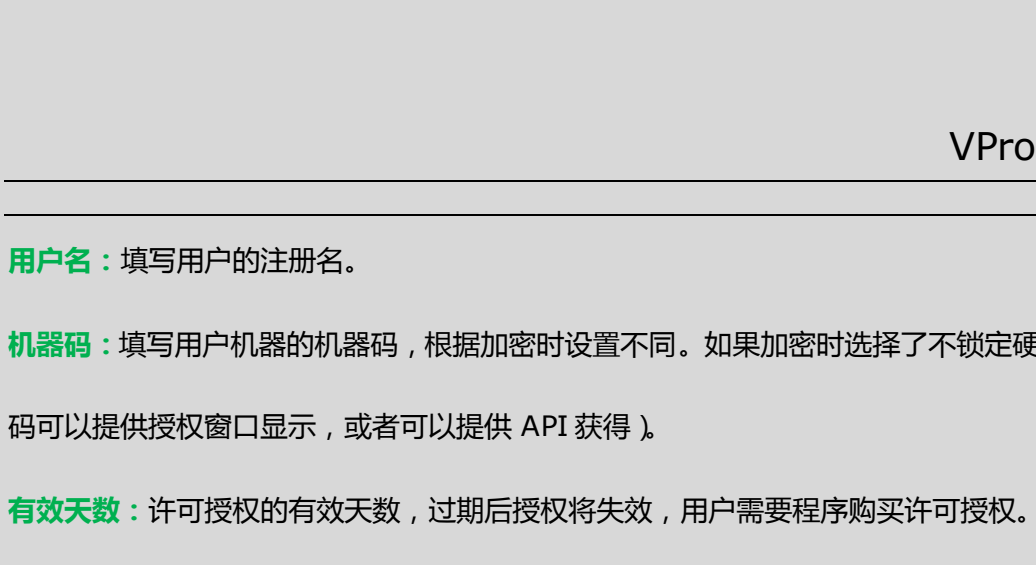
注册码：授权系统生成的注册码，您可以将他发给用户完成注册，也可以发送生成的授权文件。

限制信息：可以给授权设置限制来生成试用授权，可以不使用。

什么情况需要重新导出注册机

1:更换了授权方案

如图所示:

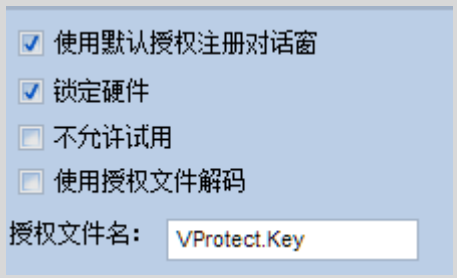


如果更换了新的授权方案,您需要重新保存工程方案并且导出注册机.

注意:之前的注册机将生成的注册码将不能用户新工程项目.

2:更换了授权选项

如图所示:



如果修改了这些选项中的任意一个，,您需要重新保存工程方案并且导出注册机.

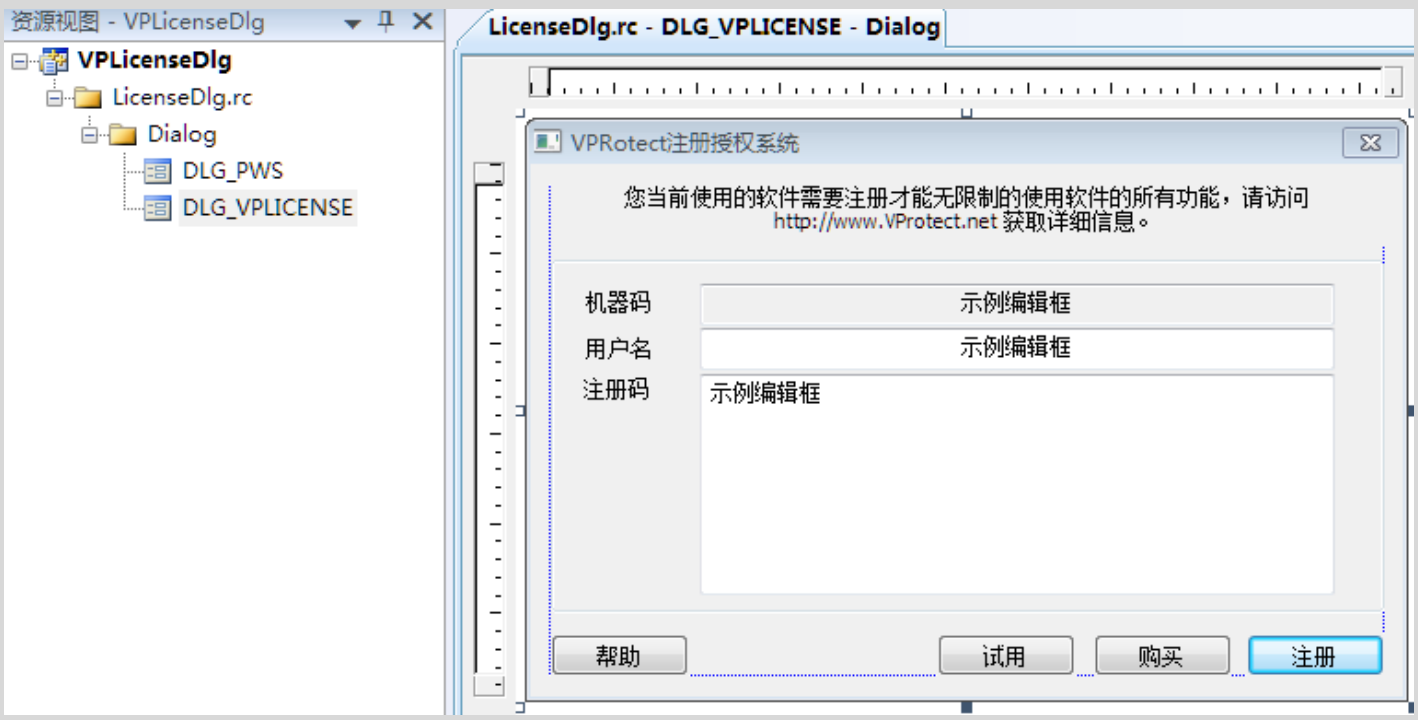
注意:之前的注册机将生成的注册码将不能用户新工程项目.

自定义授权窗口提示

Vprotect 允许用户自定义授权提示窗口，具体方法打开 DialogRes 目录。

使用 VisualStudio 编辑资源

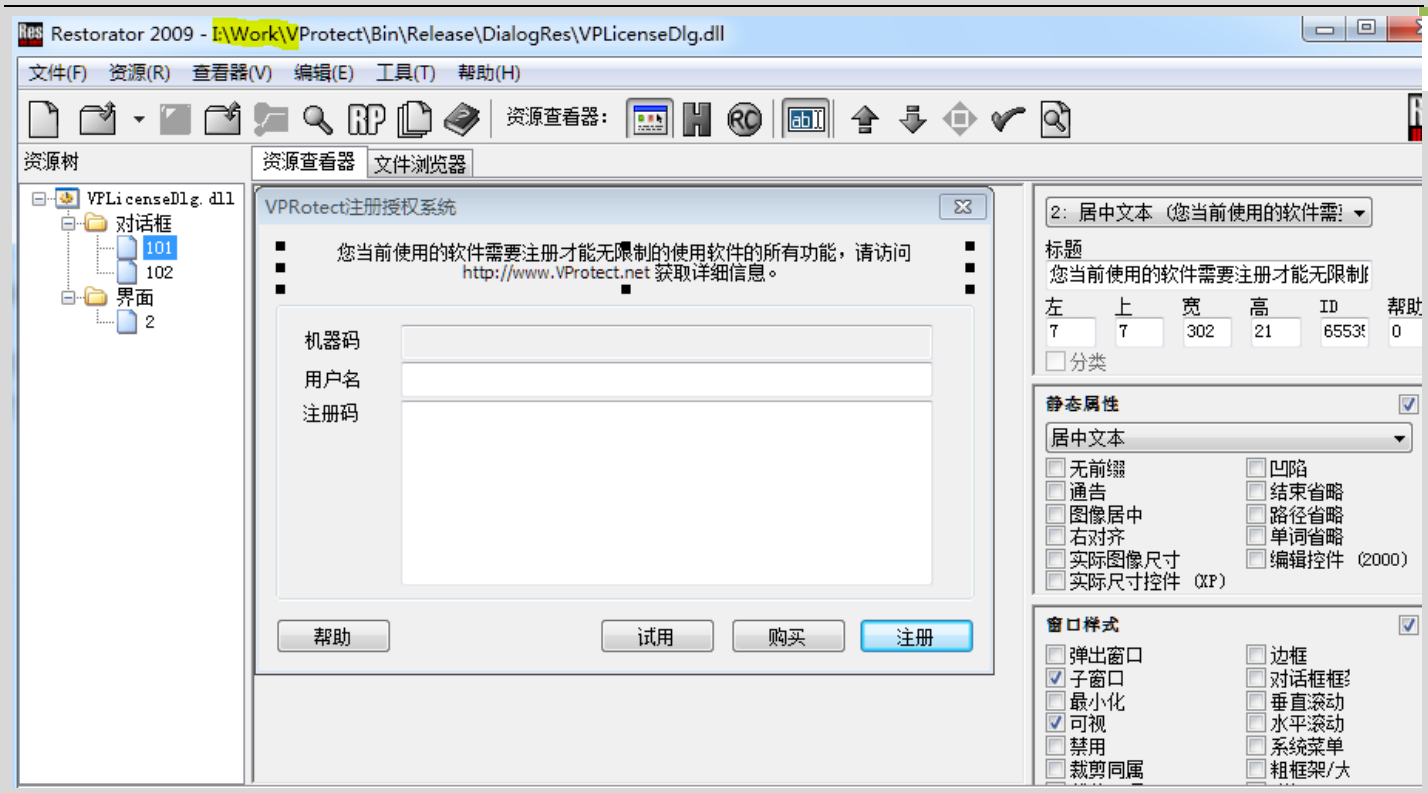
使用 VisualStudio 打开工程文件，切换到资源视图，既可自定义提示窗口，



使用资源编辑工具修改

如果没安装 VS 编辑器也可以使用一些资源编辑工具如 ResourceHack 打开 VPLicenseDlg，

将看到保护了 2 个对话框资源，用户可以借助资源修改工具完成自定义。



注意：切勿修改原控件本身 ID

使用网络云授权

什么是网络云授权

网络云授权是目前最新的共享软件授权验证解决方案，他通过类似客户端服务端的方法解决传统验证无法解决的问题。

网络云授权有什么优势

传统共享软件加密系统只能在本地做数据验证，这样导致数据可被伪造，验证系统可被修改。

并且作者无法及时作出反应。

而云授权将验证代码放在破解者无法接触的服务端机器上运行，并且服务端可以随时校验客户端数据，保证时时响应。

产品功能

一键添加网络云授权功能，无需添加任何代码，支持所有编译器的原生 32 位程序。

服务端做数据运算，校验，时时控制加密后程序运行，安全性更高。

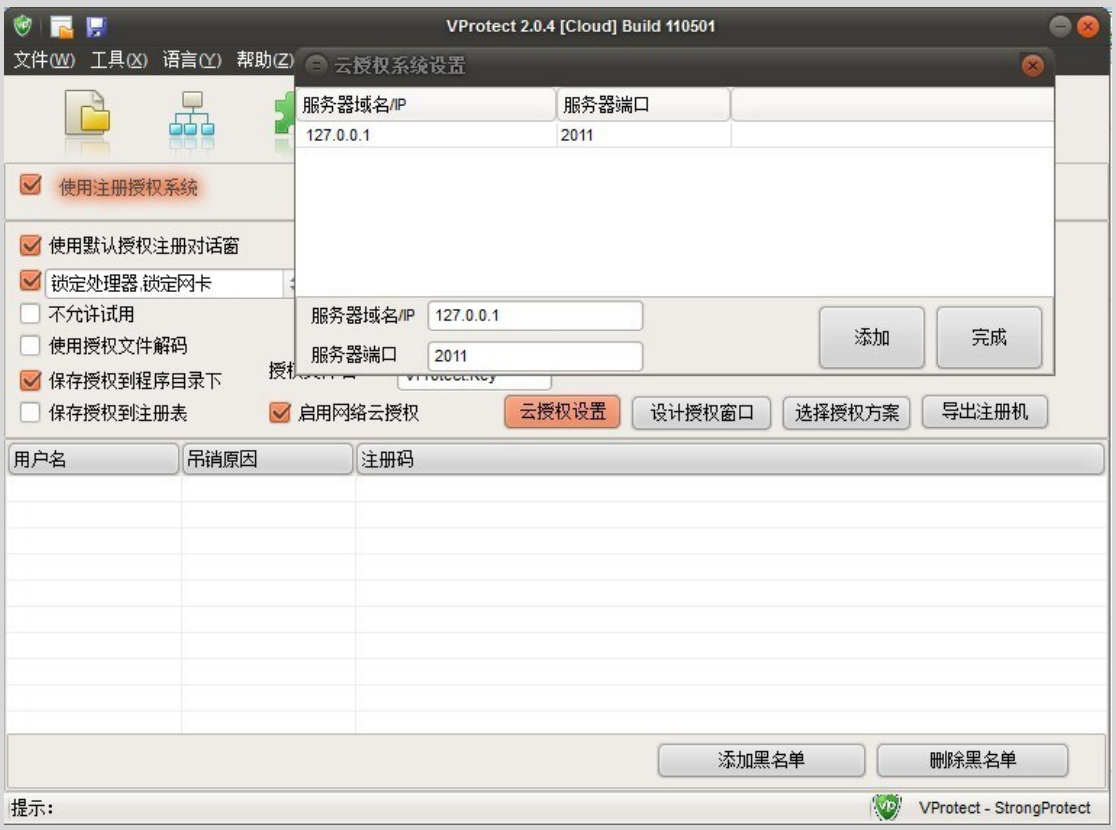
首个支持一机一码，一码多机等硬件锁定形式，锁定机器更灵活。

Http://www.Vprotect.Net

集成激活码批量生成，管理，限制等功能，使用方便。

更多功能，欢迎测试。

产品界面预览



服务端

使用方法

选择您的软件，或者将您的软件拖放到 VProtect 加密授权系统主窗口中。

根据具体情况选择你的加密选项。

如果您需要使用 Vprotect 作为程序的默认授权系统，请勾选”使用注册授权系统 “。

如果您想要使用网络云授权系统，请勾选上启用云授权系统，并点击云授权设置，填写服务端运行 ip/域名和端口（参考下节服务端使用）。

点击加密，开始加密您的软件。，完成加密后请导出注册机（服务端）。

服务端使用介绍

服务端是在加密软件完成时点击导出注册机按钮导出。

服务端是验证用户注册信息是否有效，控制用户机器锁定等功能所必需的。

运行条件

操作系统 :》Win2000

内存 :》=512M

CPU :》1.0G/HZ

接入互联网

配置服务端

服务端端口：请和加密程序时填写保持一致

工作线程数量：推荐默认设置

启动服务端

点击启动服务器按钮，服务器将正式开始运行，加密

程序运行将会连接服务器进行验证授权。

激活码生成

网络授权激活码可以批量生成，如果锁定硬件将自动

锁定当前用户登录机器。

并且支持一码多机（一个 注册吗绑定多个机器吗）。

激活码分组：随意填写方便管理

生成数量：可一次生成大量激活码，方便自动发货。

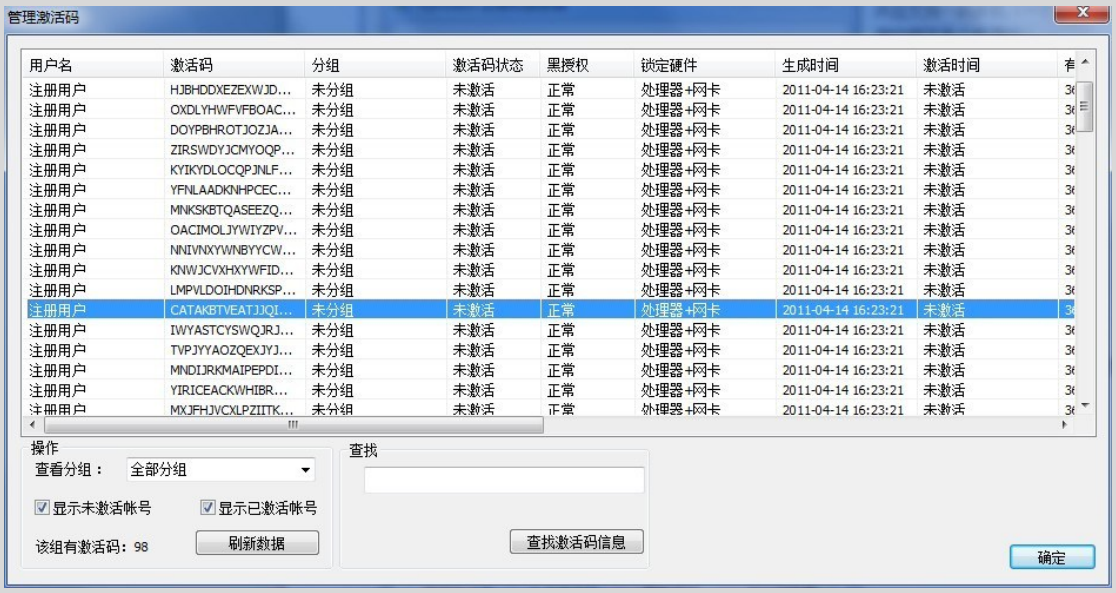
一码多机：一个注册吗绑定多个机器吗，默认只能绑定一个

限制功能：

可以对激活码添加限制信息。

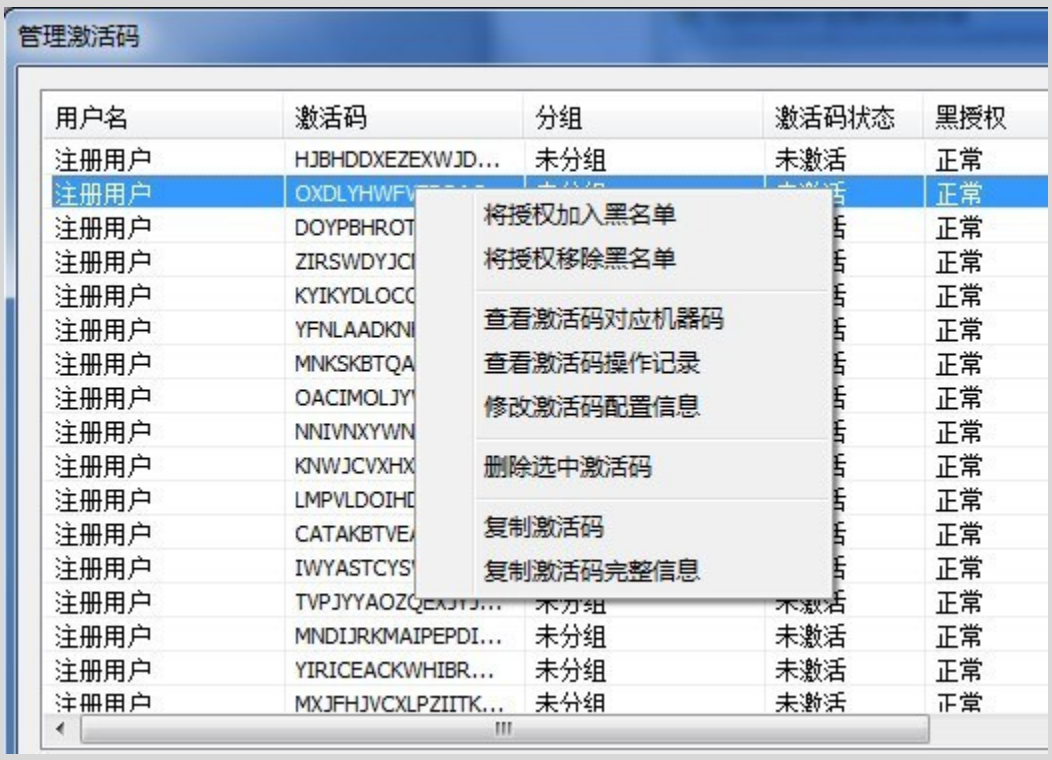
激活码管理

管理已经生成的激活码，查看激活码锁定相关信息。



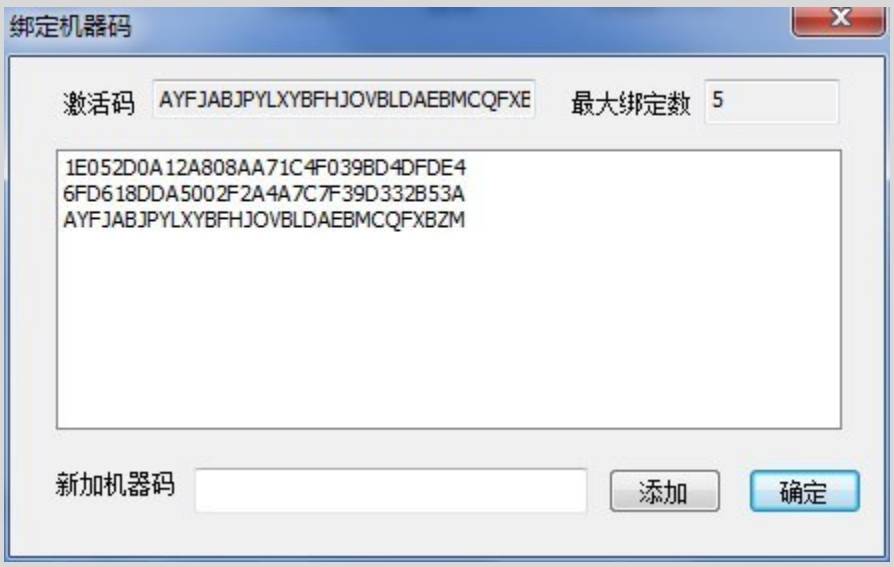
查看分组：可以查看指定分组激活码

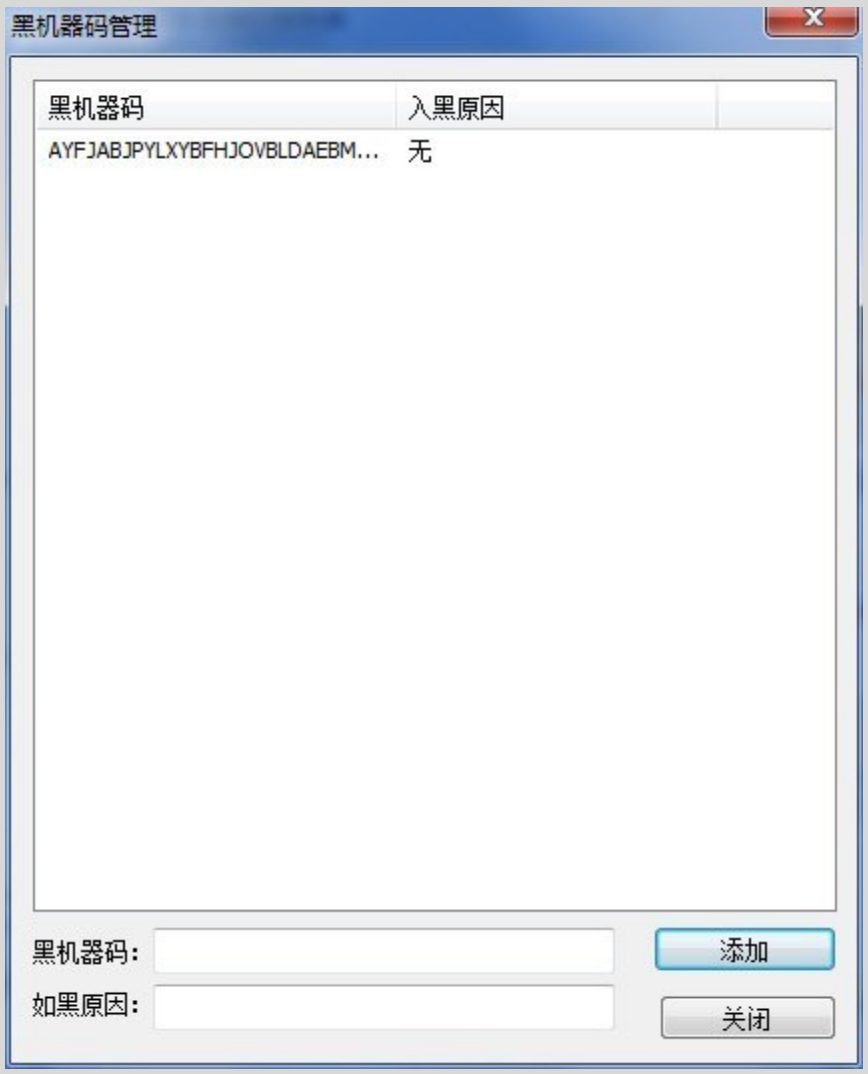
显示未激活账号：只显示未激活账号



将授权加入黑名单：在黑名单的授权将无法使用，登录时会收到提示。

查看激活码对应机器码：可以看激活码被哪些机器使用（如图）。





机器码拉黑

将指定机器码加入黑名单，在黑名单的机器码将无法通过授权验证。

IP 黑名单

什么是 ip 地址

IP 是英文 Internet Protocol（网络之间互连的协议）的缩写，中文简称为“网协”，也就是为计算机网络相互连接进行通信而设计的协议。

详细信息参阅：http://baike.baidu.com/view/8370.htm

如何使用

选中在用户在线列表中的用户，右键，将选中 IP 加入黑名单即可。

Ip 黑名单有效时间

从加入黑名单开始，到服务端重启或者管理员手动解出期间一直有效。

发布广播

批量给用户发送消息，主要用于发布版本更新提示。

后期工作计划

1. 完善云授权 API 接口
2. 添加脚本执行功能
3. 公开插件接口
4. 完善帮助文档

添加数字签名

如果您的软件需要添加签名请在使用 VProtect 加密时选中完整性效验兼容数字签名（如下图所示）。

加密完成后，即可用数字签名添加工具为您的程序签名。

注意：请先使用 VProtect 加密，后添加签名，勿颠倒操作顺序。

命令行（控制台）模式

Vprotect 为了方便用户项目自动化，提供了命令行模式，可以和编译器结合，实现自动加密。

注意：只有旗舰版极其以上版本包含命令行模式。

命令行模式有什么用

命令行模式可以和开发者编译器配合使用，实现编译后自动加密。

使用方法

VP 的命令行参数是 [-Vp] [工程文件名] [输入文件名] [输出文件名]。

如 VProtect_Con.exe -Vp C:\VPProject.Vp C:\NeedPack.exe C:\Out.exe

如果缺少[-Vp] [工程文件名]则使用默认配置方案。

如 VProtect_Con.exe C:\NeedProtect.exe C:\Out.exe

注意：文件路径需要是完整路径。

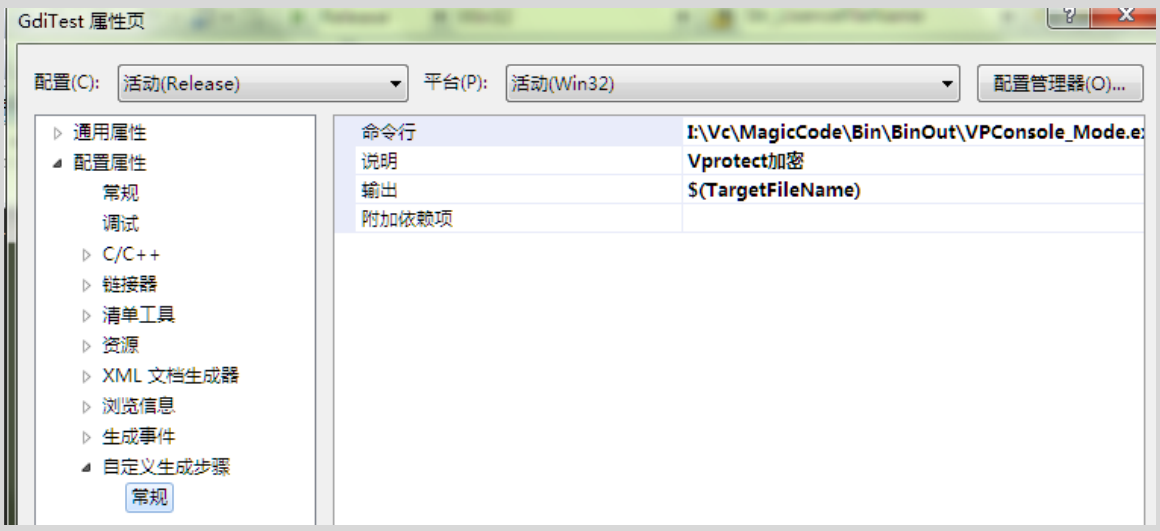
工程文件需要使用 VP 的 GUI 程序导出。第一次加密可以在 VP 的 GUI 程序窗口中设置好加密选项，然后保存工程

即可使用命令行模式。

编译器集成方法

这里以微软的 Visual Studio 2008 为例：

打开您的工程文件，点击菜单项目-》属性-》自定义生成步骤如图

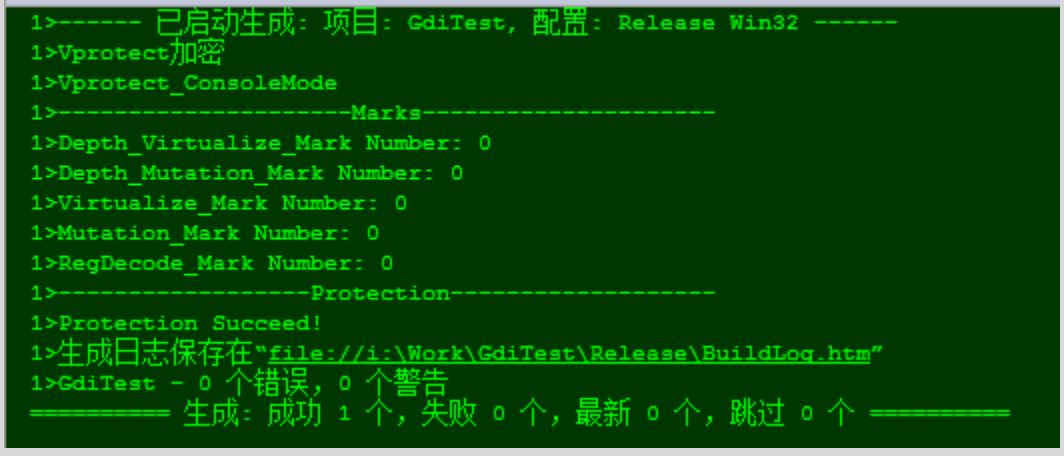


在命令行中输入 控制台模式的参数如：

I:\Vc\MagicCode\Bin\BinOut\VProtect_Con.exe -Vp I:\Vc\MagicCode\Bin\BinOut\1.vp $(TargetDir)$(TargetFileName) $(TargetDir)Out.exe

在输出中填写 VS 提供的宏 $(TargetFileName)即可。

编译项目输出如图



打开文件存放的文件夹会发现加密后的程序已经按照我们提供的文件名生成。

使用 SDK 保护您的软件

使用 Vprotect 通过的 Sdk 可以更好的保护您的软件。

Vprotect 中 SDK 分为二种类型

一：加密系统 SDK

二:授权系统 SDK 又名授权 API（只在相应的授权版本提供）

注意：加密 SDK 不能在同一过程（又名函数，子调用，或者子程序）中嵌套使用。

什么是 SDK

SDK（Software Development Kit，即软件开发工具包 ）一般是一些被软件工程师用于为特定的软件包、软件框架、硬件平台、操作系统等建立应用软件的开发工具的集合。

详细信息参考：http://baike.baidu.com/view/429424.htm

虚拟机加密标记

VP_SDK_VIRTUALIZE_BEGIN

VP_SDK_VIRTUALIZE_END

//功能：虚拟机加密开始标记

//备注：需要配合结束标记（VP_SDK_VIRTUALIZE_END）使用

使用该标记可以将你的代码使用虚拟机保护起来。

C/C++使用方法

```
{
VP_SDK_VIRTUALIZE_BEGIN
    //your Code
  VP_SDK_VIRTUALIZE_END
}
```

Delphi 使用方法

```
begin
    {$I Vp_Sdk_Virtualize_Begin.inc}
        //your Code
    {$I Vp_Sdk_Virtualize_End.inc}
End;
```

VP_SDK_VIRTUALIZE

//功能：虚拟机加密标记

//备注：**不需要结束标记，程序会自动定位函数结尾**

使用该标记可以将你的代码使用虚拟机保护起来。

C/C++使用方法

```
{
VP_SDK_VIRTUALIZE
    //your Code
}
```

Delphi 使用方法

```
begin
    {$I Vp_Sdk_Virtualize.inc}
        //your Code
End;
```

VP_SDK_DEPTH_VIRTUALIZE

//功能：深度虚拟机加密标记

//备注：**不需要结束标记，程序会自动定位函数结尾**

使用该标记可以将你关键代码，及其子调用全部使用虚拟机保护起来。

C/C++使用方法

```
{
VP_SDK_DEPTH_VIRTUALIZE
    //your Code
}
```

Delphi 使用方法

```
begin
    {$I Vp_Sdk_Depth_Virtualize.inc}
        //your Code
End;
```

代码乱序加密标记

使用该标记可以将代码顺序变形打乱，同时执行效率很高。

VP_SDK_MUTATION_BEGIN

VP_SDK_MUTATION_END

//功能：乱序加密开始标记

//备注：需要配合结束标记（VP_SDK_MUTATION_END）使用

C/C++使用方法

```
{
VP_SDK_MUTATION_BEGIN
     //your Code
  VP_SDK_MUTATION_END
}
```

Delphi 使用方法

```
begin
     {$I Vp_Sdk_Mutation_Begin.inc}
          //your Code
     {$I Vp_Sdk_Mutation_End.inc}
End;
```

VP_SDK_MUTATION

//功能：乱序加密标记

//备注：**不需要结束标记，程序会自动定位函数结尾**

C/C++使用方法

```
{
VP_SDK_MUTATION
     //your Code
}
```

Delphi 使用方法

```
Begin
     {$I Vp_Sdk_Mutation.inc}
          //your Code
End;
```

VP_SDK_DEPTH_ MUTATION

//功能：深度乱序加密标记

//备注：**不需要结束标记，程序会自动定位函数结尾**

C/C++使用方法

```
{
VP_SDK_DEPTH_MUTATION
    //your Code
}
```

Delphi 使用方法

```
begin
    {$I Vp_Sdk_Depth_Mutation.inc}
        //your Code
End;
```

注册解码标记

使用注册解码标记可以将只有在注册版才有效的代码保护起来。

并且必须在有一个有效授权的情况下才执行。

使用该标记可以使程序在无有效授权的情况下，不可能被破解。

注意：该加密标记在标准版中无效。

VP_SDK_REGDECODE_START

VP_SDK_REGDECODE_END

//功能：注册解码加密开始标记

//备注：**需要配合结束标记（VP_SDK_REGDECODE_END）使用，并且需要使用授权系统.**

C/C++使用方法

```
{
VP_SDK_REGDECODE_START
     // Only Run In registered Version
     //只有在注册版才执行
  VP_SDK_REGDECODE_END
}
```

Delphi 使用方法

```
Begin
  {$I Vp_Sdk_RegDeCode_Sta.inc}
     // Only Run In registered Version
     //只有在注册版才执行
```

```
{$I Vp_Sdk_RegDeCode_End.inc}
End;
```

一：C++使用加密系统 SDK 保护程序

SDK 位于 ..\Sdk\C++

1：

在需要保护的工程中，添加 SDK 头文件 Virtualize_Sdk.h。

2:

在关键函数中添加保护标记，见头文件。

```
#include "..\sdk\C++\Virtualize_Sdk.h"

//...其他代码
BOOL   CheckKey()
{
VIRTUALIZE_START
  //...你的代码
}

//或者使用深度保护 SDK
BOOL   CheckKey()
{
DEPTH_VIRTUALIZE_START
  //...你的代码
}
```

3：

使用编译器，编译你工程。

4：

使用 Vprotect 打开编译后的程序，进行保护。

5：

运行测试是否正常运行。

二：Delphi 使用加密系统 SDK 保护程序

SDK 位于 ..\Sdk\ Pascal

1:

在需要保护的过程中，添加 SDK 文件。

2：

在关键函数中添加保护标记。

```
//...其他代码
Function   CheckKey():BOOL;
begin
  {$I Vp_Sdk_Virtualize.inc}
  //...你的代码
end;

//或者使用深度保护 SDK
Function   CheckKey():BOOL;
begin
  {$I Vp_Sdk_Depth_Virtualize.inc}
  //...你的代码
end;
```

也可以直接内联 ASM 添加 SDK

```
//...其他代码
Function   CheckKey():BOOL;
begin
  //VP_SDK_VIRTUALIZE
  asm
    db $EB, $0B, $56, $69, $72, $74, $75, $61, $6C, $69, $7A, $65, $00
  end;  //...你的代码
end;

//或者使用深度保护 SDK
Function   CheckKey():BOOL;
begin
  //VP_SDK_DEPTH_VIRTUALIZE
  asm
    db   $EB, $11,$44, $65, $70, $74, $68, $5F, $56, $69, $72, $74, $75, $61, $6C, $69, $7A,
  $65, $00
  end;
```

```
    end;
```

三：其他程序加密系统 SDK 使用

SDK 头文件在程序目录..\SDK\目录下

使用方法同 C++ 和 DELPHI 不在一一介绍。

授权系统 API 使用

注意：使用了授权 API，则加密的时候必须使用授权系统，否则软件可能不能正常工作。

授权系统 SDK 又名授权 API，叫 API 是因为他是一个调用不在是一个标记。

有同学奇怪了，加密系统 SDK 和授权系统 SDK 有什么区别呢。

加密系统 SDK 是完成软件加密的提高软件强度。

授权系统 SDK 是配合软件开发人员完成专业的授权系统。

相当于 Vprotect 给软件开发软件提供了一套 API ,软件开发人员可以利用 Vprotect 提供的 API 编写高安全性的注册授权系统。

下面我们就来介绍一下授权系统的 API。

VP_Sdk_GetHardWareId

获取机器码字符串。

ANSI:

```
DWORD __stdcall VP_Sdk_GetHardWareIdA(
__in DWORD Sdk_Index,
__in_out char *Precv
);
```

Unicode

```
DWORD __stdcall VP_Sdk_GetHardWareIdW(
__in DWORD Sdk_Index,
__in_out WCHAR *Precv
);
```

参数

Sdk_Index

Vprotect 为每个授权 API 定义的一个索引常量。

所有 Vprotect 授权 API 都包含该参数。

Precv

用户缓冲区，用户接受返回结果。

大小 256 字节即可。

返回值

机器码字符串长度。

返回 0 表示失败。

Unicode

VP_Sdk_GetHardWareIdA（ANSI）

VP_Sdk_GetHardWareIdW(Unicode)

VP_Sdk_GetUserName

获取注册用户用户名.

ANSI
```
DWORD __stdcall VP_Sdk_GetUserNameA(
__in DWORD Sdk_Index,
__in_out char *Precv
);
```
Unicode
```
DWORD __stdcall VP_Sdk_GetUserNameW(
__in DWORD Sdk_Index,
__in_out WCHAR *Precv
);
```
参数

Sdk_Index

Vprotect 为每个授权 API 定义的一个索引常量。

所有 Vprotect 授权 API 都包含该参数。

Precv

用户缓冲区，用户接受返回结果。

大小 256 字节即可。

返回值

机器码字符串长度。

返回 0 表示失败。

Unicode

VP_Sdk_GetUserNameA（ANSI）

VP_Sdk_GetUserNameW(Unicode)

VP_Sdk_SaveKey

将注册码保存为授权文件。

ANSI
DWORD __stdcall VP_Sdk_SaveKeyA(
__in DWORD Sdk_Index,
__in char *PinputKey
);
Unicode
DWORD __stdcall VP_Sdk_SaveKeyW(
__in DWORD Sdk_Index,
__in WCHAR *PinputKey
);

参数

Sdk_Index

Vprotect 为每个授权 API 定义的一个索引常量。

所有 Vprotect 授权 API 都包含该参数。

PinputKey

存放注册码缓冲区地址

返回值

机器码字符串长度。

返回 0 表示失败。

Unicode

VP_Sdk_SaveKeyA（ANSI）

VP_Sdk_SaveKeyW(Unicode)

VP_Sdk_GetLeftCount

获取剩余使用次数（支持试用状态和授权状态获取）

```
DWORD __stdcall VP_Sdk_GetLeftCount(
__in DWORD Sdk_Index=VP_SDK_INDEX_GETLEFTCOUNT
);
```

参数

Sdk_Index

Vprotect 为每个授权 API 定义的一个索引常量。

所有 Vprotect 授权 API 都包含该参数。

返回值

软件剩余使用次数。

返回-1 表示无此限制。

VP_Sdk_GetUsedCount

获取已经使用次数（支持试用状态和授权状态获取）

```
DWORD __stdcall VP_Sdk_GetUsedCount(
__in DWORD Sdk_Index=VP_SDK_INDEX_GETUSEDCOUNT
);
```

参数

Sdk_Index

Vprotect 为每个授权 API 定义的一个索引常量。

所有 Vprotect 授权 API 都包含该参数。

返回值

软件已经用次数。

返回-1 表示无此限制。

VP_Sdk_GetVaildDay

获取授权有效天数（支持试用状态和授权状态获取）

```
DWORD __stdcall VP_Sdk_GetVaildDay(
    DWORD Sdk_Index=VP_SDK_INDEX_GETCVAILDDAY);
Sdk_Index
```

Vprotect 为每个授权 API 定义的一个索引常量。

所有 Vprotect 授权 API 都包含该参数。

对于该函数，该参数值一直是：VP_SDK_INDEX_GETCVAILDDAY。

返回值

试用状态：返回可以试用天数，无此限制返回-1.

授权状态：返回授权文件有效天数。

失败返回：0.

VP_Sdk_GetRegLeftDay

获取注册授权剩余可使用天数

```
DWORD __stdcall VP_Sdk_GetRegLeftDay(
__in DWORD Sdk_Index=VP_SDK_INDEX_GETREGLEFTDAY
);
```

参数

Sdk_Index

Vprotect 为每个授权 API 定义的一个索引常量。

所有 Vprotect 授权 API 都包含该参数。

返回值

注册授权剩余使用天数。

未注册返回-1。

VP_Sdk_GetTrialLeftDay

获取剩余试用天数

```
DWORD __stdcall VP_Sdk_GetTrialLeftDay(
__in DWORD Sdk_Index=VP_SDK_INDEX_GETTRIALLEFTDAY
);
```

参数

Sdk_Index

Vprotect 为每个授权 API 定义的一个索引常量。

所有 Vprotect 授权 API 都包含该参数。

返回值

剩余试用天数。

无此限制返回-1。

VP_Sdk_GetLicenseState

获取授权状态

```
DWORD __stdcall VP_Sdk_GetLicenseState(
__in DWORD Sdk_Index=VP_SDK_INDEX_GETLICENSESTATE
);
```

参数

Sdk_Index

Vprotect 为每个授权 API 定义的一个索引常量。

所有 Vprotect 授权 API 都包含该参数。

返回值

授权状态，可能是一下值中一个或多个，您可以使用 And 查询。

```
//授权状态

#define LICENSE_STATE_NOLICENSE_FILE 0x1          //无授权文件

#define LICENSE_STATE_ERROR                  0x2      //有错误发生

#define LICENSE_STATE_BLACKLISTED      0x4          //授权文件呗吊销

#define LICENSE_STATE_HARDID_BAD      0x8          //硬件 ID 错误

#define LICENSE_STATE_DATE_EXPIRED        0x10     //授权过期

#define LICENSE_STATE_RUNTIME_OVER        0x20     //使用时间过期

#define LICENSE_STATE_RUNCOUNT_OVER          0x40//使用次数过期

#define LICENSE_STATE_LANGID_BAD      0x80         //不支持系统语言

#define LICENSE_STATE_ACCEPT              0x100        //授权被接受

#define LICENSE_STATE_TRYMODE          0x200        //试用版本

#define LICENSE_STATE_REGMODE              0x400  //注册版本
```

VP_Sdk_GetCustomDword

这个函数可以获取生成 KEY 文件所填写的自定义 DWORD 值，

我们可以利用这个字段来区分程序各个版本。

```
DWORD __stdcall VP_Sdk_GetCustomDword
(DWORD Sdk_Index=VP_SDK_INDEX_GETCUSTOMDWORD);
```

参数

Sdk_Index

Vprotect 为每个授权 API 定义的一个索引常量。

所有 Vprotect 授权 API 都包含该参数。

返回值

生成注册文件时，所自定义的值。

使用范例：

```
#define V_Standard_Edition 1
#define V_Ultimate_Edition  2
BOOL GetSoftVersion()
{
    If(!VP_Sdk_IsRegister(DWORD Sdk_Index=VP_SDK_INDEX_ISREGISTER))
    {
        MessageBoxW(NULL,L"This Is Demo Version",L"Hint",64);
        return FALSE;
    }
    DWORD VersionNum= VP_Sdk_GetCustomDword
        (DWORD Sdk_Index=VP_SDK_INDEX_GETCUSTOMDWORD);
    If(V_Standard_Edition== VersionNum)
    {
        MessageBoxW(NULL,L"This Is Standard_Edition",L"Hint",64);
        return TRUE;
    }
    If(V_Ultimate_Edition==  VersionNum)
    {
        MessageBoxW(NULL,L"This Is Ultimate_Edition",L"Hint",64);
        return TRUE;
    }
    return FALSE;
}
```

VP_Sdk_IsRegister

判断程序是否已经授权

```
BOOL __stdcall VP_Sdk_IsRegister(
__in DWORD Sdk_Index=VP_SDK_INDEX_ISREGISTER
);
```

参数

Sdk_Index

Vprotect 为每个授权 API 定义的一个索引常量。

所有 Vprotect 授权 API 都包含该参数。

返回值

注册用户返回 TRUE.

未注册用户返回 FALSE.

VP_Sdk_CheckLicenseInMem

检测内存授权数据

BOOL __stdcall VP_Sdk_CheckLicenseInMem(DWORD Sdk_Index,BYTE *PMemKey,DWORD KeySize);

参数

Sdk_Index

Vprotect 为每个授权 API 定义的一个索引常量。

所有 Vprotect 授权 API 都包含该参数。

PMemKey

授权数据的内存地址。不能为无效地址。

KeySize

授权数据大小。

返回值

授权数据有效返回 TRUE

授权数据无效返回 FALSE

注意

- 授权数据是 KeyGen 生成的授权文件字节，非 Unicode 数据请勿使用文本函数读取授权文件。
- 正确的方法应该使用 CreateFile->ReadFile 读取。
- 或者使用注册机 API VP_SDK_GenKeyA 获得。
- 该方法检测时不生成文件，适合一次性使用。

授权系统注册机 API

使用授权系统注册机 API 可以方便在各种环境下使用 Vprotect 提供的注册授权系统。

C|C++使用方法

包含文件

```
..\Sdk\C++\VP_LIsence_Sdk.h
..\Sdk\Lib\Vp_KeyGen.lib
/*
* GenKey Interface
* 生成注册码 API
*/

DWORD_PTR __stdcall VP_SDK_GenKeyW(
__in PSDK_GENKEY_STRW PSdk_UserW
);
```

参数：

PSdk_UserW 是一个指向 SDK_GENKEY_STRW 结构体的指针

UniCode 版

```
typedef struct _SDK_GENKEY_STRW
{
    WCHAR UserName[64];          //用户名空终止字符串
    WCHAR HardID[33];            //用户机器码可以使用授权 API 获得
    DWORD_PTR VaildDay;          //指定注册码有效天数
    DWORD_PTR CanRunCount;       //限制使用次数
    DWORD_PTR CanRunDay;         //限制使用天数
    DWORD_PTR CanRunMinute;      //限制每次使用时间，单位：分
    DWORD_PTR CanRunLangId;      //限制运行的系统语言标识
    DWORD_PTR CustomDword;       //自定义 DWORD 值
    BOOL B_CreateKeyFile;        //是否生成注册文件
    WCHAR PRecvKey[2048];        //用户接受返回的注册码
}SDK_GENKEY_STRW,*PSDK_GENKEY_STRW;
```

AnsiChar 版：参数意义同上

```
typedef struct _SDK_GENKEY_STRA
{
    char UserName[64];
    char HardID[33];
    DWORD_PTR VaildDay;
    DWORD_PTR CanRunCount;
    DWORD_PTR CanRunDay;
    DWORD_PTR CanRunMinute;
    DWORD_PTR CanRunLangId;
    DWORD_PTR CustomDword;
    BOOL B_CreateKeyFile;
    char PRecvKey[2048];
}SDK_GENKEY_STRA,*PSDK_GENKEY_STRA;
```

返回值

注册码长度。

失败返回 0.

备注：

使用前，请把生成的注册机目录下 Vp_GenKey.Dat 文件一并拷贝。

否则无法生成正确授权。

Delphi 使用方法

将 ..\Sdk\VP_Lisence_Sdk.pas 文件添加到程序工程。

```
{
* GenKey Interface
* 生成注册码 API
}
```

[AnsiChar 版本]

```
function VP_Sdk_GenKeyA(PSdk_UserA:PSDK_GENKEY_STRA):DWORD;stdcall;
```

[UNICODE 版本]

```
function VP_Sdk_GenKeyW(PSdk_UserW:PSDK_GENKEY_STRW):DWORD;stdcall;
```

参数

PSdk_User 是一个指向 PSDK_GENKEY_STR 结构体的指针

结构体定义如下

[UNICODE 版本]

```
  SDK_GENKEY_STRW = record
    UserName:array  [0..63] of WideChar;      //用户名
    HardID:array  [0..32] of WideChar;        //用户机器码，可以使用授权 API 获得
    VaildDay:DWORD;                           //授权有效天数
    CanRunCount:DWORD;                        //限制使用次数
    CanRunDay:DWORD;                          //限制使用天数
    CanRunMinute:DWORD;                       //限制每次使用时间，单位分
    CanRunLangId:DWORD;                       //限制使用系统语言标识符
    B_CreateKeyFile:BOOL;                     //是否产生授权文件
    PRecvKey:array  [0..2047] of WideChar;    //接受返回注册码
  end;
  PSDK_GENKEY_STRW = ^SDK_GENKEY_STRW;
```

[ANSICHAR 版本]

```
  SDK_GENKEY_STRA = record
    UserName:array  [0..63] of AnsiChar;
    HardID:array  [0..32] of AnsiChar;
    VaildDay:DWORD;
    CanRunCount:DWORD;
    CanRunDay:DWORD;
    CanRunMinute:DWORD;
    CanRunLangId:DWORD;
    B_CreateKeyFile:BOOL;
    PRecvKey:array  [0..2047] of AnsiChar;
  end;
  PSDK_GENKEY_STRA = ^SDK_GENKEY_STRA;
```

返回值

注册码长度。

失败返回 0.

备注：

使用前，请把生成的注册机目录下 Vp_GenKey.Dat 文件一并拷贝。

否则无法生成正确授权。

用户协议

网址：http://www.vprotect.net

电子邮箱：support@vprotect.net

开发商：VProtect_Team

技术支持 QQ：1360505490

许可协议

使用前请仔细阅读“VProtect”以下条款和条件。该许可协议是您和“VProtect 开发团队”的合同。通过使用“VProtect”你是同意所有的限制和“VProtect 开发团队”提供的规则。如果您不同意以下许可协议，你不应该使用或下载或储存“VProtect”。

1。 VProtect 是一对“VProtect 开发团队”知识产权的成果，是受知识产权法律保护。您可以使用“VProtect”只在当前许可协议的范围。如果您不同意提出，请不要使用或下载或储存“VProtect”的条款。

2。 使用本“软件”由用户自己承担风险，VProtect 团队及合作单位对本“软件”不作任何类型的担保，不论是明示的、默示的或法令的保证和条件，包括但不限于本“软件”的适销性、适用性、无病毒、无疏忽或无技术瑕疵问题、所有权和无侵权的明示或默示担保和条件，对在任何情况下因使用或不能使用本“软件”所产生的直接、间接、偶然、特殊及后续的损害及风险，VProtect 团队及合作单位不承担任何责任。

3。 “VProtect”不是一个免费软件。在试用版中的限制不会出现在注册版本中。请参阅为注册版本以下许可协议。如果你觉得 Vprotect 系列软件非常有用，并希望利用其所有功能，你就必须购买注册版。这项注册费将授权一个或多个用户许可证副本。

用户在遵守法律及本《协议》的前提下可依本《协议》使用本“软件”。用户无权实施包括但不限于下列行为：

1。您不能，也不允许或允许他人，出售，分发，出租，出借，租赁，“VProtect”分许可证。

2。通过修改或伪造软件作品运行中的指令、数据、数据包，增加、删减、变动软件的功能或运行效果。

3。您应遵守所有有关法律，法规和规章影响“VProtect”或它的任何部分，包括有限的功能评估副本。

4。您不能使用 VProtect 加密木马，病毒，后门等违反相关法律犯规的应用程序。

5。 其他以任何不合法的方式、为任何不合法的目的、或以任何与本协议不一致的方式使用本软件和和 VProtect 提供的其他服务；

6。用户若违反上述规定，VProtect 有权采取终止、完全或部分中止、限制使用功能。

注册版本许可

1。对“VProtect”注册版本允许你注册单个或公司的授权版本。注册信息包含注册码，你不能公布或分发。若你分发或发布的注册信息，“VProtect”许可使用将自动终止。

常见问题：

一：什么是 Virtualize Protect：

Virtualize Protect(VProtect) 是一款基于虚拟机和乱序等引擎的可执行文件加密保护系统。

二：什么是代码保护引擎

代码保护引擎是基于分析程序源代码功能，然后使用虚拟或者乱序变形等手段对代码加密保护方法的总称。

三：使用 VProtect 代码保护引擎有什么好处

代码保护引擎的优势十分明显。因为他是提供分析源程序代码，任何进行加密。所以被加密后的程序将不会在出现源程序代码。如果使用虚拟机保护甚至不会程序 X86 代码。

四：什么是外壳保护引擎

外壳保护引擎相当于给程序加了一层壳。就像一些果实的外壳一样，如果你想吃到果实。那么你首先需要破坏他的外壳。

五：使用 Vprotect 的外壳引擎有什么好处

因为外壳可以在程序启动的时候获取完整的流程控制权限。这样可以更好的保护您的软件。

他可以在程序强度的时候检测文件是否被非法修改，文件解压，自校验等。

六：什么是虚拟机加密引擎：

软件保护的所指的虚拟机不同于 Vbox,Vpc。软件保护的虚拟机用于执行 X86 指令被转换为自定义格式的指令。

七：虚拟机保护有什么好处：

虚拟机保护比常规的加密代码保护强度更高。他将原始可读的 X86 指令翻译成只有虚拟机可读的指令。如果别人想分析 破解 逆向，他需要分析虚拟机的指令含义。Virtualize Protect 的指令，每次生成都随机打乱。所以他将 分析 破解 逆向 这一技术活，转换成 高难度技术活，和高强度的体力活。所以使用 Virtualize Protect 保护后的程序，只有极少数高手可以继续分析，大大的提高了软件的安全性。

八：什么是乱序保护引擎：

乱序引擎，是一种可以分析源程序流程，并且生成被混乱后的流程。它通过很小的效率开销，获得了很高的强度。是目前高强度加密程序必备引擎。

九：Vprotect 乱序引擎有什么特点：

Vprotect 乱序引擎不仅扰乱了原程序执行流程，并且在流程中夹杂随机算法生成的代码。

防止了动态和静态逆向分析。

十：深度保护和普通保护有什么不同。

深度保护会自动搜索函数的子调用，并一同保护，相对普通保护，深度保护强度更高，更安全。所以普通保护是平面的，而深度保护是立体的。

例：

```
//普通保护
BOOL Md5(char  *UserName)
{
  //Md5 算法代码不会被 Vprotect 保护
}
BOOL   CheckKey()
{
  VIRTUALIZE_START
  //...你的代码
  Md5(UserName);
}
//深度保护
BOOL Md5(char  *UserName)
{
  //Md5 算法代码会被 Vprotect 自动保护
}
//使用深度保护 SDK
BOOL   CheckKey()
{
  DEPTH_VIRTUALIZE_START
  //...你的代码
  Md5(UserName);
}
```

十一：什么是流程

也就是函数过程，一个函数是一个流程。

```
BOOL   CheckKey()
{← 流程开始位置

  VIRTUALIZE_START
  //...你的代码
```

}← 流程结束位置

十二：什么是 VA,RVA,RAW

VA 全称 VirtualAddres,中文翻译为 虚拟地址，是程序加载到内存后在的地址。

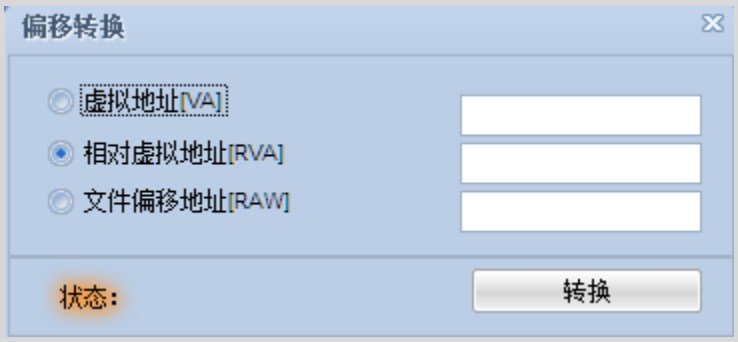
RVA 全称 Relative VirtualAddres，中文翻译为 相对虚拟地址。是程序加载到内存后在的地址相对于文件加载基质的偏移。和 VA 地址的换算公式为：RVA = VA – ImageBase

具体信息可以查看 PE 文件格式相关书籍。

什么是 RAW

RAW 是二进制文件偏移地址，可以使用 16 进制文件查看工具查看。

VA,RVA,RAW 的相互转换可以使用 Vprotect 提供的偏移转换工具。



十三：使用 Vprotect 保护好程序不能正常运行。

这个可能是由于我们程序 BUG 导致，也可能是你程序的特殊性导致。

遇到这种问题，你可以和客服联系，也可以通过邮箱将保护失败的程序发送给我们进行分析。

BUG 反馈格式

你使用的操作系统版本：*
Vprotect 程序版本*
保护选项*
错误信息
你的联系方式

十四：使用注册授权系统需要修改源代码么

使用注册授权系统不一定需要修改源代码。可以不修改源代码添加授权系统。

如果你想使用 Vprotect 提供的授权 API 来编写自己的授权系统，那么你需要修改源代码。

十五:Vprotect 各个版本之间有什么不同

注释：

√：功能受限

√：功能完整

版本|功能	演示版	标准版	专业版	旗舰版	云版本	企业版
虚拟机加密引擎	√	√	√	√	√	√
代码乱序引擎	√	√	√	√	√	√
网络云授权系统	√				√	√
外壳保护引擎	√	√	√	√	√	√
加密引擎 SDK	√	√	√	√	√	√
注册授权系统	√		√	√	√	√
授权系统 SDK	√			√	√	√
命令行模式	√			√	√	√
注册方式	无法注册	锁定硬件	锁定硬件	不锁硬件	不锁硬件	自由注册
购买价格	免费试用	￥700	￥1000	￥1500	￥2500	￥4500
授权有效时间	无	一年	一年	一年	一年	二年
授权续费价格	无	￥300	￥400	￥600	￥800	￥1000
续费有效时间	无	一年	一年	一年	一年	一年

演示版限制如下：

- 虚拟机和乱序引擎只能加密一个流程。
- 不能新建授权方案。
- 无法使用黑名单功能。

Http://www.Vprotect.Net

- 不允许商业使用。

各个版本差别

注释：

√：功能受限

√：功能完整

版本|功能	演示版	标准版	专业版	旗舰版	云版本	企业版
虚拟机加密引擎	√	√	√	√	√	√
代码乱序引擎	√	√	√	√	√	√
网络云授权系统	√				√	√
外壳保护引擎	√	√	√	√	√	√
加密引擎 SDK	√	√	√	√	√	√
注册授权系统	√		√	√	√	√
授权系统 SDK	√			√	√	√
命令行模式	√			√	√	√
注册方式	无法注册	锁定硬件	锁定硬件	不锁硬件	不锁硬件	自由注册
购买价格	免费试用	￥800	￥1000	￥1500	￥2500	￥4500
授权有效时间	无	一年	一年	一年	一年	二年
授权续费价格	无	￥300	￥400	￥600	￥800	￥1000
续费有效时间	无	一年	一年	一年	一年	一年

演示版限制如下：

- 虚拟机和乱序引擎只能加密一个流程。
- 不能新建授权方案和自定义密匙，所有用户演示版生成的注册码通用。
- 无法使用黑名单功能。
- 不允许商业使用。

购买 Vprotect

版本	授权价格	授权有效期	续费价格	续费有效期	立即购买
标准版	￥800	一年	￥300	一年	购买页面
专业版	￥1000	一年	￥400	一年	购买页面
旗舰版	￥1500	一年	￥600	一年	购买页面
云版本	￥2500	一年	￥800	一年	购买页面
企业版	￥4500	二年	￥1000	一年	购买页面
定制版					购买页面

技术支持

如果您对 Vprotect 软件保护系统的使用有任何疑问，请通过下面的联系方式与我们取得联系：

- 官方主页：http://www.VProtect.net
- 技术支持邮箱：support@VProtect.net
- 作者邮箱：cooolie@126.com
- QQ： 1360505490.

BUG 反馈格式

你使用的操作系统版本：*

Vprotect 程序版本*

保护选项*

错误信息

你的联系方式

我们会对您提交的信息严格保密，再次感谢您对我们产品的支持！

致谢

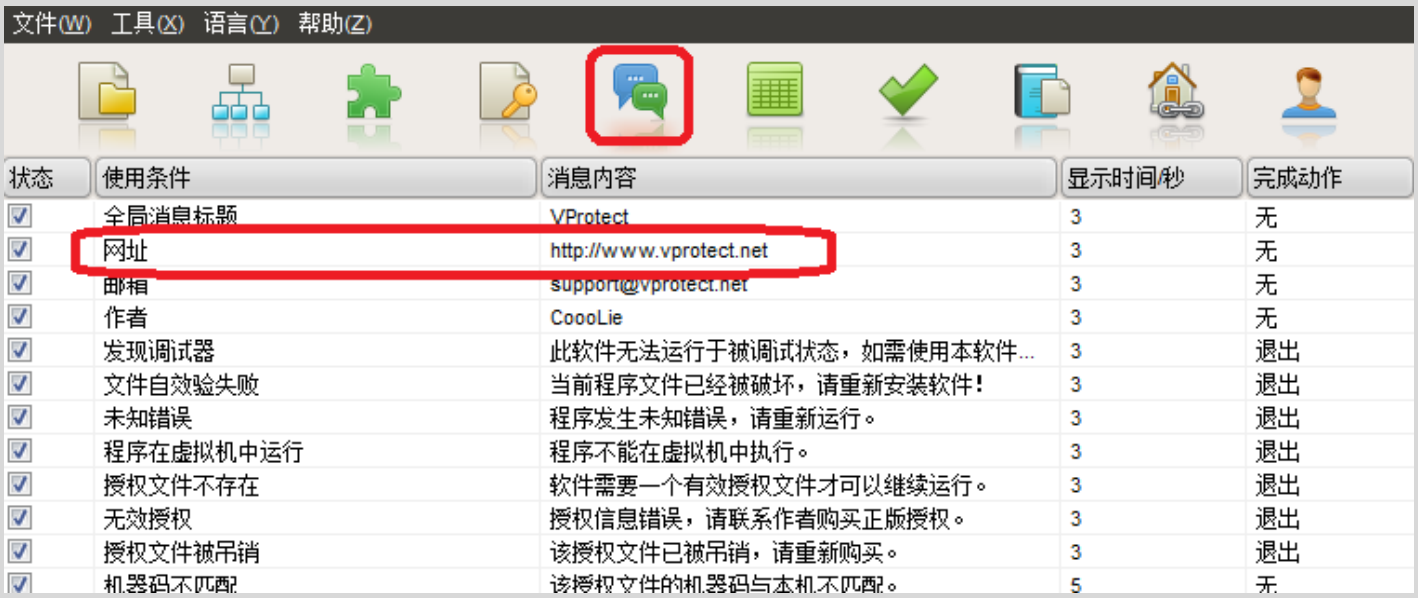
断雪 Kissy starfall hflywolf HyperChem ZeNiX KuNgBiM davis7 wuqing1501FSLove 海风影月 ximo.etc

UPK 论坛极其所有 UPK 会员,和您的使用。

相关文章

修改购买按钮链接到自己的网站

在使用 VProtect 注册授权系统的时候，默认授权对话框会有一个购买按钮，点击后默认打开的是 Vprotect 的官方网站。很多童鞋不知道如何修改，其实很简单如图操作。



Http://www.Vprotect.Net

更新记录

VProtect Version 2.1.0.build 111030(2011-10.30)

[+]添加新的虚拟机自效验方案

[+]云授权系统支持对 DLL 授权的支持

[+]云授权服务端机器码锁定支持删除功能

[!]修正 2003 系统下授权系统无法加载对话框问题

[!]修正外壳引擎对共享区段的支持

[!]修正了 c 无法编译使用授权 Api 问题

VProtect Version 2.0.9.build 110918(2011-09.18)

[+]云授权添加短连接支持

[!]修正机器码锁定在绑定硬盘情况下不稳定问题

[!]修正虚拟机引擎对部分指令符号位扩展错误 bug(thx 轩辕反馈)

[!]增强虚拟机引擎自效验强度(thx kissy)

[*]减少混淆代码提高执行效率

VProtect Version 2.0.8.build 110820(2011-08.20)

[+]云授权版本添加 IP 黑名单管理

[+]云授权版本在线列表添加按字段查找功能

[!]修正特殊情况下第一次加密文件效验不正常问题(thx 最硬的石头反馈)

[!]修正云授权服务端右键拉黑机器码功能无效问题

[*]改善了启动画面功能显示时间优化用户体验

VProtect Version 2.0.7.build 110724(2011-07.24)

[+]新加 VM 多出口技术

[+]云授权添加 IP 黑名单功能

[+]新输入表加密方案提高加密后程序执行效率

[!]更新云授权服务端用户界面

[!]修正虚拟机引擎对边界跳转处理错误问题

[!]修正输入表处理 Bug

VProtect Version 2.0.6.build 110702(2011-07.02)

[+]网络云授权系统公开试用

[+]添加数字签名不影响文件自校验

[+]提高导入表加密兼容性

[+]单机版注册机支持显示自定义值字段

VProtect Version 2.0.5.build 110618(2011-06.18)

[+]添加和网络云授权对接模块

[!]修正重定位分析崩溃 bug

[!]优化大文件加密后执行效率

VProtect Version 2.0.4.build 110505(2011-05.05)

[!]修正部分函数机器码码长度识别错误问题

[!]修正授权窗口文本标签有背景问题

VProtect Version 2.0.3.build 110419(2011-04.19)

[+]更改授权系统对话框为资源方式方便自定义

[!]修正一个文件校验漏洞

VProtect Version 2.0.2.build 110329(2011-03.29)

[+]添加 SDK 对易语言黑月编译模式的支持

[!]修正 DF 标志位处理可能造成虚拟机引擎执行错误问题

[!]修正授权信息过期后提示信息不现实问题

[!]修正授权提示信息自定义不工作问题

VProtect Version 2.0.1.build 110304(2011-03.04)

[+]添加将授权文件保存到注册表选项

[!]更换注册解码算法

[!]修正虚拟机一个指令处理错误

VProtect Version 2.0.0.0(2011-02.24)

[+]更换全新的虚拟机加密引擎，提高强度,效率和兼容性。

[+]添加对手动知道代码加密开始结束位置的支持

[+]添加工程项目保存自定义函数加密信息

[+]添加对注册授权系统 RSA 密钥效验

[!]优化加密后程序体积和执行效率

[*]用户界面小幅调整

VProtect Version 1.9.3.0(2011-02.10)

[+]添加 Dll 补丁检测

[!]修正注册授权系统机器码锁定漏洞（需要重新导出注册机）

[!]修正 IAT 自校验冲突

[!]修正 IAT 加密对 Vs 库函数处理出错问题

VProtect Version 1.9.2.0(2011-01.14)

[!]修正一个虚拟机引擎标志位处理错误

[!]修正 DLL 文件加密后，动态卸载出错问题

[*]增强虚拟机自身效验

VProtect Version 1.9.1.0(2011-01.04)

[+]添加 IAT 代码效验

[+]添加新的授权 API

[!]修正虚拟机检测在 2008r2 系统下兼容性问题

[!]修正主程序在 2003 服务器系统下运行出错问题

[!]修正启动界面显示关闭后，被加密程序窗口不在最前端问题

[!]修正外壳引擎对 Vs 编译器 Debug 模式编译生成的文件加密输入表出错问题

[*]增强 VM 引擎对部分浮点指令处理兼容性

VProtect Version 1.9.0.0(2010-12.23)

[!]修正机器码绑定对硬盘不敏感问题

[!]修正乱序引擎和输入表加密可能出现的兼容性问题

[!]修正输入表加密对 Mov 类型重定位处理 BUG

[!]修正注册机 API 一个可能导致调用短崩溃问题

VProtect Version 1.8.9.0(2010-12.13)

[+]添加反调试器附加

[+]添加 Map 函数搜索功能

[!]修正虚拟机加密对浮点指令的兼容性

[!]修正一个文件效验效验漏洞

VProtect Version 1.8.8.0(2010-12.04)

[+]添加虚拟机关键代码校验功能

[+]添加新的输入表(IAT)加密方案

[+]添加保存 MAP 函数和导入功能（保存在工程文件中）

VProtectVersion1.8.7.0(2010-11.28)

[+]添加新的 ANtiDump 方案

[+]添加 MAP 分析对 BC++程序支持

[!]修正启动密码校验漏洞

[*]增加加密后程序执行效率

VProtect Version 1.8.6.0(2010-11.20)

[+]添加选择硬件绑定项目功能

[!]修正一个 Tls 处理缺陷，可能导致部分 DELPHI 程序加密在 WIN7/Vista 系统下出错问题

VProtect Version 1.8.5.0(2010-11.04)

[+]添加 Visual Basic 加密 SDK 支持

[+]为虚拟机和乱序分别添加新的 SDK 加密标记

[!]修正一个不规则文件标记支持问题

VProtect Version 1.8.4.0(2010-10.25)

[!]增加输入表加密强度

[!]修正一处重定位数据处理缺陷

[!]修正一个由 DEP 导致的内存访问错误

[*]一些细节调整

VProtect Version 1.8.3.0(2010-10.10)

[+]添加新的反调试功能

[+]注册授权管理系统添加查找更新功能

[!]修正虚拟机和乱序引擎对部分分支识别错误问题

[!]修正在 P4 超线程单核超线程情况下机器码可能变化问题（会导致之前授权文件机器码错误，需要更新。）

[*]主程序界面和注册管理系统界面微调，更加人性化

VProtect Version 1.8.2.0(2010-10.01)

[+]添加反内存转储存（ANTI Dump）

[*]修改虚拟机（VM）加密引擎，提高强度

[!]修正一些特殊情况下资源处理 BUG

VProtect Version 1.8.1.0(2010-09.24)

[+]添加命令行模式方便编译器自动化（旗舰版极其以上版本包含该功能）

[+]添加授权系统 API VP_Sdk_GetCustomDword

[+]添加 E 语言授权 API 模块，和使用范例

[*]修正对一些编译器 Debug 模式编译出的程序兼容性问题

VProtect Version 1.8.0.0(2010-09.10)

[+]添加更新授权方案密匙功能

[+]添加对 Typelib or Registry 资源的特殊处理

[+]添加从 MAP 文件分析函数功能

VProtect Version 1.7.8(2010-09.03)

[+]添加文件完整性校验

[*]修正多网卡机器，机器码可能变换问题

[*]修正一个授权系统试用处理逻辑漏洞

VProtect Version 1.7.7(2010-08.27)

[+]添加硬件断点检测

[+]完善注册授权管理系统，添加多语言和一些管理功能

[-]移除一个可能导致兼容性问题的反调试方案

[*]修正添加启动画面图片文件存在中文名时无效问题

[*]修正主程序细节问题如：启用授权方案选择不显眼，选择非 PE 文件可能出错问题

VProtect Version 1.7.6(2010-08.15)

[+]新加设置程序启动界面功能

[!]修正 VP_Sdk_CheckLisenceInMem 忽略注册解码 SDK 问题

[*]增加了加密后程序执行效率，和兼容性

VProtect Version 1.7.5(2010-08.06)

[+]新加内存授权数据检测 API VP_Sdk_CheckLisenceInMem

[+]新加删除授权试用数据

[!]修正特殊情况下 VM 引擎标记位处理 BUG

[*]修正一些不规则 PE 加密兼容性

VProtect Version 1.7.4(2010-07.29)

[+]添加区段融合功能

[!]修正在 32 位 CPU 模拟 64 位运算结果可能不正确的问题

[*]修正一个 IAT 填充 BUG

VProtect Version 1.7.3(2010-07.24)

[!]修正一个资源处理 BUG

[!]修正一个机器码获取安全性问题

（注意：会和之前版本获取的机器码不兼容，如果之前客户使用了机器码锁定则需要从新计算机器码。）

[*]增加乱序随机代码强度

[*]增强 VM 代码多线程安全性

VProtect Version 1.7.2(2010-07.17)

[+]添加资源加密功能

[!]修正一个 IAT 加密 BUG

[*]优化乱序算法减小加密后程序体积

VProtect Version 1.7.1(2010-07.10)

[+]添加授权解码标记

[+]添加注册机 API

[+]添加授权系统 API 和加密标记使用范例

[!]修正虚拟机处理逻辑错误

[!]修正加密选项处理错误

[!]修正 VC 程序浮点指令导致的 R6002 错误

VProtect Version 1.7. 0 (2010-07.04)

[+]添加注册授权系统

[+]添加自定义消息提示功能

[*]增加虚拟机校验强度

VProtect Version 1.6.4(2010-06.15)

[*]继续增强 IAT 加密强度

[!]修正一个内存分配算法 BUG(严重)

[!]修正一个 E 语言 SDK 找不到问题

VProtect Version 1.6.3(2010-06.07)

[*]修改输入表保存，加密，查找方案。提高了效率和强度减小了体积。

[*]修正一个 DLL 保护 BUG

VProtect Version 1.6.2(2010-05.29)

[+]添加乱序加密引擎

[+]添加各个编译器乱序 SDK

[*]增加输入表加密强度

[!]使用新的软件界面

VProtect Version 1.5.1(2010-05.21)

[+]添加压缩文件功能

[+添加 IAT 加密功能

[+]添加反调试，资源校验等功能

[+]添加易语言 SDK

[!]修复反汇编引擎处理 Rox,Shx 指令 BUG

[*]一些其他修改

VProtect Version 0.5.07(2010-05.07)

[+]添加 NEG 指令处理。

[!]修复源操作数是内存地址时，标志位处理不正确问题。

[!]修复处理 Pushad,Popad,Xchg 指令 Bug。

VProtect Version 0.5.03(2010-05.03)

[+]新加重定位处理选择。

[+]添加 C++和 Delphi 编译器 SDK 使用范例。

[!]重新处理进入 Vm 指令。

[!]修正*.exe 文件重定位处理的错误。

[!]修正 Sbb 指令 CF 标记位处理的错误。

VProtect Version 0.4.29(2010-04-29)

[!]重新编写重定位处理模块,解决 VM 指令被重定位后数据覆盖问题。

[!]修正 Shx,RoX 指令处理 BUG。

[+]添加 Leave,Xchg 指令支持。

[+]添加驱动设备程序(*.Sys)支持。

[+]添加多国语言

VProtect Version 0.4.2.4(2010-04-24)

[!]修正一个标志位处理问题

[!]修正 UI 窗口错位问题

[+]合并壳区段至最后一个区段（隐藏壳区段）

[+]反调试功能(稳定)

[+]支持文件拖放操作

[+]添加 BCB 编译器 SDK 文件

[*]VM 入口调用指令随机排序

Version 0.4.0.0(2010-04-20)

[X]发布第一个版本

[Chinese]

[!]Bug 修复

[*]功能加强

[+]增加功能

[-]移除功能