软件加密技术
从入门到精通

史子荣　编著

附光盘

清华大学出版社

软件加密技术
从入门到精通

史子荣　编著

附光盘

清华大学出版社
·北京·

内 容 简 介

本书全面介绍软件加密保护的知识，全书从基本的软件加密基础开始，逐步深入地讲解软件加密的各种技术。全书介绍了注册表和文件的操作知识；讲述如何实现日期限制、次数限制、最后试用期限制、软件启动后的执行时间限制、NAG 窗口限制、各类破解工具的介绍和对工具的应用，以及如何实现对这些工具的反跟踪；介绍如何用硬件系列号、用户名作为加密依据进行注册认证，应用随机数方式进行注册认证，应用 KeyFile 方式进行注册认证，以及如何制作相应的注册机；讲解了 DLL 实现注册认证；Web 服务器方式的网络验证和本地服务器方式的网络验证技术；PE 文件结构的知识。

本书是软件开发人员，特别是共享软件开发人员的重要参考书。

图书在版编目（CIP）数据

软件加密技术从入门到精通 / 史子荣编著. —北京：清华大学出版社，2007.5
ISBN 978-7-302-14960-6

Ⅰ. 软…　Ⅱ. 史…　Ⅲ. 软件－加密　Ⅳ. TP309.7

中国版本图书馆 CIP 数据核字（2007）第 044233 号

责任编辑：夏兆彦　王冰飞
责任校对：张　剑
责任印制：李红英

出版发行：清华大学出版社　　**地　　址**：北京清华大学学研大厦 A 座
http://www.tup.com.cn　　**邮　　编**：100084
c-service@tup.tsinghua.edu.cn
社 总 机：010-62770175　　**邮购热线**：010-62786544
投稿咨询：010-62772015　　**客户服务**：010-62776969
印 刷 者：北京鑫丰华彩印有限公司
装 订 者：三河市溧源装订厂
经　　销：全国新华书店
开　　本：203×260　　**印　张**：16.5　　**字　数**：426 千字
（附光盘 1 张）
版　　次：2007 年 5 月第 1 版　　**印　次**：2007 年 5 月第 1 次印刷
印　　数：1～5000
定　　价：39.00 元

本书如存在文字不清、漏印、缺页、倒页、脱页等印装质量问题，请与清华大学出版社出版部联系调换。
联系电话：010-62770177 转 3103　　产品编号：024358-01

ISBN 7-302-14021-9
定价:35.00元

ISBN 978-7-302-14610-0
定价:35.00元

ISBN 978-7-302-14088-7
定价:49.00元(附光盘1张)

封面设计:子时文化 (010)86390064 ▪ 俞兆君

FOREWORD

前言

越来越多的软件可以轻易地被破解掉，破解后的软件注册机、注册码、破解补丁在网上被公布出来，使软件作者蒙受了巨大的经济损失。造成这种情况的原因主要是软件开发者对软件加密的认识不足或是对软件加密不知道从何入手。他们要么太过于相信成熟的加密算法或是过于相信加密工具，要么没有分清注册码的明码比较和非明码比较，要么把加密算法的使用方法弄错了等等，如此做出来的软件保护方案在解密者手中简直就是不堪一击。

关于本书

本书将从最基本的基础知识开始讲起，逐步深入地介绍如何对软件进行加密，以及软件加密过程中所用到的各种知识，引导软件开发者设计出难以破解的软件保护方案。

本书各章的具体内容安排如下：

第 1 章　基础知识

讲述软件加密过程中用到的各种基础知识，包括注册表的读写操作、INI 文件和自定义文件的读写操作、DLL 文件和 BPL 控件的设计、结构化异常处理、防止应用程序的多个事例运行，以及在 Delphi 嵌入汇编进行编程。

第 2 章　加密算法

先介绍 HASH 算法（CRC32、MD5、SHA 算法）、对称算法（DES、双重 DES、三重 DES、Blowfish、AES 算法）、公开密钥算法（RSA 算法）、其他算法（BASE64 算法）的基本原理，然后用实例演示各种加密算法的实现。

第 3 章　软件试用期

以原理和实例来讲述软件加密过程的试用期设置技术，这些设置技术包括软件试用次数、软件试用天数、软件最后试用日期、限制软件启动后的执行时间、NAG 窗口提示等。

第 4 章　反跟踪技术

先介绍软件破解过程需要用到的各种工具，包括：SoftICE、OllyDbg、FileMon、RegMon、IDA Pro、W32DASM、DeDe，然后介绍检测这些工具的技术以及防脱壳技术、信息隐藏技术等。

第 5 章　注册认证和注册机

介绍软件的注册认证部分，包括如何选用加密算法和校验方式、硬件系列号保护方式、用户名保护方式、硬件系列号与用户名保护方式、随机注册码模式、KeyFile 保护方式、用 DLL 实现注册认证、控件的注册认证方式、完整的软件保护方案实现，同时以实例演示各种技术。

第 6 章　网络验证

以 Web 服务器验证和本地服务器验证来讲述基于网络的注册验证技术，用网络验证技术实现在线升级的验证。

第 7 章　PE 文件知识

讲述 Win32 平台下 PE 文件结构的知识，让读者能洞悉操作系统的秘密。

示例代码

本书是以实际应用作为写作依据的，在书中会出现大量的示例代码，限于篇幅，并没有把所有的代码放到书中，书里只放了关键的部分代码，书中省略代码处以“//...”标记，完整的示例代码可参看随书光盘。

本书全部的示例代码都是用 Delphi 7.0 进行编写的，书中的十六进制是以 Delphi 方式的$表示而不是 0x 表示，使用其他开发语言的读者需要区分一下。

致谢

首先要感谢母校的钱波、藤春芳、吴文虹，另外要感谢在本书写作过程中帮助与支持我的王国彦，最后感谢给予我支持和帮助的所有朋友。

史子荣

2007 年 2 月于昆明

http://www.pefine.com

CONTENTS

目录

第1章 基础知识

在对软件进行加密之前，首先要了解一些在加密过程中经常用到的基础知识：如何将加密信息保存到注册表？如何保存加密信息到文件？在Delphi中如何封装DLL和BPL？如何防止应用程序的多个实例运行？如何在Delphi中嵌入汇编进行编程？本章将详细讲述这些知识。

1.1 注册表知识

注册表是 Windows 的核心数据库，保存着各种硬件、软件的配置信息和参数。Windows 系统及 Windows 系统下的应用程序都可以对注册表进行操作，包括对注册表数据的读取、写入、删除、修改。

1.1.1 注册表结构

注册表按树状进行分类，逐层地往下分，形成根、根主键、主键、子键、值项的分层结构，如同 Windows 的资源管理器一样。注册表的根主键不能被删除，也不可以新建根主键，但可以对根主键下的各项进行删除、修改、增加操作。在“开始”菜单中的“运行”对话框里面输入“regedit”，打开注册表，就可以看到注册表的结构，注册表结构如图 1-1 所示。

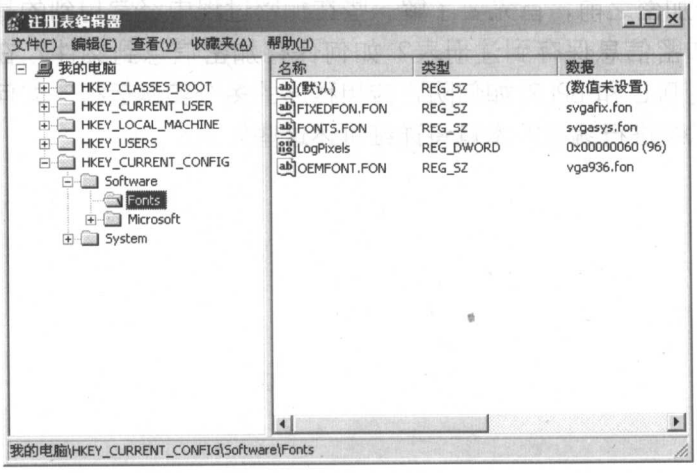

图 1-1 Windows 注册表

各个根主键的含义如下：

1. HKEY_CLASSES_ROOT（简称 HKCR）

HKCR 根主键为系统的每个资源进行归类，这些资源是文件扩展名和 COM 组件的注册信息。

2. HKEY_CURRENT_USER（简称 HKCU）

HKCU 根主键包括了当前用户的一些具体信息，这些信息是当前登录用户相关软件的配置、信息、参数。

3. HKEY_LOCAL_MACHINE（简称 HKLM）

HKLM 根主键里存放着本机的软件、硬件、系统和安全性等信息。

4. HKEY_USERS（简称 HKU）

HKU 根主键里包括了一些针对每个具体用户的有关信息，这些信息是动态加载的用户配置文件和默认配置文件的信息，同时还包含了出现在 HKEY_CURRENT_USER 根主键中的信息。

5. HKEY_CURRENT_CONFIG（简称 HKCC）

HKCC 根主键里存放着当前除了本地配置之外的一些系统配置。

注册表有它自身的数据类型，用于存储不同类型的数据。注册表有存储字符串数据的类型、存储二进制数据的类型、存储十六进制数据的类型等。注册表数据类型及描述见表 1-1。

表 1-1　注册表数据类型及描述

类型	描述
REG_SZ	以空字符结尾的字符串
REG_BINARY	二进制原始数据类型
REG_DWORD	十六进制信息
REG_MULTI_SZ	以空字符结尾的字符串数组，它将以两个字符串结尾
REG_EXPAND_SZ	以空字符结尾的字符串，其中包括对环境变量的未展开的引用

1.1.2　注册表相关函数

可以直接使用 Delphi 封装好的函数来对注册表进行操作，而不用直接使用 API。在使用注册表操作函数之前，应先引用 Registry 单元，该单元封装了对注册表进行操作的各种 API 函数。

常用函数的定义以及功能如下：

```
function OpenKey(const Key: String; Cancreate: Boolean): Boolean;
```

功能：打开指定的主键。

参数：Key 表示主键，Cancreate 表示如果指定的主键不存在时是否创建，True 为创建，False 为不创建。函数调用成功返回 True。当 Cancreate 使用 True 的时候可以省去用 CreateKey 函数来创建主键。

```
procedure WriteString(const Name, Value: string);
```

功能：把一个字符串值写入到指定的名称中。

参数：Name 表示名称，Value 表示要写入的键值。

```
procedure WriteInteger(const Name: string; Value: Integer);
```

功能：把一个整数值写入到指定的名称中。

参数：Name 表示名称，Value 表示要写入的键值。

```
function ReadString(const Name: string): string;
```

功能：从指定的字符串类型的名称中读取键值。

参数：Name 表示键的名称，返回值为键值。

```
function ReadInteger(const Name: string): Integer;
```

功能：从指定的整数类型的名称中读取键值。
参数：Name 表示键的名称，返回值为键值。


```
procedure CloseKey;
```

功能：关闭打开的注册表键。

上面仅仅介绍了在加密过程需要用到的操作注册表的几个函数，如果需要了解更多关于操作注册表的函数的资料可以查阅 Delphi 的联机帮助文件。

1.1.3 注册表读操作

从注册表中读取信息的步骤如下：

（1）先创建 TRegistry 对象。

（2）用 OpenKey 打开想要操作的主键。

（3）用相关读取函数如 ReadString 读取指定名称的键值。

（4）操作完毕以后，用 CloseKey 关闭打开的主键和使用 Destroy 释放创建的对象。

示例代码如下：

```
uses Registry;

procedure TForm1.Button1Click(Sender: TObject);
var
  Reg:TRegistry;
begin
  Reg:=TRegistry.Create;                    //创建 TRegistry 对象
  Reg.RootKey:=HKEY_CURRENT_USER;           //根主键
  if Reg.OpenKey('\Software\Microsoft\Notepad',False) then  //打开指定主键
    Edit1.Text:=Reg.ReadString('lfFaceName');               //读取指定名称的键值
  Reg.CloseKey;                             //关闭打开的键
  Reg.Destroy;                              //释放内存
end;
```

1.1.4 注册表写操作

往注册表中写入信息的步骤如下：

（1）先创建 TRegistry 对象。

（2）用 OpenKey 打开想要操作的主键。

（3）用相关写入函数（如 WriteString）写入指定名称的键值。

（4）操作完毕以后，用 CloseKey 关闭打开的主键并使用 Destroy 释放创建的对象。

示例代码如下：

```
uses Registry;

procedure TForm1.Button2Click(Sender: TObject);
var
  Reg:TRegistry;
begin
  Reg:=TRegistry.Create;                    //创建 TRegistry 对象
  Reg.RootKey:=HKEY_CURRENT_USER;           //根主键
  if Reg.OpenKey('\Software\Microsoft\Notepad',False) then      //打开指定主键
    Reg.WriteString('BitEncrypt',Edit2.Text);  //把 Edit2 的内容写入到 BitEncrypt 中
  Reg.CloseKey;             //关闭打开的键
  Reg.Destroy;              //释放内存
end;
```

1.2 文件读写知识

除了把软件加密信息和用户信息保存到注册表中之外，还可以把这些信息保存到文件中。对不同的文件有不同的操作方法，在这里主要介绍对 INI 文件和自定义文件的操作方法。

1.2.1 INI 文件知识

INI 文件在 Windows 3.1 时就存在，它在系统配置及应用程序参数保存与设置方面具有很重要的作用，它同时也可以在软件加密过程中用于保存加密信息和用户信息。INI 文件的格式是文本文件，它的大小限制为 64KB。

1. INI 文件的结构

```
;注释
[节名]
关键字 1=值
关键字 2=值
…
```

INI 文件就是按上面给出的结构来进行设置的，它同时可以有多个节名，每个节下面又可以有多个关键字。使用“;”可以注释 INI 文件的内容。关键字后面的值有字符串型、整型、布尔型等数据类型。

2. INI 文件相关函数

同样，在操作 INI 的时候可以不直接使用 API 函数，可以直接使用 Delphi 封装好的函数。在使用 INI 文件操作函数之前需要先引用 IniFiles 单元，它对操作 INI 文件的 API 函数进行了封装。

常用函数的定义及功能如下：

```
function ReadString(const Section, Ident, Default: string): string;
```

功能：读取指定节名下面的关键字的值。

参数：Section 表示节名，Ident 表示关键字，Default 表示默认值，该值为当 INI 文件或是节名、关键字不存在时返回的值，这个值可以根据自己的需要来设定。函数调用成功后，返回值为关键字的值（字符串型）。

```
function ReadInteger(const Section, Ident: string;Default: Longint): Longint;
```

功能：读取指定节名下面的关键字的值。

参数：Section 表示节名，Ident 表示关键字，Default 表示默认值，该值为当 INI 文件不存在或是节名、关键字不存在时返回的值，这个值可以根据自己的需要来设定。函数调用成功后，返回值为关键字的值（长整型）。

```
procedure WriteString(const Section, Ident, Value: string);
```

功能：写入指定的字符串信息到 INI 文件中。

参数：Section 表示节名，Ident 表示关键字，Value 表示要写入的值。写入时如果节名不存在，那么会创建一个以 Section 内容命名的节。

```
procedure WriteInteger(const Section, Ident: string; Value: Longint);
```

功能：写入指定的长整型信息到 INI 文件中。

参数：Section 表示节名，Ident 表示关键字，Value 表示要写入的值。写入时如果节名不存在，那么会创建一个以 Section 内容命名的节。

以上介绍的 4 个函数同样也只是在加密过程中经常用得到，需要了解更多的关于操作 INI 文件的函数的资料可以查阅 Delphi 的联机帮助文件。

3. INI 文件的读操作

从 INI 文件中读取信息的步骤如下：

（1）创建一个 TIniFile 对象。

（2）使用 INI 文件读操作函数如 ReadString 读入信息。

（3）操作完毕以后使用 Destroy 释放创建的对象。

示例代码如下：

```
uses IniFiles;

procedure TForm1.Button1Click(Sender: TObject);
var
  IniFileName:String;
  MyIniFile:TIniFile;
begin
  //Ini 文件名和路径
```

```
  IniFileName:=ExtractFilePath(Application.ExeName)+'MyIni.ini';
  MyIniFile:=TIniFile.Create(IniFileName);//创建 TIniFile 对象
  Edit1.Text:=MyIniFile.ReadString('软件加密','加密内容','错误');//读取信息
  MyIniFile.Destroy;//释放对象
end;
```


4. INI 文件的写操作

往 INI 文件中写入信息的步骤如下：

（1）创建一个 TIniFile 对象。

（2）使用 INI 文件写操作函数如 WriteString 写入信息。

（3）操作完毕以后，使用 Destroy 释放创建的对象。

示例代码如下：

```
uses IniFiles;

procedure TForm1.Button2Click(Sender: TObject);
var
  IniFileName:String;
  MyIniFile:TIniFile;
begin
  //Ini 文件名和路径
  IniFileName:=ExtractFilePath(Application.ExeName)+'MyIni.ini';
  MyIniFile:=TIniFile.Create(IniFileName);//创建 TIniFile 对象
  MyIniFile.WriteString('软件加密','技术内容',Edit2.Text);//写入信息
  MyIniFile.Destroy;//释放对象
end;
```

1.2.2 自定义文件知识

在保存加密信息或者使用 KeyFile 方式对程序进行保护的时候，可以按照自己的需要来定义文件内容格式，这样非常方便、灵活。由于开发人员可以按照自己的需要来定义各种各样的文件内容格式，在这里不可能把所有的方式都列出来，所以只以一种简单的方法来讲解如何设计和使用自定义文件。

文件内容格式定义如下：

```
系列号*用户名*注册码
```

文件格式说明：这种方法采用“*”把信息分割开来，从文件读入信息以后，只要以“*”作为分割点进行分割就可以得到各条信息，写入的时候只要把各条信息用“*”连起来写到文件即可。需要注意的是，需要分割的内容里不能含有与分割符相同的字符，选择分割符的时候，要确认分割符不会与需要分割的内容包含的字符相同。在实际应用中，不仅可以采用字符作为分割符，也可以采用自定义的字符串作为分割符。

读入信息并分割信息的示例代码如下：

```
//分割字符串
function GetMsg(RootStr,ChildStr: string): TStringList;
var
  iTemp: integer;
begin
  result := TStringList.Create;
  iTemp := pos(ChildStr,RootStr);
  while iTemp>0 do begin
    if iTemp>1 then result.Append(copy(RootStr,1,iTemp-1));
    delete(RootStr,1,iTemp+length(ChildStr)-1);
    iTemp := pos(ChildStr,RootStr);
  end;
  if RootStr<>'' then result.Append(RootStr);
end;

procedure TForm1.Button1Click(Sender: TObject);
var
  CustFileName:String;
  MsgStr,TempList:TStrings;
begin
  CustFileName:=ExtractFilePath(Application.ExeName)+'Key.lic';
  if FileExists(CustFilename) then
    begin
      MsgStr:=TStringList.Create;              //创建 TStringList 对象
      TempList:=TStringList.Create;
      MsgStr.LoadFromFile(CustFileName);       //读入信息
      TempList:=GetMsg(MsgStr[0],'*');
      //获取各信息
      Edit1.Text:=Templist[0];
      Edit2.Text:=TempList[1];
      Edit3.Text:=TempList[2];
      //释放 TStringList 对象
      MsgStr.Free;
      TempList.Free;
  end;
end;
```

写入信息示例代码如下：

```
procedure TForm1.Button2Click(Sender: TObject);
var
  CustFileName:String;
  MsgStr:TStrings;
begin
  CustFileName:=ExtractFilePath(Application.ExeName)+'Key.lic';
  MsgStr:=TStringList.Create;                  //创建 TStringList 对象
```