利用控件隐藏方法实现数据库的自动加密解密

常新功 （太原 山西财经大学信息系 030006）

摘要：本文就典型数据库应用程序中后台数据库的加密解密问题，提供了一种简便易行的解决办法—控件隐藏方法。

关键词：控件隐藏 数据库 加密 解密

1 问题的提出

一个典型的数据库应用程序从广义上讲应该由后台数据库、前台应用程序、用户（见图1）三部分构成。由于数据库应用程序通常会脱离数据库操作系统独立运行，从而使后台数据库的安全性大打折扣。而前台应用程序可以提供各种安全机制，因此用户应通过前台应用程序访问后台数据库，以确保数据库的安全访问。但道高一尺，魔高一丈，非法用户可利用各种数据库操作系统绕过前台应用程序直接访问后台数据库，从而造成后台数据库中某些机密数据的泄露。

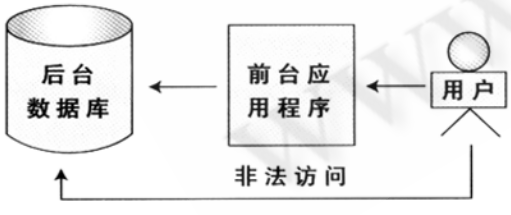

图 1 典型数据库应用程序的组成及访问方式

解决问题的办法应该是对后台数据库中的关键数据进行加密，使用户无法绕过前台应用程序直接从后台数据库中获得机密数据。但由于用户通过应用程序经常要对后台数据库进行增、删、改、查询等操作，这就要求应用程序应该具有对数据库中关键数据的加密解密功能，且这些功能对用户来讲应是透明的，即用户在操作过程中不会感觉到加密解密过程的存在。实现上述想法的常见方法是使用一个中间数据库作为中转。在程序启动时，将后台数据库解密为中间数据库，在程序运行过程中，用户对数据库的增、删、改、查询均是对此中间数据库进行，在程序退出时，再将中间数据库加密为后台数据库。此方法程序代码的工作量太大，效率不高。本文提供另外一种方法——控件隐藏方法，此方法简捷易行，新颖别致，尤其是在控件数目不多时更见高效。

2 利用控件隐藏自动加密解密数据库

VFP应用中，最常见、最快捷的数据库访问方法是将控件直接绑定到数据库。这样，在控件中输入的数据会自动存入绑定的数据库字段中；随着数据库记录指针的移动，当前记录相应字段的值也会在相应的绑定控件中显示出来。因此，我们可用以下方法实现数据库中机密数据的自动加密解密。

假设在后台数据库中，字段F中数据属于机密数据。在程序运行过程中，当用户录入或修改F字段的内容时，程序应对输入的值加密，然后存于后台数据库F字段中；当用户检索或浏览后台数据库F字段的内容时，程序应对F字段中内容解密，然后将解密后的数据显示于屏幕上。为此，我们可在表单中加入两个同类型的和F字段相适应的控件（如F字段为数值型，可选文本框控件；如F字段

为备注类型，可选编辑框控件。以此类推），将其中一个控件的Name属性设置为A，另一个控件的Name属性设置为A1。A1将与F字段直接交互，因此将A1隐藏并绑定到F字段；控件A既不隐藏，也不绑定到任何数据源，它将作为一个输入输出的接口与用户直接交互。

增、删、改、查询是最基本的数据库操作。由于删除操作并不涉及到加、解密问题，故在此不予考虑。当用户要增加或修改F字段的内容时，可将值输入到控件A中，在A的LostFocus事件中对该值加密，并将加密后的值赋于A1，由于A1已绑定到了F字段，加密后的值会直接写入到后台数据库当前记录F字段中；当用户要查询或检索F字段的内容时，由于A1已绑定到了F字段，随着数据库记录指针的移动，A1的值会随之改变。我们所要做的就是把A1中的值解密并在A中显示出来。完成这个操作只要在A1的refresh方法中加入解密代码即可。但要注意一个次序问题，由于A1.refresh执行后，A1中的值才更新。因此在A1的refresh方法代码中，应先执行dodefault()函数（调用父类的refresh方法，使A1中值更新）再解密。

3 一个典型示例

以下是一个常见的员工基本信息管理程序，运行该程序可以对员工的个人信息进行浏览、添加和修改。由于篇幅所限，本文在此作了一些简化。

设其后台数据库及数据库的结构为：

staff.dbf:

姓名 c(10)

工资 n(7,2)

简历 memo

其中，“工资”和“简历”字段属于机密数据，在程序中要对其进行加密解密。

程序界面如图2。

图 2 员工基本信息管理程序

在本程序中，我们可按“添加”按钮向staff.dbf添加新记录，也可直接修改原记录内容，还可按“上一个”和“下一个”按钮浏览数据库内容。

我们可按以下步骤创建该程序：

(1) 创建一个项目 staff.pjx

(2) 在其 Data 选项卡中添加 staff.dbf

(3) 在其 Code 选项卡中添加 nf.prg 和 cf.prg，分别用于对数值型数据和字符型数据加密。

```
nf.prg  && 对数值型数据加密或解密
function nf
lparameters n
#define nkey 12345
return bitxor(n,nkey)
endfunc
cf.prg  && 对字符型数据加密或解密
function cf
lparameters str
deststr=""
local i
j=0
for i=1 to len(str)
  j=(j+1)%10
  deststr=deststr+chr(bitxor(asc(substr(str,i,1)),j*12))
endfor
return deststr
endfunc
```

由于nf()和cf()均是以异或为基本运算的函数，它们自己是自己的逆函数，即nf(nf(x))=x,cf(cf(x))=x，因此它们也可用于对已加密数据的解密。

(4) 在项目管理器 Documents 选项卡中新建表单 staff.scx：

① 适当设置表单的左、右、高、宽，并设置其Caption属性为：

Caption= 员工基本信息管理程序

② 如图2在适当位置加入三个标签对象：“姓名:”、“工资:”、“简历:”

③ 在表单中加入一个文本框，绑定到staff.dbf的“姓名”字段

Name=nam ControlSource=staff. 姓名

④ 由于“工资”字段是机密字段，按照以上所讲方法，我们加入两个文本框控件，一个叫sal，另一个叫sal1，

并将 sal1 隐藏并绑定到 staff. 工资。即

sal1.Visible=.F.　　sal1.ControlSource= staff. 工资

⑤ 由于“简历”字段也是机密字段，因此我们加入两个编辑框控件，一个叫res，另一个叫res1，并将res1隐藏并绑定到 staff. 简历。即

res1.Visible=.F.　　res1.ControlSource= staff. 简历

⑥ 加入四个按钮，其Caption属性分别为：“上一个”、“下一个”、“添加”、“退出”

(5) 编制各控件的事件响应代码

① staff.Load:

```
use staff
```

staff.Destroy:

```
close all
```

② sal.LostFocus:

```
*将用户输入到sal的值加密并赋给sal1的value属性
this.parent.sal1.value=nf(this.value*100)/100
```

sal1.refresh:

*随着后台数据库记录指针的移动，sal1中的值会跟着改变，将其*更新后的值解密并赋给 sal 的 value 属性 dodefault()&& 如不执行此函数，会将上个记录的 sal1.value 解密后赋给本记录的 sal.value，造成程序混乱

```
thisform.sal.value=nf(this.value*100)/100
```

③ res.LostFocus:

```
*将用户输入到res的值加密并赋给res1的value属性
thisform.res1.value=cf(this.value)
```

④ res1.refresh:

```
* 将 res1 的值解密并赋给 res 的 value 属性
dodefault()
thisform.res.value=cf(this.value)
```

⑤ 在按钮“上一个”的 Click 事件中加入代码：

```
* 记录指针前移一条记录
if not bof()
 skip -1
else
 return
endif
thisform.refresh
```

⑥ 在按钮“下一个”的 Click 事件中加入代码：

```
* 记录指针后移一条记录
if !eof()
 skip 1
endif
if eof()
 skip -1
 return
endif
thisform.refresh
```

⑦ 在按钮“添加”的 Click 事件中加入代码：

```
* 添加一条新记录
append blank
go bottom
thisform.refresh
thisform.sal.value=0
thisform.res.value=""
```

⑧ 在按钮“退出”的 Click 事件中加入代码：

```
thisform.release
```

运行上述程序并观察，就会发现虽然在前台程序中我们可以随意地对数据库记录进行录入、添加、修改和浏览，但后台数据库staff.dbf中，“工资”字段和“简历”字段是加了密的。而这一切对用户都是透明的。■

参考文献

1 *康博创作室编著，郗学奎主编《Visual FoxPro 6.0 使用指南》，人民邮电出版社，1999 年 9 月。*

2 *沈惠璋，马英骐，吴继泽编著《深入 Visual FoxPro 6.0 面向对象程序设计》，清华大学出版社，1999 年 10 月。*