10 tips for securing a Microsoft Access database

Version 1.0
February 26, 2009

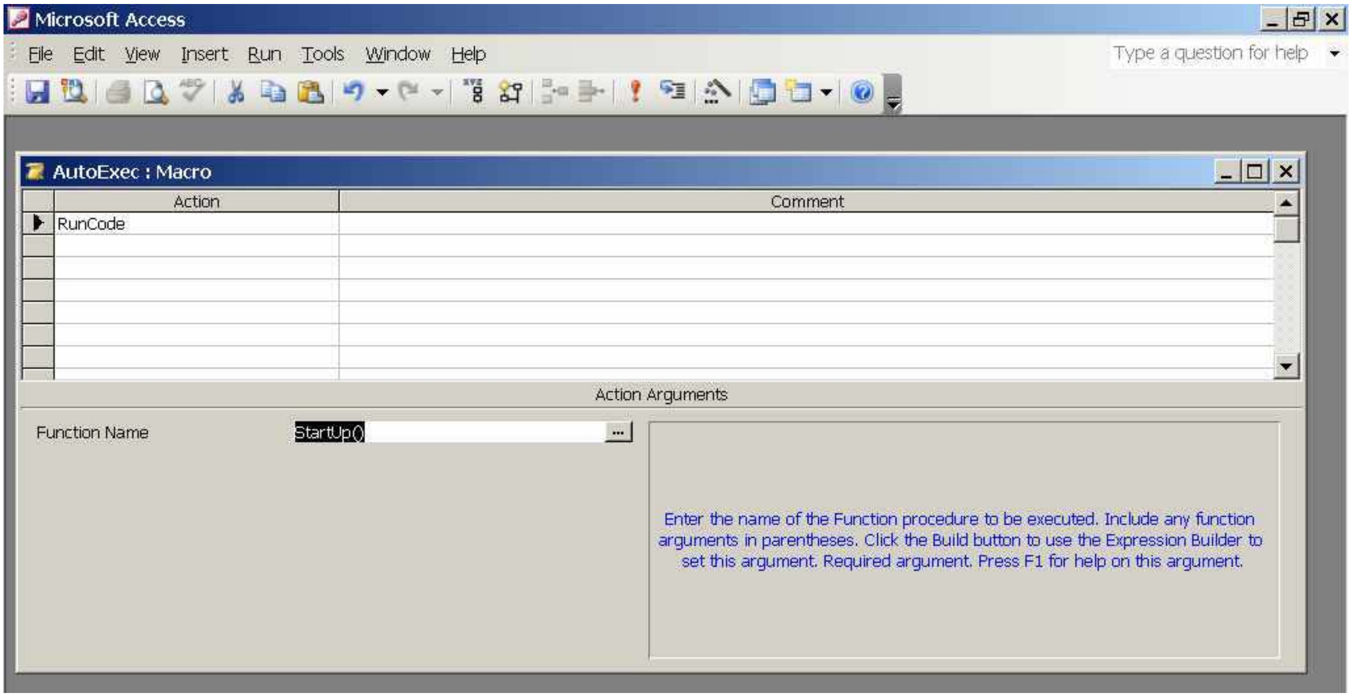
By Susan Sales Harkins

Data is an asset. Therefore, data is money. Even if the data isn't directly involved in the exchange of goods or services, it still has value. That's why it's so important to protect it. The operating system employs the best security, but it's not always practical, especially on a stand-alone system. In the Access world, the next best thing is the user-level model (which Access 2007 doesn't even support). User-level security is complicated and deploying it takes time and special knowledge.

When the best security measures aren't possible (or necessary), you can implement less robust security measures to protect your data and design. Just keep in mind that the following tips prevent accidents by honest users and the mildly curious with enough knowledge to be dangerous. These tips don't offer reliable security, in and of themselves. But by combining a number of them, you can get a level of security that's better than no security at all.

1: Check and reset settings using the AutoExec macro

Use the AutoExec macro to check and reset security options that processes might have changed during the last work session. AutoExec is a special macro that executes when the database opens. To create an AutoExec macro, simply name a new macro AutoExec. For instance, the macro in **Figure A** runs a user-defined function named Startup(), which does the real work of checking and setting security properties before the user can go to work. The macro just executes it.



AutoExec : Macro

Action	Comment
RunCode	

Action Arguments

Function Name: StartUp()

Enter the name of the Function procedure to be executed. Include any function arguments in parentheses. Click the Build button to use the Expression Builder to set this argument. Required argument. Press F1 for help on this argument.

Figure A

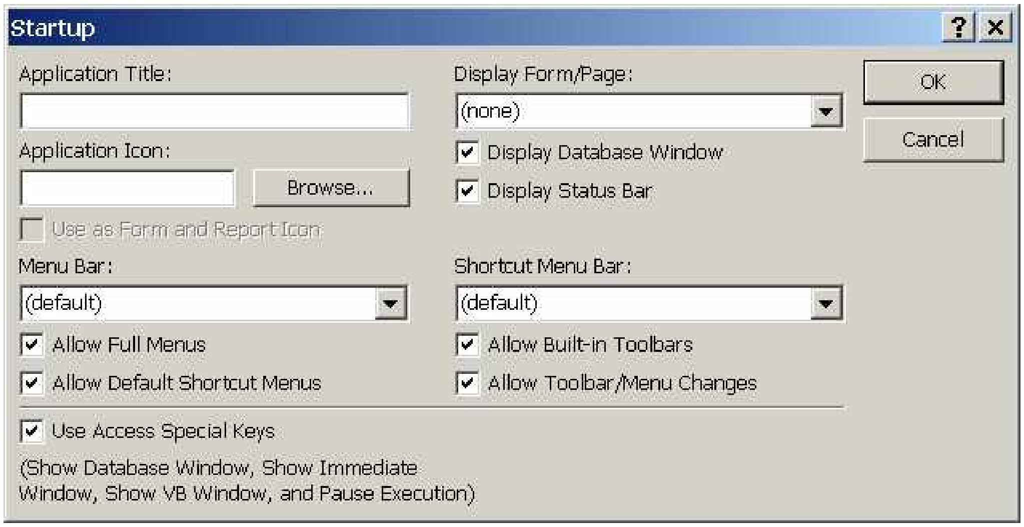
2: Hide the Database window

Startup options, shown in **Figure B**, let you determine specific behaviors when the database opens. Two of these features lend a hand toward securing your database a bit:

- **Display Database Window:** Deselect this option, and the next time someone opens the database, Access will hide the Database window. Users won't have immediate access to any objects.
- **Use Access Special Keys**: Deselect this option to inhibit the use of F11 to unhide the Database Window.

Both settings work together. If you don't deselect the Use Access Special Keys option, users can press F11 to unhide the Database window.

To access the Startup options, choose Startup from the Tools menu. In Access 2007, click the Office button and then click the Access Options button. Select Current Database in the left pane and you'll find these options in the Application Options section. Access 2007 doesn't have a Database window, but you can hide the Navigation Pane in a similar manor. That option is in the Navigation section just below the Application Options section.



Application Title:

Display Form/Page: (none)

Application Icon:

☑ Display Database Window

☑ Display Status Bar

□ Use as Form and Report Icon

Menu Bar: (default)

Shortcut Menu Bar: (default)

☑ Allow Full Menus

☑ Allow Built-in Toolbars

☑ Allow Default Shortcut Menus

☑ Allow Toolbar/Menu Changes

☑ Use Access Special Keys

(Show Database Window, Show Immediate Window, Show VB Window, and Pause Execution)

Figure B

Deselecting the Display Database Window option will also disable the Startup command. Users can bypass all these options by holding down the [Shift] key while opening the database. That trick's handy for you, but leaves the database vulnerable to anyone else who knows about it. A user can also import objects into a blank database to bypass startup settings.

3: Bypass the bypass

You can use the interface to hide the Database window, but the [Shift] key bypass renders the database vulnerable to anyone who knows about it. That's why there's a bypass to the bypass... seriously. To close the bypass crack, set the AllowBypassKey property to False when the database closes. Automate this process by calling the following code from a close task -- just which task is up to you:

```
Public Sub SetStartupOptions(propname As String, _
 propdb As Variant, prop As Variant)
  'Set passed startup property.
  Dim dbs As Object
  Dim prp As Object
  Set dbs = CurrentDb
  On Error Resume Next
  dbs.Properties(propname) = prop
  If Err.Number = 3270 Then
    Set prp = dbs.CreateProperty(propname, _
     propdb, prop)
    dbs.Properties.Append prp
  End If
  Set dbs = Nothing
  Set prp = Nothing
End Sub
```

When you call the procedure, be sure to pass the appropriate startup option text, as follows:

```
Call SetStartupOptions("AllowBypassKey", dbBoolean, False)
```

After setting this property during the close process, the database will ignore the [Shift] key bypass if one of your users is wily enough to try it.

Use this to set any of the startup properties. For instance, this call hides the Database window:

```
Call SetStartupOptions("StartupShowDBWindow", dbBoolean, False)
```

You can set options when you close or open the database with one exception. The AllowBypassKey property must be set when you close the database. Be sure to set a reference to the Data Access Objects library (DAO). Otherwise, this procedure will generate a reference error. (There's bound to be an ADO alternative, but DAO is efficient in this area.)

It makes sense that anyone who knows about the [Shift] key bypass (#2) might also know how to enabling the [Shift] bypass by resetting the AllowBypassKey property to True. If this is the case, you'll have to apply workgroup security to restrict access to this property to the administrator. Someone can try to reset the property, but the effort will fail unless that person is working through the administrator login.

4: Split the database

A split database is easier to protect than a single database that contains the data and the interface objects. By split, I mean having a database that stores tables and relationships in one database, known as the *backend*, and the interface objects in a second database, known as the *front end*. The two databases communicate through linked tables. Here's why all that's important: Users in the front end can't alter the design of tables in the backend. (There are many reasons to split a database, but this discussion is about just security.)

To split a database, choose Database Utilities from the Tools menu. Then, select Database Splitter. The wizard will walk you through the process. In Access 2007, click Access Database in the Move Data group on the Database Tools tab.

5: Avoid Compact On Close

Anybody who uses Access knows that compacting regularly can mean the difference between a successful application and a bomb. Compacting makes a copy of the file, overhauls its objects, deletes temporary data, and rearranges the fragmented pieces on your disk. In short, compacting keeps a database in good working order.

Starting with Access 2000, Access offers the Compact On Close option, which compacts the database automatically when the last person closes it. Unfortunately, the process sometimes forgets to clean up after itself. If you find temporary files, with names like db1.mdb, db2.mdb, and so on, in the same folder as your database, they're most likely a byproduct of the compact feature.

Those leftover files can be a problem. Anyone who has access to the folder has access to the temporary files, and that's a breach in security. There are two ways to protect your database:

- Check regularly and delete any temporary files (but this isn't really a practical or even effective solution).
- Don't use the Compact On Close feature. This is the best way to protect a database from this particular vulnerability. Compact the database manually. You can even train someone to do it.

6: Hide objects—a subtle form of protection

It's a good idea to hide objects -- tables, queries, forms, and so on -- from users. Doing so won't protect these objects in the traditional sense, because if the user can find them, the user can alter them. However, if the user doesn't know the objects exist, chances are the object will be safe enough from users who have no wish to break the database or steal data. Hiding objects simply keeps them safe from the mistakes an honest user might make, without malice or intent. To hide an object in the Database window (or Navigation pane), right-click it and choose Properties. Then, check the Hidden Attribute option.

Now, let me stress one more time (to save you the time of complaining) that someone who understands Access can unhide these objects just as easily as you can hide them. To view all hidden objects, you simply choose Options from the Tools menu, click the View tab, and then select the Hidden Objects option in the Show section. (If you select an object's Hidden attribute, but the Database window still lists it, the Hidden Objects option is probably selected.) In Access 2007, right-click the Navigation pane's menu bar, select Navigation Options, select Show Hidden Objects, and click OK.

As you can see, hiding an object doesn't secure it; it just stores it out of sight (and out of mind). If you use this technique, remember that hidden modules are still visible in the Visual Basic Editor (VBE). In addition, consider hiding only the most important objects. A snoopy user who finds an empty Database window is apt to go looking. You can't import hidden objects, which can be problematic if the import process is legitimate.

You can programmatically hide an object using VBA code, as follows

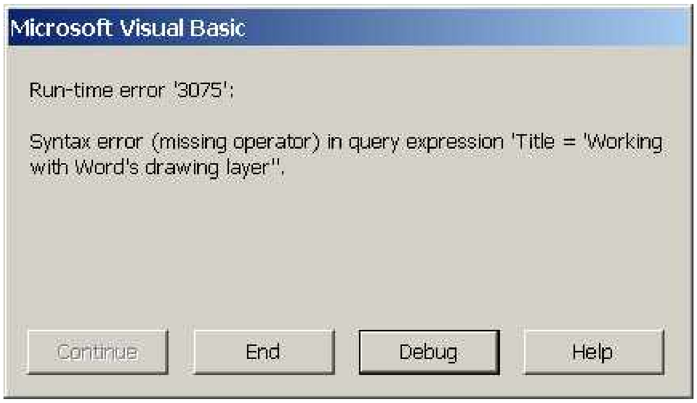
```
CurrentDb.TableDefs(tablename).Attributes = dbHiddenObject
```

In older versions (through 2000), assigning the hidden attribute to a table via code is problematic because Access flags the table as temporary. Then, during the next compact, Access deletes it, along with your data! Avoid this option if you're working with an older version.

7: Use error handling to protect code

When code generates an error, VBA displays an error message similar to the one shown in **Figure C**. If a user gets that form and clicks Debug, he or she will be staring right into the heart of your application -- the module that contains the error-generating code in the VBE. At this point, the user has full access to your code. Most likely, the user won't know what to do and will call you for help. On the other hand, in a panic (or with a little mischief) the user could wipe out all of your code.

During the development stage, the ability to access code quickly is a timesaving feature. In a production database, it's a disaster waiting to happen. As a matter of good practice, all procedures should have some level of error handling to inhibit the generic message and its Debug button.



Run-time error '3075':

Syntax error (missing operator) in query expression 'Title = 'Working with Word's drawing layer''.

Figure C

8: Password-protect the database

A password is like a pin number -- without it, you can't access the database. Once you've password-protected a database, any user who wants access must know the password. There are third-party products that can crack a password-protected database, so this protection isn't foolproof, but it's an adequate tool nonetheless. To password-protect a database, do the following:

1. Open the database in Exclusive mode by choosing Open Exclusive in the Open dialog box, as shown in **Figure D**.

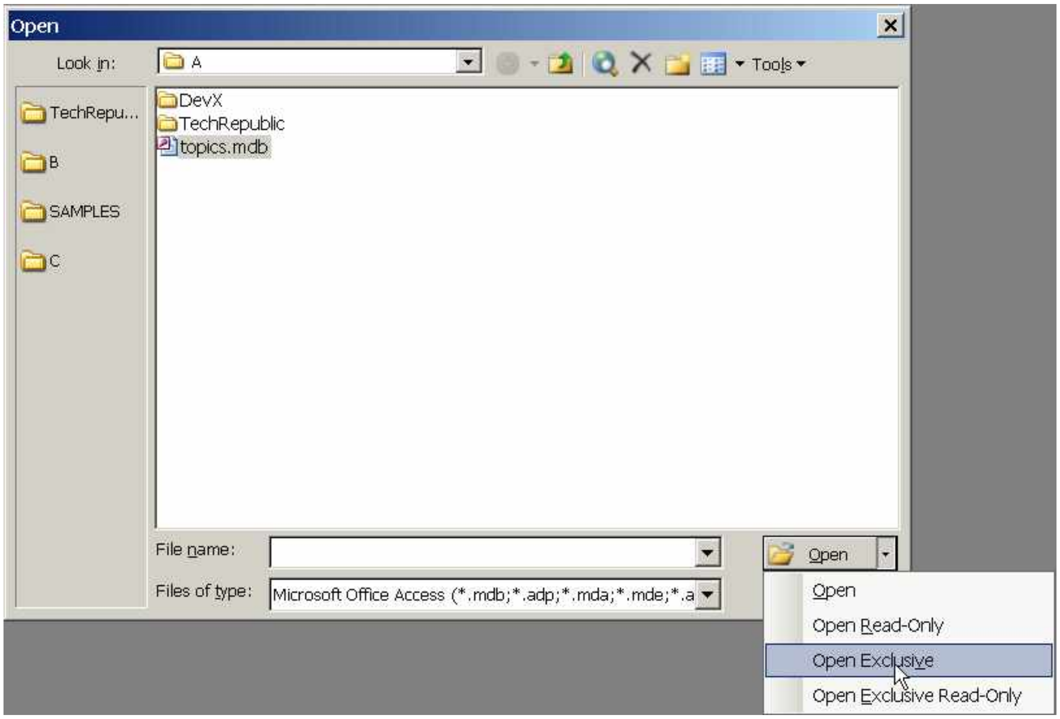


Figure D

2. Choose Security from the Tools menu and then select Set Database Password.
3. Enter the password twice.
4. Click OK.

The owner of the database can remove the password as follows:

1. Open the database in Exclusive mode.
2. Choose Security from the Tools menu and select Unset Database Password.
3. Enter the password.
4. Click OK.

You can also password-protect your VBA modules (code) as follows:

1. Choose *project* Properties from the Tools menu (in the VBE).
2. Click the Protection tab.
3. Check the Lock Project For Viewing option.
4. Fill in the two password fields (using the same password, of course).
5. Click OK.

Password protection is better than no password at all, but it won't stop someone with the right tools and a little time.

9: Convert to mde or accde format

Access offers a security feature in the guise of a file format: mde and accde (in Access 2007). This format is an execute-only version of the database. That means users don't have access to the code via the VBE, nor can they make design changes to objects. This format protects the validity of your design (mostly) but it doesn't protect the data. (Be sure to keep a copy of the original mdb/accdb file for upgrades and other modifications.)

This format has it issues, as you might suspect:

- Use this format for the front end of a split database. Don't use it to secure the backend or a stand-alone database. If you do, you'll have to transfer all the data into a new database every time you upgrade the front end.
- This format doesn't protect tables, queries, macros, relationships, database properties, or startup options. <groan>

To convert a front-end database to the mde or accde format, do the following:

1. In Access XP and earlier, choose Database Utilities from the Tools menu and then select Make MDE File. In Access 2007, click Make ACCDE in the Database Tools group on the Database Tools tab. (You can't convert Access 2000 format or earlier.)

2. In the resulting dialog box, specify a folder and name for the new database and click Save.

10: Password-protect the system

You can't expect users to never leave their system. And while the system's unattended, your database is vulnerable -- especially if users tend to leave the database open during breaks, lunch, and so on. You can set rules, but users forget.

One way to protect against intrusion is to password-protect the system's screensaver feature. Windows' screensaver kicks in when the system sits idle. If you password-protect the feature, the user must enter a login password to regain access to the system. Under Windows XP, you can password-protect the system's screensaver, as follows:

1. Click Start and choose Control Panel.
2. Double-click Display.
3. Click the Screen Saver tab.
4. Select a screensaver (if necessary).
5. Set a Wait time -- that's up to you.
6. Check the On Resume, Password Protect option.
7. Click OK.

After the screensaver kicks in (step 5), a user must enter a login password to regain access to the system. If the database is networked, the administrator can set up this protection for everyone who uses the database, saving you the trouble of visiting individual computers.

Additional resources

- TechRepublic's Downloads RSS Feed XML
- Sign up for the Downloads at TechRepublic newsletter
- Sign up for our IT Leadership Newsletter
- Check out all of TechRepublic's free newsletters
- 10 reasons why IT pros hate Microsoft Access (but really shouldn't)
- 10+ things you should do before building a custom Access database
- 10 ways to prevent Access database corruption

Version history

Version: 1.0
Published: February 26, 2009

Tell us what you think

TechRepublic downloads are designed to help you get your job done as painlessly and effectively as possible. Because we're continually looking for ways to improve the usefulness of these tools, we need your feedback. Please take a minute to drop us a line and tell us how well this download worked for you and offer your suggestions for improvement.

Thanks!

—The TechRepublic Content Team