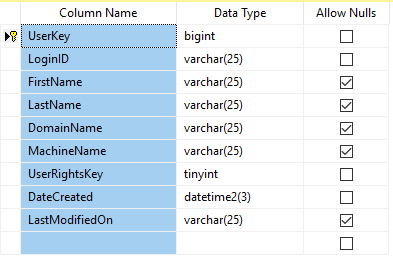
**[User Lever Security in Microsoft Access](http://www.accesssecurityblog.com/post/2011/04/02/Login-Security-using-Access-VBA.aspx)**

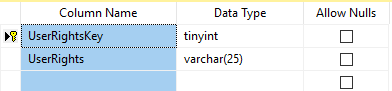
By Assigning default permission on their databases, we can limit and manage the accessibility of data and documents. Restricting permission also helps in making your database more secure and manageable. Permissions can be assigned based on different MDOT groups, rather than individually. As a new user login to your MS Access, he/she will automatically inherit the permissions, which is assigned to his/her designated group. Even though MS Access is not the most secure platform, only users with a valid Windows login ID will have access to open the application. Those users will be assigned different permission levels that will drive security.

Two tables are needed:

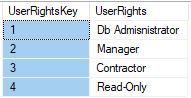
**tblUsers:**



**tblUserRights:**



The second table is a lookup for the UserRightsKey in tblUsers. We will have the role-based security levels as defined below:



New user names and their permission levels are entered or imported via script to table tblUsers.

When a user opens the Access database, the first task the code performs is to read his Windows login ID, Domain Name and Computer name. Then it queries the Users table. If his login ID is found, his permission level is read from table. If the user login ID is not found, he is automatically assigned the Read-Only permission level. In either case, the switchboard form will be displayed.

The Access Level read found in the User table is then used to restrict access to various parts of the application. This is done by checking the Access level using the following expression:

?

|  |  |
| --- | --- |
| 1 | userPermissionLevel = DLookup("[UserRightsKey]", "tblUsers", "[LoginID] = " & Chr(39) & strUserName & Chr(39)) |

Once the Access level is retrieved, it can be used to restrict access to forms and controls based on the access level assigned to the user. Following are some examples:

Case 1: Restricting access to a form only to administrators  
The following code should be placed In the On Open event of the form.

?

|  |  |
| --- | --- |
| 1  2  3  4 | If userPermissionLevel <> 1 Then      MsgBox “You are not authorized to open this form!”, vbOkOnly +vbExclamation      Cancel = True  End If |

Case 2: Making a form read-only for Readers  
The following code should be placed In the On Open event of the form.

[?](http://www.accesssecurityblog.com/post/2011/04/02/Login-Security-using-Access-VBA.aspx)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | If userPermissionLevel = 4 Then      Me.AllowEdits = False      Me.AllowAdditions = False      Me.AllowDeletes = False  Else      Me.AllowEdits = True      Me.AllowAdditions = True      Me.AllowDeletes = True  End If |

Case 3: Restricting user input in form controls. Command buttons, combo boxes and check boxes can be locked.  
The following code should be placed In the On Open event of the form.

1 If lockYesNo = False Then ' Even when the report is 2unlocked, some fields must always be locked ' Add them for each report

3 Select Case currForm.Name

4 Case "frmPropertyInfo"

5 currForm.Controls("McId").Locked = True

6 currForm.Controls("Property Name").Locked = True

7 currForm.Controls("TBU Acronym").Locked = True

8 Case "frmReports"

9 currForm.Controls("tblMCMain.McId").Locked = True

10 currForm.Controls("Property Name").Locked = True

11 currForm.Controls("Acreage (SDAT)").Locked = True

12 currForm.Controls("Acreage (Plat)").Locked = True

13 currForm.Controls("Zoning Category").Locked = True

14 currForm.Controls("Zoning Source").Locked = True

15 End Select

16 End If

These are just three examples. There are different ways to use these techniques to restrict what a user can access in the application.

Implementation:

When the switchboard form opens, the Form\_Open event calls the procedure below to obtain the permission level for the user.

Private Sub GetUserPermissionLevel()

Dim strUserName As String ' ULS

Dim strDomainName As String ' ULS

Dim dbDomainName As String ' ULS

Dim dbMachineName As String ' ULS

Dim strMachineName As String ' ULS

Dim SQL As String ' ULS

Dim userPermissionLevel As Integer ' ULS

strUserName = Environ("USERNAME") ' ULS

strDomainName = Environ("USERDOMAIN") ' ULS

strMachineName = Environ("COMPUTERNAME") ' ULS

If IsNull(DLookup("[LoginID]", "tblUsers", "[LoginID] = " & Chr(39) & strUserName & Chr(39))) Then

userPermissionLevel = 4 ' 4 = Read-Only

SQL = "INSERT INTO tblUsers (LoginID, DomainName, MachineName, UserRightsKey)" & \_

" VALUES (" & Chr(39) & strUserName & Chr(39) & "," & Chr(39) & strDomainName & Chr(39) & "," & \_

Chr(39) & strMachineName & Chr(39) & "," & userPermissionLevel & ");"

DoCmd.SetWarnings False

DoCmd.RunSQL SQL

DoCmd.SetWarnings True

Else

userPermissionLevel = DLookup("[UserRightsKey]", "tblUsers", "[LoginID] = " & Chr(39) & strUserName & Chr(39))

If IsNull(DLookup("[DomainName]", "tblUsers", "[LoginID] = " & Chr(39) & strUserName & Chr(39))) Then ' check if Domain Name is null

dbDomainName = vbNullString

Else

dbDomainName = DLookup("[DomainName]", "tblUsers", "[LoginID] = " & Chr(39) & strUserName & Chr(39))

End If

If IsNull(DLookup("[MachineName]", "tblUsers", "[LoginID] = " & Chr(39) & strUserName & Chr(39))) Then ' check if Machine Name is null

dbMachineName = vbNullString

Else

dbMachineName = DLookup("[MachineName]", "tblUsers", "[LoginID] = " & Chr(39) & strUserName & Chr(39))

End If

If dbDomainName = strDomainName And dbMachineName = strMachineName Then

' Update nothing

Else

SQL = "UPDATE tblUsers" & \_

" SET tblUsers.DomainName =" & Chr(39) & strDomainName & Chr(39) & "," & \_

" tblUsers.MachineName =" & Chr(39) & strMachineName & Chr(39) & "," & \_

" tblUsers.LastModifiedOn =" & Chr(39) & Format(Now(), "yyyy-mm-dd hh:mm:ss") & Chr(39) & \_

" WHERE tblUsers.LoginID =" & Chr(39) & strUserName & Chr(39) & ";"

DoCmd.SetWarnings False

DoCmd.RunSQL SQL

DoCmd.SetWarnings True

End If

End If

End Sub