Listing Program

```
Option Explicit

Private m_Rijndael As New cRijndael
Dim Time As Long
Dim mdetik As Long
Dim a As Long
Dim b As Long
Dim c As Long
Private Sub DisplayString(TheTextBox As  TextBox, ByVal TheString As String)
   If Len(TheString) < 49999 Then
      txtSend.Text = TheString
   Else
      MsgBox "Can not assign a String larger than 49k " & vbCrLf & _
           "to the Text property of a TextBox control." & vbCrLf & _
           "If you need to support Strings longer than 49k," & vbCrLf & _
           "you can use a RichTextBox control instead.", vbInformation
   End If
End Sub
Private Sub DisplayString2(TheTextBox As TextBox, ByVal TheString As String)
      Text1.Text = TheString
End Sub
Private Function HexDisplay(data() As Byte, n As Long, k As Long) As String
   Dim i As Long
   Dim j As Long
   Dim c As Long
   Dim data2() As Byte

   If LBound(data) = 0 Then
      ReDim data2(n * 4 - 1 + ((n - 1) \ k) * 4)
      j = 0
      For i = 0 To n - 1
         If i Mod k = 0 Then
            If i <> 0 Then
               data2(j) = 32
               data2(j + 2) = 32
               j = j + 4
            End If
         End If
         c = data(i) \ 16&
         If c < 10 Then
            data2(j) = c + 48
         Else
            data2(j) = c + 55
         End If
         c = data(i) And 15&
         If c < 10 Then
            data2(j + 2) = c + 48
```

```
        Else
            data2(j + 2) = c + 55
        End If
        j = j + 4
    Next i
Debug.Assert j = UBound(data2) + 1
    HexDisplay = data2
  End If

End Function

Private Function HexDisplayRev(TheString As String, data() As Byte) As Long
  Dim i As Long
  Dim j As Long
  Dim c As Long
  Dim d As Long
  Dim n As Long
  Dim data2() As Byte

  n = 2 * Len(TheString)
  data2 = TheString

  ReDim data(n \ 4 - 1)

  d = 0
  i = 0
  j = 0
  Do While j < n
    c = data2(j)
    Select Case c
    Case 48 To 57
      If d = 0 Then
        d = c
      Else
        data(i) = (c - 48) Or ((d - 48) * 16&)
        i = i + 1
        d = 0
      End If
    Case 65 To 70
      If d = 0 Then
        d = c - 7
      Else
        data(i) = (c - 55) Or ((d - 48) * 16&)
        i = i + 1
        d = 0
      End If
    Case 97 To 102
      If d = 0 Then
```

```
            d = c - 39
        Else
            data(i) = (c - 87) Or ((d - 48) * 16&)
            i = i + 1
            d = 0
        End If
    End Select
    j = j + 2
  Loop
  n = i
  If n = 0 Then
    Erase data
  Else
    ReDim Preserve data(n - 1)
  End If
  HexDisplayRev = n
End Function

Private Function GetPassword() As Byte()
  Dim data() As Byte

    data = StrConv(txtpass.Text, vbFromUnicode)
    ReDim Preserve data(31)

  GetPassword = data
End Function

Private Sub cmdConnect_Click()
If Len(Trim$(txtHost.Text)) = 0 Then
  MsgBox "Enter a host to connect to", vbCritical, "Host Require d"
  txtHost.SetFocus
  Exit Sub
End If
  SaveSetting sApp, "Main", "Host", EncryptHex(StrReverse$(txtHost.Text), True)
  SessionInfo.RemoteHost = txtHost.Text
  SessionInfo.FT_InProgress = False
  SessionInfo.Connected = False
  SessionInfo.Nick = txtNick.Text
  SessionInfo.SessionType = SESSION_CLIENT
  sckServer.Close
  sckClient.Close
  sckRec.Close
  sckSend.Close
  sckClient.Connect txtHost.Text, 7802
  StatusBar.SimpleText = "Status : Connecting to User . . ."
End Sub

Private Sub cmdDisco_Click()
```

```
If Not SessionInfo.Connected Then
   MsgBox "You are not currently connected", vbCritical, "No Present Connection"
Else
   If SessionInfo.SessionType = SESSION_SERVER Then
      sckServer.Close
      SessionInfo.Connected = False
      StatusBar.SimpleText = "Status : Disconnected."
      Call AddRTFStatus("You have closed the connection.", RGB(123, 0, 0))
      sckRec.Close
      sckSend.Close
      SessionInfo.FT_InProgress = False
   ElseIf SessionInfo.SessionType = SESSION_CLIENT Then
      sckClient.Close
      SessionInfo.Connected = False
      StatusBar.SimpleText = "Status : Disconnected."
      Call AddRTFStatus("You have closed the connection.", RGB(123, 0, 0))
      sckRec.Close
      sckSend.Close
      SessionInfo.FT_InProgress = False
   End If
End If
End Sub

Private Sub cmdSend_Click()
On Error Resume Next
If Len(txtSend.Text) > 0 Then
   If SessionInfo.Connected = False Then
      Call AddRTFStatus("No Connection Is Present.", RGB(123, 0, 0))
      Exit Sub
   End If

   If SessionInfo.SessionType = SESSION_SERVER Then
      sckServer.SendData sHeader & sDelim & "M" & sDelim & txtNick.Text & sDelim &
txtSend.Text
   ElseIf SessionInfo.SessionType = SESSION_CLIENT Then
      sckClient.SendData sHeader & sDelim & "M" & sDelim & txtNick.Text & sDelim &
txtSend.Text
   End If
   With txtChat
      If Len(.Text) = 0 Then
         .SelBold = True
         .SelItalic = False
         .SelUnderline = False
         .SelColor = RGB(0, 0, 123)
         .SelText = txtNick.Text
         .SelBold = False
         .SelColor = vbBlack
         .SelText = ": "
```

```
        .SelColor = RGB(0, 0, 123)
        .SelText = txtSend.Text
      Else
        .SelBold = True
        .SelItalic = False
        .SelUnderline = False
        .SelColor = RGB(0, 0, 123)
        .SelText = vbNewLine & txtNick.Text
        .SelBold = False
        .SelColor = vbBlack
        .SelText = ": "
        .SelColor = RGB(0, 0, 123)
        .SelText = txtSend.Text
      End If
   End With
   txtSend.Text = Empty
   txtSend.SetFocus
End If

End Sub


Private Sub enc_Click()
   Dim pass()       As Byte
   Dim plaintext()   As Byte
   Dim ciphertext()  As Byte
   Dim KeyBits      As Long
   Dim BlockBits     As Long
   If Len(txtSend.Text) = 0 Then
      MsgBox "No Plaintext"
   Else
      If Len(txtpass.Text) = 0 Then
         MsgBox "No Password"
      Else
a = Text5.Text
Text6.Text = a
b = a

        KeyBits = Combo1.ItemData(Combo1.ListIndex)
        BlockBits = 128
        pass = GetPassword

        plaintext = StrConv(txtSend.Text, vbFromUnicode)

        m_Rijndael.SetCipherKey pass, KeyBits, BlockBits
        m_Rijndael.ArrayEncrypt plaintext, ciphertext, 0, BlockBits

        DisplayString txtSend, HexDisplay(ciphertext, UBound(ciphertext) + 1, BlockBits  \ 8)
```

```
If st.Value = 1 Then
c = b
Text7.Text = GetTickCount() - a
mdetik = GetTickCount() - c
Text5.Text = GetTickCount()
MsgBox ("Kecepatan enkripsi plaintext tsb pa da PC anda = " & mdetik & " millisecond")
Text6.Text = 0
Text7.Text = 0
c = 0
End If
      End If
   End If
End Sub

Private Sub EncFile_Click()
   Dim FileName  As String
   Dim FileName2 As String
   Dim pass()    As Byte
   Dim KeyBits   As Long
   Dim BlockBits As Long

   If Len(txtpass.Text) = 0 Then
      MsgBox "No Password"
   Else
      FileName = FileDialog(Me, False, "File to Encrypt", "*.*|*.*")
      If Len(FileName) <> 0 Then
         FileName2 = FileDialog(Me, True, "Save  Encrypted Data As ...", "*.sie|*.sie|*.*|*.*",
FileName & ".sie")
         If Len(FileName2) <> 0 Then
a = Text5.Text
Text6.Text = a
b = a
            RidFile FileName2
            KeyBits = Combo1.ItemData(Combo1.ListIndex)
            BlockBits = 128
            pass = GetPassword

            m_Rijndael.SetCipherKey pass, KeyBits, BlockBits
            m_Rijndael.FileEncrypt FileName, FileName2, BlockBits
If st.Value = 1 Then
c = b
Text7.Text = GetTickCount() - a
mdetik = GetTickCount() - c
Text5.Text = GetTickCount()
MsgBox ("Kecepatan enkripsi file tsb pada PC anda = " & mdetik & " millisecond")
End If
         End If
         End If
```

```
        End If

End Sub

Private Sub DecFile_Click()
Dim FileName  As String
   Dim FileName2 As String
   Dim pass()    As Byte
   Dim KeyBits   As Long
   Dim BlockBits As Long

   If Len(txtpass.Text) = 0 Then
      MsgBox "No Password"
   Else
      FileName = FileDialog(Me, False, "File to D ecrypt", "*.sie|*.sie|*.*|*.*")
      If Len(FileName) <> 0 Then
         If InStrRev(FileName, ".sie") = Len(FileName)  - 3 Then FileName2 = Left$(FileName,
Len(FileName) - 4)
         FileName2 = FileDialog(Me, True, "Save Decrypted Data As ...", "*.*|*.*", FileName2)
         If Len(FileName2) <> 0 Then
a = Text5.Text
Text6.Text = a
b = a
            RidFile FileName2
            KeyBits = Combo1.ItemData(Combo1. ListIndex)
            BlockBits = 128
            pass = GetPassword

            m_Rijndael.SetCipherKey pass, KeyBits, BlockBits
            m_Rijndael.FileDecrypt FileName2, FileName, BlockBits
If st.Value = 1 Then
c = b
Text7.Text = GetTickCount() - a
mdetik = GetTickCount() - c
Text5.Text = GetTickCount()
MsgBox ("Kecepatan dekripsi file tsb pada PC anda = " & mdetik & " millisecond")
End If
         End If
         End If
         End If
End Sub

Private Sub FileSend_Click()
If SessionInfo.FT_InProgress Then
   MsgBox "File transfer is already in progress; please wait for it to finish", vbCritical, "File
Transfer in Progress"
   Exit Sub
Else
```

```
   frmSendFile.Show , Me
   frmSendFile.txtPath.Text = Empty: frmSendFile.lblKBPS.Caption = "0 KB/Sec":
frmSendFile.lblBS.Caption = "Bytes Sent : 0": frmSendFile.StatusBar.SimpleText = "Status :
Idle."
End If
End Sub

Private Sub decClear_Click()
Timer1.Enabled = False
Text1.Text = Empty
txtSend.Text = Empty
Timer1.Enabled = True
End Sub

Private Sub Form_Load()
Skin1.LoadSkin (App.Path + "\" + "11.SKN")
Skin1.ApplySkin Me.hwnd
FileNum = FreeFile
Call MakeRecDir
On Error Resume Next
txtHost.Text = StrReverse$(EncryptHex(GetSetting (sApp, "Main", "Host", ""), False))
txtNick.Text = StrReverse$(EncryptHex(GetSetting(sApp, "Main", "Nick", ""), False))

If Len(Trim$(txtHost.Text)) = 0 Then
   cmdConnect.Enabled = False
Else
   cmdConnect.Enabled = True
End If

If Len(txtNick.Text) = 0 Then
   cmdSend.Enabled = False
Else
   cmdSend.Enabled = True
End If
On Error Resume Next
sckServer.Listen
SessionInfo.SessionType = SESSION_SERVER
End Sub

Private Sub Form_QueryUnload(Cancel As Integer, UnloadMode As Integer)
   If Len(txtNick.Text) > 0 Then
      SaveSetting sApp, "Main", "Nick", EncryptHex(StrReverse$(txtNick.Text), True)
   End If
   On Error Resume Next
   Unload frmRecFile
   Unload frmSendFile
   Unload Me
End Sub
```

```
Private Sub mnuCLS_Click()
txtChat.Text = Empty
End Sub

Private Sub mnuExit_Click()
If Len(txtNick.Text) > 0 Then
   SaveSetting sApp, "Main", "Nick", EncryptHex(StrReverse$(txtNick.Text), True)
End If
On Error Resume Next
Unload frmRecFile
Unload frmSendFile
Unload Me
End Sub

Private Sub mnuSaveConvo_Click()
   With CD
   .DialogTitle = "Save Conversation"
   .Filter = "Rich Text Files|*.rtf|Text Files|*.txt"
   .ShowSave
   If Len(.FileName) > 0 Then
      Dim iFile As Integer
      iFile = GetFileType(.FileName)
         If iFile = FILETYPE_RTF Then
            txtChat.SaveFile .FileName, rtfRTF
         Else
            txtChat.SaveFile .FileName, rtfText
         End If
End If
End With
End Sub


Private Sub sckClient_Close()
   On Error Resume Next
   Unload frmRecFile
   Unload frmSendFile
   SessionInfo.Connected = False
   FrameConvo.Caption = " Conversation (Not Started) "
   SessionInfo.SessionType = SESSION_SERVER
   sckServer.Close
   SessionInfo.FT_InProgress = False
   sckRec.Close
   sckSend.Close
   sckClient.Close
   sckServer.Listen
   StatusBar.SimpleText = "Status : Connection to User Was Closed / Lost."
```

```
    Call AddRTFStatus(sckClient.RemoteHostIP & " Has Left the Conversation.", RGB(123, 0, 0))
End Sub

Private Sub sckClient_Connect()
    SessionInfo.Connected = True
    StatusBar.SimpleText = "Status : Awaiting Authorization . . ."
End Sub

Private Sub sckClient_DataArrival(ByVal bytesTotal As Long)
    Dim sData As String: sData = Empty
    sckClient.GetData sData
    Debug.Print sData

    Select Case Mid(sData, 19, 1)

    Case "S"
    Call ParseSessionReply(sData)
    Case "M"
    Call ParseMessage(sData)
    Case "F"
    Call ParseTransferRequest(sData)
    Case "R"
    Call ParseTransferReply(sData)
    Case "D"
    Call ParseReadySignal(sData)
    Case "C"
    Call ParseTransferCancel(sData)
    Case "W"
    Call ParseWhoIsRequest(sData)
    End Select



End Sub

Private Sub sckClient_Error(ByVal Number As Integer, Description As String, ByVal Scode As
Long, ByVal Source As String, ByVal HelpFile As String, ByVal HelpContext As Long,
CancelDisplay As Boolean)
    SessionInfo.Connected = False
    SessionInfo.SessionType = SESSION_SERVER
    SessionInfo.FT_InProgress = False
    sckRec.Close
    sckSend.Close
    sckClient.Close
    sckServer.Close
    On Error Resume Next
    sckServer.Listen
    StatusBar.SimpleText = "Status : Unable to Connect to User."
```

```
End Sub

Private Sub sckRec_Close()
   Close #FileNum
   TotalByteNow = 0
   frmRecFile.tmrDownload.Enabled = False
   SessionInfo.FT_InProgress = False

   If KeepLog = True Then
      Call AddLog("File Received " & Chr$(34) & RecFileInfo.FileName & Chr$(34) & " (" &
RecFileInfo.FileSize & " Bytes) at " & Now)
   End If

   If IsCancel = True Then
   Call ResetRecForm
   frmRecFile.StatusBar.SimpleText = "Status : File Transfer Canceled."
   IsCancel = False
   Exit Sub
   Else

   frmRecFile.Bar.Value = 100
   frmRecFile.lblKBPS.Caption = "0 KB/Sec"
   frmRecFile.lblBR.Caption = "Bytes Received : " & RecFileInfo.FileSize
   frmRecFile.StatusBar.SimpleText = "Status : File Transfer Complete."
   frmRecFile.Hide
   End If
End Sub

Private Sub sckRec_Connect()
   Call MakeRecDir
   Open App.Path & "\Received Files\" & RecFileInfo.FileName For Binary Access Write As
#FileNum
   SessionInfo.FT_InProgress = True
   frmRecFile.Bar.Max = RecFileInfo.FileSize
   frmRecFile.StatusBar.SimpleText = "Status : Receiving File . . ."
End Sub

Private Sub sckRec_DataArrival(ByVal bytesTotal As Long)
   On Error Resume Next
   Dim FileData As String: FileData = Empty
   frmRecFile.tmrDownload.Enabled = True
   frmRecFile.Bar.Value = frmRecFile.Bar.Value + bytesTotal
   frmRecFile.lblBR.Caption = "Bytes Received : " & bytesTotal
   sckRec.GetData FileData
   TotalByteNow = TotalByteNow + bytesTotal
   Put #FileNum, , FileData

End Sub
```

```
Private Sub sckRec_Error(ByVal Number As Integer, Description As String, ByVal S code As Long, ByVal Source As String, ByVal HelpFile As String, ByVal HelpContext As Long, CancelDisplay As Boolean)
   SessionInfo.FT_InProgress = False
   frmRecFile.tmrDownload.Enabled = False
   frmRecFile.StatusBar.SimpleText = "Status : Unable to  Connect to User."
End Sub

Private Sub sckSend_Close()
   sckSend.Close
   frmSendFile.tmrUpload.Enabled = False
   Call ResetSendForm
   frmSendFile.StatusBar.SimpleText = "Status : File Transfer Canceled."

   If KeepLog = True Then
      Call AddLog("File Sent " & Chr$(34) & SendFileInfo.FileName & Chr$(34) & " (" & SendFileInfo.FileSize & " Bytes) at " & Now)
   End If

   On Error Resume Next
   sckSend.Listen

End Sub

Private Sub sckSend_ConnectionRequest(ByVal requestID  As Long)
   SendTotal = 0
   sckSend.Close
   sckSend.Accept requestID
   frmSendFile.StatusBar.SimpleText = "Status : Sending File . . ."
   frmSendFile.Bar.Max = SendFileInfo.FileSize
   Call SendFile(SendFileInfo.FileSource)

End Sub

Private Sub sckSend_Error(ByVal Number As Integer, Description As String, ByVal Scode As Long, ByVal Source As String, ByVal HelpFile As String, ByVal HelpContext As Long, CancelDisplay As Boolean)
   SessionInfo.FT_InProgress = False
End Sub

Private Sub sckSend_SendProgress(ByVal bytesSent As Long, ByVal bytesRemaining As Long)
SendTotal = SendTotal + bytesSent
On Error Resume Next
frmSendFile.Bar.Value = frmSendFile.Bar.Value + bytesSent
frmSendFile.lblBS.Caption = "Bytes Sent : " & bytesSent
If SendTotal >= SendFileInfo.FileSize Then
frmSendFile.Bar.Value = 100
```

```
frmSendFile.lblKBPS.Caption = "0 KB/Sec"
frmSendFile.lblBS.Caption = "Bytes Sent : " & SendFileInfo.FileSize
frmSendFile.StatusBar.SimpleText = "Status : File Transfer Complete."
sckSend.Close
CurByte = 0
SendTotal = 0
SessionInfo.FT_InProgress = False
frmSendFile.Hide
End If
End Sub

Private Sub sckServer_Close()
On Error Resume Next
sckServer.Close
Unload frmRecFile
Unload frmSendFile
SessionInfo.Connected = False
FrameConvo.Caption = " Conversation ( Not Started) "
StatusBar.SimpleText = "Status : User Disconnected."
Call AddRTFStatus(sckServer.RemoteHostIP & " Has Left the Conversation.", RGB(123, 0, 0))
SessionInfo.FT_InProgress = False

If KeepLog = True Then
   Call AddLog(sckServer.RemoteHostIP &  " disconnected at " & Now)
End If

If SessionInfo.SessionType = SESSION_SERVER Then
   sckServer.Listen
End If
End Sub

Private Sub sckServer_ConnectionRequest(ByVal requestID As Long)
If sckServer.State <> sckConnected Then
   If SessionInfo.SessionTyp e = SESSION_SERVER Then
      sckServer.Close
      sckServer.Accept requestID
      Dim iRep As Integer
      iRep = MsgBox(sckServer.RemoteHostIP & " is attempting to start a session with you.
Accept ?", vbQuestion + vbYesNo, "Session Request")
         Dim sPack As String
         If iRep = vbNo Then
            sPack = sHeader & sDelim & "S" & sDelim & "Denied"
            sckServer.SendData sPack
            SessionInfo.Connected = False
         ElseIf iRep = vbYes Then
            sPack = sHeader & sDelim & "S" & sDelim & "Accepted"
            sckServer.SendData sPack
```

```
            Call AddRTFStatus(sckServer.RemoteHostIP & " Joined the Conversation.", RGB(0, 0,
123))
            SessionInfo.Connected = True
            StatusBar.SimpleText = "Status : Session Started."
            FrameConvo.Caption = " Conversation (In Progress)  "
               If KeepLog = True Then
                  Call AddLog(sckServer.RemoteHostIP &  " connected at " & Now)
               End If

         End If
    End If
End If
End Sub

Private Sub sckServer_DataArrival(ByVal bytesTotal As Long)
Dim cData As String: cData = Empty
sckServer.GetData cData
Select Case Mid(cData, 19, 1)
   Case "M"
      Call ParseMessage(cData)
   Case "F"
      Call ParseTransferRequest(cData)
   Case "R"
      Call ParseTransferReply(cData)
   Case "D"
      Call ParseReadySignal(cData)
   Case "C"
      Call ParseTransferCancel(cData)
   Case "W"
      Call ParseWhoIsRequest(cData)
End Select
End Sub

Private Sub st_Click()
If st.Value = 1 Then
Timer1.Enabled = True
Else
Timer1.Enabled = False
Text5.Text = 0
Text6.Text = 0
Text7.Text = 0
End If
End Sub

Private Sub Timer1_Timer()
Time = GetTickCount()
Text5.Text = Time
End Sub
```

```
Private Sub txtChat_Change()
txtChat.SelStart = Len(txtChat.Text)
End Sub

Private Sub txtHost_Change()
If Len(Trim$(txtHost.Text)) = 0 Then
   cmdConnect.Enabled = False
Else
   cmdConnect.Enabled = True
End If
End Sub

Private Sub txtNick_Change()
If Len(Trim$(txtNick.Text)) = 0 Then
   cmdSend.Enabled = False
   txtSend.Enabled = False
Else
   cmdSend.Enabled = True
   txtSend.Enabled = True
End If
End Sub

Private Sub txtSend_KeyPress(KeyAscii As Integer)
If KeyAscii = vbKeyReturn Then
KeyAscii = 0
cmdSend_Click
End If
End Sub

Private Sub decText_Click()
   Dim pass()       As Byte
   Dim plaintext()   As Byte
   Dim ciphertext()  As Byte
   Dim KeyBits      As Long
   Dim BlockBits     As Long

   If Len(Text1.Text) = 0 Then
      MsgBox "No Plaintext"
   Else
      If Len(txtpass.Text) = 0 Then
         MsgBox "No Password"
      Else
a = Text5.Text
Text6.Text = a
b = a
         KeyBits = Combo1.ItemData(Combo1.ListIndex)
         BlockBits = 128
```

```
        pass = GetPassword

        If HexDisplayRev(Text1.Text, ciphertext) = 0 Then
           MsgBox "Text not Hex data"
           Exit Sub
        End If

        m_Rijndael.SetCipherKey pass, KeyBits, BlockBits
        If m_Rijndael.ArrayDecrypt(plaintext, ciphertext, 0, BlockBits) <> 0 Then

        End If

           DisplayString2 Text1, StrConv(plaintext, vbUnicode)
If st.Value = 1 Then
c = Text6.Text
Text7.Text = GetTickCount() - c
mdetik = GetTickCount() - c
Text5.Text = GetTickCount()
MsgBox ("Kecepatan dekripsi chipertext tsb pada PC anda = " & mdetik & " millisecond")
Text6.Text = 0
Text7.Text = 0
c = 0
End If
          End If

   End If

End Sub


Private Sub Form_Initialize()

   Combo1.AddItem "128 Bit"
   Combo1.ItemData(Combo1.NewIndex) = 128

   Combo1.AddItem "192 Bit"
   Combo1.ItemData(Combo1.NewIndex) = 192

   Combo1.AddItem "256 Bit"
   Combo1.ItemData(Combo1.NewIndex) = 256

   Combo1.ListIndex = 0

End Sub

Option Explicit

#Const COMPILE_CONSTANTS = 0
```

```
Private Te0(255)     As Long
Private Te1(255)     As Long
Private Te2(255)     As Long
Private Te3(255)     As Long
Private Te4(255)     As Long

Private Td0(255)     As Long
Private Td1(255)     As Long
Private Td2(255)     As Long
Private Td3(255)     As Long
Private Td4(255)     As Long


Private rco(28)      As Long

Private Nr           As Long

Private fkey(119)    As Long
Private rkey(119)    As Long

Private Const MaxFileChunkSize As Long = 4000000

Private Declare Sub CopyMemory Lib "kernel32" Alias "RtlMoveMemory" (Destination As
Any, Source As Any, ByVal Length As Long)
Private Sub CreateDecryptionKeys(Nb As Long)
   Dim i    As Long
   Dim j    As Long
   Dim k    As Long
   Dim s(3) As Byte


   i = 0
   j = Nb * Nr
   For k = 0 To Nr
      CopyMemory rkey(i), fkey(j), Nb * 4&
      i = i + Nb
      j = j - Nb
   Next k


   For i = Nb To Nb * Nr - 1
      CopyMemory s(0), rkey(i), 4&
      rkey(i) = Td0(Te4(s(0)) And &HFF&) Xor _
            Td1(Te4(s(1)) And &HFF&) Xor _
            Td2(Te4(s(2)) And &HFF&) Xor _
            Td3(Te4(s(3)) And &HFF&)
   Next i
```

```
End Sub
Public Function SetCipherKey(pass() As Byte, KeyBits As Long, BlockBits As Long) As Long

   Dim i    As Long
   Dim j    As Long
   Dim s(3) As Byte

   Select Case KeyBits
   Case 128
      i = 4
      CopyMemory fkey(0), pass(0), 4& * i
      For j = 0 To 9
         CopyMemory s(0), fkey(i - 1), 4&
         fkey(i) = fkey(i - 4) Xor (Te4(s(0)) And &HFF000000) _
                   Xor (Te4(s(3)) And &HFF0000) _
                   Xor (Te4(s(2)) And &HFF00&) _
                   Xor (Te4(s(1)) And &HFF&) _
                   Xor rco(j)
         fkey(i + 1) = fkey(i - 3) Xor fkey(i)
         fkey(i + 2) = fkey(i - 2) Xor fkey(i + 1)
         fkey(i + 3) = fkey(i - 1) Xor fkey(i + 2)
         i = i + 4
      Next j
      Nr = 10


   Case 192
      i = 6
      j = 0
      CopyMemory fkey(0), pass(0), 4& * i
      Do
         CopyMemory s(0), fkey(i - 1), 4&
         fkey(i) = fkey(i - 6) Xor (Te4(s(0)) And &HFF000000) _
                   Xor (Te4(s(3)) And &HFF0000) _
                   Xor (Te4(s(2)) And &HFF00&) _
                   Xor (Te4(s(1)) And &HFF&) _
                   Xor rco(j)
         fkey(i + 1) = fkey(i - 5) Xor fkey(i)
         fkey(i + 2) = fkey(i - 4) Xor fkey(i + 1)
         fkey(i + 3) = fkey(i - 3) Xor fkey(i + 2)
         If j = 7 Then Exit Do
         fkey(i + 4) = fkey(i - 2) Xor fkey(i + 3)
         fkey(i + 5) = fkey(i - 1) Xor fkey(i + 4)
         i = i + 6
         j = j + 1
      Loop
      Nr = 12
```

```
    Case 256
      i = 8
      j = 0
      CopyMemory fkey(0), pass(0), 4& * i
      Do
        CopyMemory s(0), fkey(i - 1), 4&
        fkey(i) = fkey(i - 8) Xor (Te4(s(0)) And &HFF000000) _
                  Xor (Te4(s(3)) And &HFF0000) _
                  Xor (Te4(s(2)) And &HFF00&) _
                  Xor (Te4(s(1)) And &HFF&) _
                  Xor rco(j)
        fkey(i + 1) = fkey(i - 7) Xor fkey(i)
        fkey(i + 2) = fkey(i - 6) Xor fkey(i + 1)
        fkey(i + 3) = fkey(i - 5) Xor fkey(i + 2)
        If j = 6 Then Exit Do
        CopyMemory s(0), fkey(i + 3), 4&
        fkey(i + 4) = fkey(i - 4) Xor (Te4(s(3)) And &HFF000000) _
                      Xor (Te4(s(2)) And &HFF0000) _
                      Xor (Te4(s(1)) And &HFF00&) _
                      Xor (Te4(s(0)) And &HFF&)
        fkey(i + 5) = fkey(i - 3) Xor fkey(i + 4)
        fkey(i + 6) = fkey(i - 2) Xor fkey(i + 5)
        fkey(i + 7) = fkey(i - 1) Xor fkey(i + 6)
        i = i + 8
        j = j + 1
      Loop
      Nr = 14

    Case Else
      Err.Raise 1, , "cRijndael.SetCipherKey  - Illegal KeyBits Value"
      SetCipherKey = 1
      Exit Function
    End Select
    CreateDecryptionKeys 4


End Function


Public Function SetCipherKeyString(PassPhrase As String, KeyBits As Long, BlockBits As Long)
As Long
    Dim pass() As Byte

    pass = StrConv(PassPhrase, vbFromUnicode)
    ReDim Preserve pass(31)
    SetCipherKeyString = SetCipherKey(pass, KeyBits, BlockBits)
End Function
```

```
Public Sub BlockEncrypt(plaintext() As Byte, ciphertext() As Byte, p As Long, q As Long)
   Dim i    As Long
   Dim k    As Long
   Dim t0   As Long
   Dim t1   As Long
   Dim t2   As Long
   Dim t3   As Long
   Dim s(15) As Byte

   CopyMemory t0, plaintext(p + 0), 4&
   CopyMemory t1, plaintext(p + 4), 4&
   CopyMemory t2, plaintext(p + 8), 4&
   CopyMemory t3, plaintext(p + 12), 4&
   t0 = t0 Xor fkey(0)
   t1 = t1 Xor fkey(1)
   t2 = t2 Xor fkey(2)
   t3 = t3 Xor fkey(3)
   k = 4

   For i = 1 To Nr - 1
      CopyMemory s(0), t0, 4&
      CopyMemory s(4), t1, 4&
      CopyMemory s(8), t2, 4&
      CopyMemory s(12), t3, 4&
      t0 = Te0(s(0)) Xor Te1(s(5)) Xor Te2(s(10)) Xor Te3(s(15)) Xor fkey(k + 0)
      t1 = Te0(s(4)) Xor Te1(s(9)) Xor Te2(s(14)) Xor Te3(s(3)) Xor fkey(k + 1)
      t2 = Te0(s(8)) Xor Te1(s(13)) Xor Te2(s(2)) Xor Te3(s(7)) Xor fkey(k + 2)
      t3 = Te0(s(12)) Xor Te1(s(1)) Xor Te2(s(6)) Xor Te3(s(11)) Xor fkey(k + 3)
      k = k + 4
   Next i

   CopyMemory s(0), t0, 4&
   CopyMemory s(4), t1, 4&
   CopyMemory s(8), t2, 4&
   CopyMemory s(12), t3, 4&
   t0 = (Te4(s(0)) And &HFF&) Xor (Te4(s(5)) And &HFF00&) Xor (Te4(s(10)) And &HFF0000)
Xor (Te4(s(15)) And &HFF000000) Xor fkey(k + 0)
   t1 = (Te4(s(4)) And &HFF&) Xor (Te4(s(9)) And &HFF00&) Xor (Te4 (s(14)) And &HFF0000)
Xor (Te4(s(3)) And &HFF000000) Xor fkey(k + 1)
   t2 = (Te4(s(8)) And &HFF&) Xor (Te4(s(13)) And &HFF00&) Xor (Te4(s(2)) And &HFF0000)
Xor (Te4(s(7)) And &HFF000000) Xor fkey(k + 2)
   t3 = (Te4(s(12)) And &HFF&) Xor (Te4(s(1)) And  &HFF00&) Xor (Te4(s(6)) And &HFF0000)
Xor (Te4(s(11)) And &HFF000000) Xor fkey(k + 3)
   CopyMemory ciphertext(q + 0), t0, 4&
   CopyMemory ciphertext(q + 4), t1, 4&
   CopyMemory ciphertext(q + 8), t2, 4&
   CopyMemory ciphertext(q + 12), t3, 4&
```

```
End Sub

Public Sub BlockDecrypt(plaintext() As Byte, ciphertext() As Byte, p As Long, q As Long)
   Dim i    As Long
   Dim k    As Long
   Dim t0   As Long
   Dim t1   As Long
   Dim t2   As Long
   Dim t3   As Long
   Dim s(15) As Byte

   CopyMemory t0, ciphertext(q + 0), 4&
   CopyMemory t1, ciphertext(q + 4), 4&
   CopyMemory t2, ciphertext(q + 8), 4&
   CopyMemory t3, ciphertext(q + 12), 4&
   t0 = t0 Xor rkey(0)
   t1 = t1 Xor rkey(1)
   t2 = t2 Xor rkey(2)
   t3 = t3 Xor rkey(3)
   k = 4

   For i = 1 To Nr - 1
      CopyMemory s(0), t0, 4&
      CopyMemory s(4), t1, 4&
      CopyMemory s(8), t2, 4&
      CopyMemory s(12), t3, 4&
      t0 = Td0(s(0)) Xor Td1(s(13)) Xor Td2(s(10)) Xor Td3(s(7)) Xor rkey(k + 0)
      t1 = Td0(s(4)) Xor Td1(s(1)) Xor Td2(s(14)) Xor Td3(s(11)) Xor rkey(k + 1)
      t2 = Td0(s(8)) Xor Td1(s(5)) Xor Td2(s(2)) Xor Td3(s(15)) Xor rkey(k + 2)
      t3 = Td0(s(12)) Xor Td1(s(9)) Xor Td2(s(6)) Xor Td3(s(3)) Xor rkey(k + 3)
      k = k + 4
   Next i

   CopyMemory s(0), t0, 4&
   CopyMemory s(4), t1, 4&
   CopyMemory s(8), t2, 4&
   CopyMemory s(12), t3, 4&
   t0 = (Td4(s(0)) And &HFF&) Xor (Td4(s(13)) And &HFF00&) Xor (Td4(s(10)) And &HFF0000)
Xor (Td4(s(7)) And &HFF000000) Xor rkey(k + 0)
   t1 = (Td4(s(4)) And &HFF&) Xor (Td4(s(1)) And &HFF00&) Xor (Td4(s(14)) And &HFF0000)
Xor (Td4(s(11)) And &HFF000000) Xor rkey(k + 1)
   t2 = (Td4(s(8)) And &HFF&) Xor (Td4(s(5)) And &HFF00&) Xor (Td4(s(2)) And & HFF0000)
Xor (Td4(s(15)) And &HFF000000) Xor rkey(k + 2)
   t3 = (Td4(s(12)) And &HFF&) Xor (Td4(s(9)) And &HFF00&) Xor (Td4(s(6)) And &HFF0000)
Xor (Td4(s(3)) And &HFF000000) Xor rkey(k + 3)
   CopyMemory plaintext(p + 0), t0, 4&
   CopyMemory plaintext(p + 4), t1, 4&
   CopyMemory plaintext(p + 8), t2, 4&
```

```
    CopyMemory plaintext(p + 12), t3, 4&
End Sub

Public Function ArrayEncrypt(plaintext() As Byte, ciphertext() As Byte, appendsize As Long,
BlockBits As Long) As Long

    Dim i         As Long
    Dim m         As Long
    Dim n         As Long

    Const BlockSize As Long = 16 'bytes

    If LBound(plaintext) <> 0 Then Err.Raise 1, , "cRijndael.ArrayEncrypt  - plaintext must be
zero based array"

    n = UBound(plaintext) + 1
    If appendsize = 0 Then

        m = ((n + BlockSize - 1) \ BlockSize) * BlockSize

        ReDim ciphertext(m - 1)
    Else

        m = ((n + BlockSize) \ BlockSize) * BlockSize

        ReDim ciphertext(m - 1)
        ciphertext(m - 1) = n Mod BlockSize
    End If



    For i = 0 To n - BlockSize Step BlockSize
        BlockEncrypt plaintext, ciphertext, i, i
    Next i


    If (n Mod BlockSize) <> 0 Then CopyMemory ciphertext(i), plaintext(i), n Mod BlockSize



    If i <> m Then BlockEncrypt ciphertext, cipherte xt, i, i

End Function


Public Function ArrayDecrypt(plaintext() As Byte, ciphertext() As Byte, appendsize As Long,
BlockBits As Long) As Long
```

```
    Dim i         As Long
    Dim m         As Long
    Dim n         As Long

    Const BlockSize As Long = 16 'bytes

    If LBound(ciphertext) <> 0 Then Err.Raise 1, , "cRijndael.ArrayDecrypt  - ciphertext must be
zero based array"

    n = UBound(ciphertext) + 1
    If ((n Mod BlockSize) = 0) Then
       ReDim plaintext(n - 1)



       For i = 0 To n - BlockSize Step BlockSize
          BlockDecrypt plaintext, ciphertext, i, i
       Next i

       If appendsize Then
          If plaintext(n - 1) < BlockSize Then
             n = n - BlockSize + plaintext(n - 1)
             If n > 0 Then ReDim Preserve plaintext(n  - 1)
          Else
             MsgBox "warning - incorrect length field"
             ArrayDecrypt = 1
          End If
       End If
    Else
       MsgBox "ciphertext size not a multiple of  block size"
       ArrayDecrypt = -1
    End If

End Function

Public  Function  FileEncrypt(PlaintextFileName  As  String,  CiphertextFileName  As  String,
BlockBits As Long) As Long

    Dim FileNum     As Integer
    Dim FileNum2    As Integer
    Dim i         As Long
    Dim m         As Long
    Dim n         As Long
    Dim data()     As Byte

    Const BlockSize As Long = 16 'bytes
    Const MaxBlocks As Long = MaxFileChunkSize  \ BlockSize
```

```
    n = FileLen(PlaintextFileName)

    m = ((n + BlockSize) \ BlockSize) * BlockSize

    FileNum = FreeFile
    Open PlaintextFileName For Binary Access Read As FileNum
    FileNum2 = FreeFile
    Open CiphertextFileName For Binary Access Write As FileNum2


    If m > MaxBlocks * BlockSize The n
       ReDim data(MaxBlocks * BlockSize - 1)
       Do
          Get #FileNum, , data

          For i = 0 To (MaxBlocks - 1) * BlockSize Step BlockSize
             BlockEncrypt data, data, i, i
          Next i

          Put #FileNum2, , data
          m = m - MaxBlocks * BlockSize
       Loop While m > MaxBlocks * BlockSize
    End If


    ReDim data(m - 1)
    Get #FileNum, , data
    data(m - 1) = n Mod BlockSize

    For i = 0 To m - BlockSize Step BlockSize
       BlockEncrypt data, data, i, i
    Next i

    Put FileNum2, , data

    Close FileNum
    Close FileNum2
End Function


Public Function FileDecrypt(PlaintextFileName As String, CiphertextFileName As String,
BlockBits As Long) As Long

    Dim FileNum     As Integer
    Dim FileNum2    As Integer
    Dim i           As Long
    Dim m           As Long
    Dim n           As Long
```

```
Dim data()     As Byte

Const BlockSize As Long = 16 'bytes
Const MaxBlocks As Long = MaxFileChunkSize  \ BlockSize

m = FileLen(CiphertextFileName)

If (m = 0) Or ((m Mod BlockSize) <> 0) Then

   MsgBox "File Size Error - ciphertext file not a multiple of block size"
   FileDecrypt = 1
Else
   FileNum = FreeFile
   Open CiphertextFileName For Binary Access Read As FileNum
   FileNum2 = FreeFile
   Open PlaintextFileName For Binary Access Write As FileNum2

   If m > MaxBlocks * BlockSize Then
      ReDim data(MaxBlocks * BlockSize - 1)
      Do
         Get #FileNum, , data

         For i = 0 To (MaxBlocks - 1) * BlockSize Step BlockSize
            BlockDecrypt data, data, i, i
         Next i

         Put #FileNum2, , data
         m = m - MaxBlocks * BlockSize
      Loop While m > MaxBlocks * BlockSize
   End If

   ReDim data(m - 1)
   Get #FileNum, , data

   For i = 0 To m - BlockSize Step BlockSize
      BlockDecrypt data, data, i, i
   Next i

   If data(m - 1) < BlockSize Then
      n = m - BlockSize + CLng(data(m - 1))
   Else

      MsgBox "warning - incorrect length field in decrypted file." & vbCrLf & "Wrong key,
keysize, or blocksize?"

      n = m
   End If
   If n > 0 Then
```

```
        ReDim Preserve data(n - 1)
        Put FileNum2, , data
      End If

      Close FileNum
      Close FileNum2
    End If
End Function

End Sub
```