Name of Experiment: Encryption and Decryption	**Exp No:** MC13

Background: Student should have basic knowledge of C#.

Summary: Encryption and Decryption techniques are required to solve the confidential issues related to mobile communication such as SMS, Emails etc. Some information is very critical and need to be confined between sender and recipient only.

Learning Objective: Student should get to know the process of securing the message and emails in order to maintain the confidentiality.

Target Platforms: This experiment is tested on Windows Phone Emulator and Nokia Lumia 800.

Procedure:

Step 1. Repeat steps [1-4] as in experiment no MC1 [Refer Hello World].

Step 2. Drag and drop the UI controls as 3 textboxes, 3 buttons and 1textblock on the MainPage.xaml.[refer Source code section]

Step 3. For each button there is a button event handler defined in the MainPage.xaml.cs.

Step 4. Add following using statements as using System.Text, using System.Security.Cryptography , using Microsoft.Phone.Controls, using System.Windows and using System to the MainPage.xaml.cs.

Step 5. Define an instance for RSACryptoServiceProvider as RSA, UnicodeEncoding as ByteConverter and two byte[] arrays as plaintext and encryptedtxt.

Step 6. Now define the two static methods as RSAEncrypt and RSADecrypt.[refer source code section]

Step 7. Now inside each button event handler, call these methods and validate the click event in the MainPage.xaml.cs. [refer source code section]

Step 8. Save all the changes, by pressing ctrl + S.

Step 9. Press F5, for debugging the experiment on the Windows Phone Emulator.

Step 10. By this way, we are able to test and deploy the experiment on Windows Phone Emulator.

Source Code	Comments

MainPage.xaml
 	
`<!--LayoutRoot is the root grid where all page content is placed-->`
`<Grid x:Name="LayoutRoot" Background="Transparent">`
`<Grid.RowDefinitions>`
`<RowDefinition Height="Auto"/>`
`<RowDefinition Height="*"/>`
`</Grid.RowDefinitions>`
 	
`<!--TitlePanel contains the name of the application and page title-->`
`<StackPanel x:Name="TitlePanel" Grid.Row="0" Margin="12,17,0,28">`
`<TextBlock x:Name="ExperimentTitle" Text="ExpNO MC13" TextAlignment="Right" Style="{StaticResource PhoneTextNormalStyle}"/>`
`<TextBlock x:Name="ApplicationTitle" Text="Mobile Computing" Style="{StaticResource PhoneTextNormalStyle}"/>`
`<TextBlock x:Name="PageTitle" Text="Mobile Security" Margin="9,-7,0,0" Style="{StaticResource PhoneTextTitle1Style}"/>`
`</StackPanel>`
 	
`<!--ContentPanel - place additional content here-->`
`<Grid x:Name="ContentPanel" Grid.Row="1" Margin="12,0,12,0">`
`<TextBlock Height="30" HorizontalAlignment="Left" Margin="12,28,0,0" Name="textBlock1" Text="Message" VerticalAlignment="Top" />`
`<TextBox Height="72" HorizontalAlignment="Left" Margin="6,64,0,0" Name="txtMessg" Text="" VerticalAlignment="Top" Width="450" />`
`<Button Content="Encrypt" Height="72" HorizontalAlignment="Left" Margin="128,142,0,0" Name="bEncrypt" VerticalAlignment="Top" Width="160" Click="bEncrypt_Click"/>`
`<TextBox Height="159" HorizontalAlignment="Left" Margin="9,205,0,0" Name="txtEncrypt" Text="" " VerticalAlignment="Top" Width="438" AcceptsReturn="False" BorderBrush="Green" Background="Black" Foreground="Gray" TextWrapping="Wrap" FontSize="{StaticResource PhoneFontSizeSmall}" />`
`<Button Content="Decrypt" Height="72" HorizontalAlignment="Left" Margin="141,354,0,0" Name="bDecrypt" VerticalAlignment="Top" Width="160"`
	← ExpNo13 as(Experiment Title)
	
	← Mobile Computing (Application Title)
	
	←Mobile Security (Page Title)

```
Click="bDecrypt_Click" />
            <TextBox Height="136"
HorizontalAlignment="Left"
Margin="3,412,0,0" Name="txtDecrypt" Text="
" VerticalAlignment="Top" Width="447"
BorderBrush="Green" Background="Black"
Foreground="Gray" TextWrapping="Wrap"
FontSize="{StaticResource
PhoneFontSizeSmall}"/>
            <Button Content="Reset"
Height="72" HorizontalAlignment="Right"
Margin="0,530,181,0" Name="bReset"
VerticalAlignment="Top" Width="134"
Click="bReset_Click" />
        </Grid>
    </Grid>
```

MainPage.xaml.cs

```
using System;
using System.Windows;
using Microsoft.Phone.Controls;
using System.Security.Cryptography;
using System.Text;

namespace Mobile_Security
{
    public partial class MainPage :
PhoneApplicationPage
    {
        UnicodeEncoding ByteConverter = new
UnicodeEncoding();
        RSACryptoServiceProvider RSA = new
RSACryptoServiceProvider();
        byte[] plaintxt;
        byte[] encryptedtxt;

        // Constructor
        public MainPage()
        {
            InitializeComponent();
        }


        static public byte[]
RSAEncrypt(byte[] byteEncrypt, RSAParameters
RSAInfo, bool isOAEP)
        {
            try
            {
          byte[] encryptedData;
   //Create a new instance of
RSACryptoServiceProvider.
using (RSACryptoServiceProvider RSA = new
RSACryptoServiceProvider())
                {
RSA.ImportParameters(RSAInfo);

  //Encrypt the passed byte array.
 encryptedData = RSA.Encrypt(byteEncrypt,
isOAEP);
                }
              return encryptedData;
```

← instance for RSACryptoServiceProvider
← byte [ ] arrays

```
            }
            //Catch and display a
CryptographicException
            //to the console.
            catch (CryptographicException e)
            {

Console.WriteLine(e.Message);

                return null;
            }

        }
        static public byte[]
RSADecrypt(byte[] byteDecrypt, RSAParameters
RSAInfo, bool isOAEP)
        {
            try
            {
                byte[] decryptedData;
 //Create a new instance of
RSACryptoServiceProvider.
                using
(RSACryptoServiceProvider RSA = new
RSACryptoServiceProvider())
                {

RSA.ImportParameters(RSAInfo);

 //Decrypt the passed byte array and specify
OAEP padding.
 decryptedData = RSA.Decrypt(byteDecrypt,
isOAEP);
                }
                return decryptedData;
            }
//Catch and display a CryptographicException
            //to the console.
            catch (CryptographicException e)
            {

Console.WriteLine(e.ToString());

                return null;
            }
        }
  public void bEncrypt_Click(object sender,
RoutedEventArgs e)
        {
            if (Validate())
            {
                plaintxt =
ByteConverter.GetBytes(txtMessg.Text);
                encryptedtxt =
RSAEncrypt(plaintxt,
RSA.ExportParameters(false), false);
                txtEncrypt.Text =
ByteConverter.GetString(encryptedtxt, 0,
encryptedtxt.Length);

            }
        }
```

←Encryption of Text

```
        public void bDecrypt_Click(object
sender, RoutedEventArgs e)
        {
            if (Validate1())
            {
                byte[] decryptedtxt =
RSADecrypt(encryptedtxt,
RSA.ExportParameters(true), false);
txtDecrypt.Text =
ByteConverter.GetString(decryptedtxt, 0,
decryptedtxt.Length);
            }
        }
```

← Decryption of Text

```
        #region UIValidation
        private bool Validate() {
            if
(string.IsNullOrWhiteSpace(txtMessg.Text)) {
MessageBox.Show("Please Enter some mesage.
");
                return false;
            }
            return true;
        }
        private bool Validate1() {
            if
(string.IsNullOrWhiteSpace(txtEncrypt.Text))
{
 MessageBox.Show("Please do encryption
first");
                return false;

            }
            return true;
        }
        #endregion
```

← For UI Validation

```
        private void bReset_Click(object
sender, RoutedEventArgs e)
        {
            txtMessg.Text = string.Empty;
            txtEncrypt.Text = string.Empty;
            txtDecrypt.Text = string.Empty;

        }

    }
}
```

← For Reset the Values

Screenshots:

Fig. No 1 Home Screen

Fig. No 2 Type Message

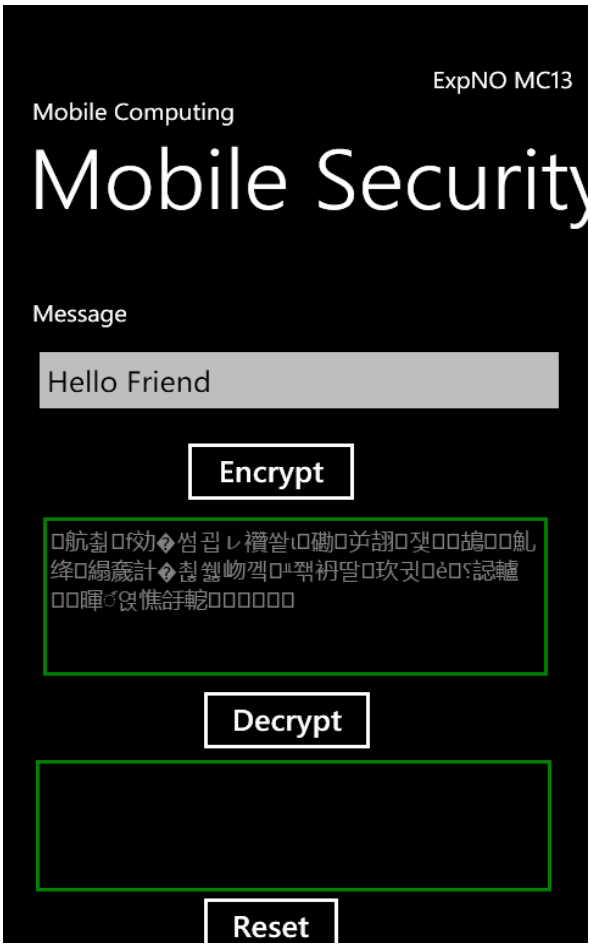

Fig. No 3 Press Encrypt button

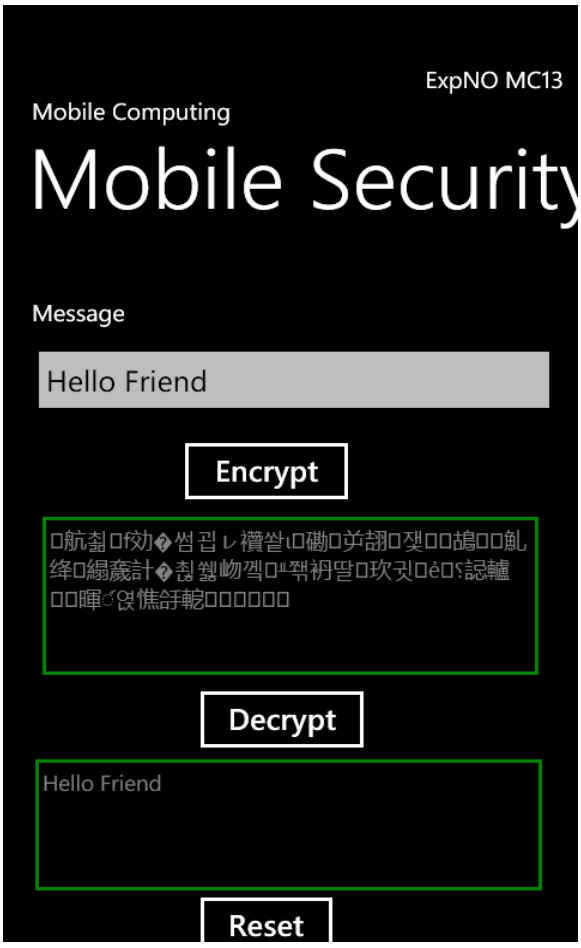

Fig. No 4 Press Decrypt button

Observations: It is observed by the developer that RSA algorithm proved to be useful for the encryption/decryption of message and emails.