Erstellen sicherer ASP.NET-Anwendungen

Authentifizierung, Autorisierung und sichere Kommunikation

Auf der Orientierungsseite finden Sie einen Ausgangspunkt und eine vollständige Übersicht zum *Erstellen sicherer ASP.NET-Anwendungen*.

Zusammenfassung

Bei manchen Anwendungen erweist es sich als zweckmäßig, verschlüsselte Daten wie Verbindungszeichenfolgen und Kontoinformationen in der Windows-Registrierung zu speichern. Nachstehend wird erläutert, wie verschlüsselte Zeichenfolgen in der Registrierung gespeichert und aus dieser wieder abgerufen werden.

Vorgehensweise: Speichern von verschlüsselten Verbindungszeichenfolgen in der Registrierung

Eine mögliche Position zum Speichern von Datenbank-Verbindungszeichenfolgen ist die Registrierung. Zwar können einzelne Registrierungsschlüssel mit Windows-Zugriffssteuerungslisten (Access Control Lists, ACLs) gesichert werden, dennoch sollten Sie Verbindungszeichenfolgen verschlüsselt speichern, um die Sicherheit zu erhöhen.

In dieser Vorgehensweise wird erläutert, wie eine verschlüsselte Datenbank-Verbindungszeichenfolge in der Registrierung gespeichert und mithilfe einer ASP.NET-Webanwendung abgerufen wird. Hierbei wird die generische verwaltete Klassenbibliothek für Ver- und Entschlüsselung verwendet, die in "Vorgehensweise: Erstellen einer Verschlüsselungsbibliothek" erstellt wurde. Diese Vorgehensweise finden Sie im Abschnitt "Referenz" dieses Handbuchs.

Falls Sie diese Verschlüsselungsklassenbibliothek noch nicht erstellt haben, sollten Sie dies nachholen, bevor Sie mit der aktuellen Vorgehensweise fortfahren.

Weitere Informationen über andere Positionen und Möglichkeiten zum sicheren Speichern von Datenbank-Verbindungszeichenfolgen finden Sie unter "Sicheres Speichern von Datenbank-Verbindungszeichenfolgen" in Kapitel 12, "Datenzugriffssicherheit".

Anmerkungen

- Die Verbindungszeichenfolge, der Initialisierungsvektor und der Schlüssel für die Verschlüsselung werden in der Registrierung als benannte Werte unter dem folgenden Registrierungsschlüssel gespeichert:

  ```
  HKEY_LOCAL_MACHINE\Software\TestApplication
  ```

- Initialisierungsvektor und Schüssel müssen gespeichert werden, damit die Verbindungszeichenfolge entschlüsselt werden kann.

Anforderungen

Im Folgenden finden Sie eine Liste der empfohlenen Hardware und Software und eine Beschreibung der Netzwerkinfrastruktur, der Fähigkeiten und Kenntnisse sowie der Service Packs, die Sie benötigen.

- Microsoft® Windows® 2000 als Betriebssystem
- Microsoft Visual Studio® .NET als Entwicklungssystem

Die hierin erläuterten Verfahren setzen zudem Kenntnisse des Entwicklungstools Microsoft Visual C#™ voraus.

Zusammenfassung

Diese Vorgehensweise enthält folgende Verfahren:

1. Speichern der verschlüsselten Daten in der Registrierung
2. Erstellen einer ASP.NET-Webanwendung

1. Speichern der verschlüsselten Daten in der Registrierung

Mit diesem Verfahren wird eine Windows-Anwendung erstellt, die zum Verschlüsseln einer Datenbank-Verbindungszeichenfolge und zum Speichern derselben in der Registrierung verwendet wird.

▶ **So speichern Sie die verschlüsselten Daten in der Registrierung**

1. Starten Sie Visual Studio .NET, und erstellen Sie mit C# ein neues Windows-Projekt mit Namen **EncryptionTestApp**.
2. Fügen Sie einen Assemblyverweis auf die Assembly **Encryption.dll** hinzu.
   Zum Erstellen dieser Assembly führen Sie die unter "Vorgehensweise: Erstellen einer Verschlüsselungsbibliothek" im Abschnitt "Referenz" dieses Handbuchs genannten Schritte durch.
3. Fügen Sie am Anfang von **Form1.cs** unterhalb der vorhandenen **using**-Anweisungen die folgenden **using**-Anweisungen hinzu.

```
using Encryption;
using System.Text;
using Microsoft.Win32;
```

4. Fügen Sie **Form1** die in Tabelle 1 aufgeführten Steuerelemente hinzu, und ordnen Sie diese wie in Abbildung 1 gezeigt an.

Tabelle 1: Steuerelemente für **EncryptionTestApp**

Steuerelement	Text	ID
Bezeichnungsfeld	Connection String:	
Textfeld		txtConnectionString
Bezeichnungsfeld	Key:	
Textfeld		txtKey
Bezeichnungsfeld	Initialization Vector:	
Textfeld		txtInitializationVector
Bezeichnungsfeld	Encrypted String	
Textfeld		txtEncryptedString
Bezeichnungsfeld	Decrypted String	
Textfeld		txtDecryptedString
Schaltfläche	Encrypt	btnEncrypt
Schaltfläche	Decrypt	btnDecrypt
Schaltfläche	Write Registry Data	btnWriteRegistryData

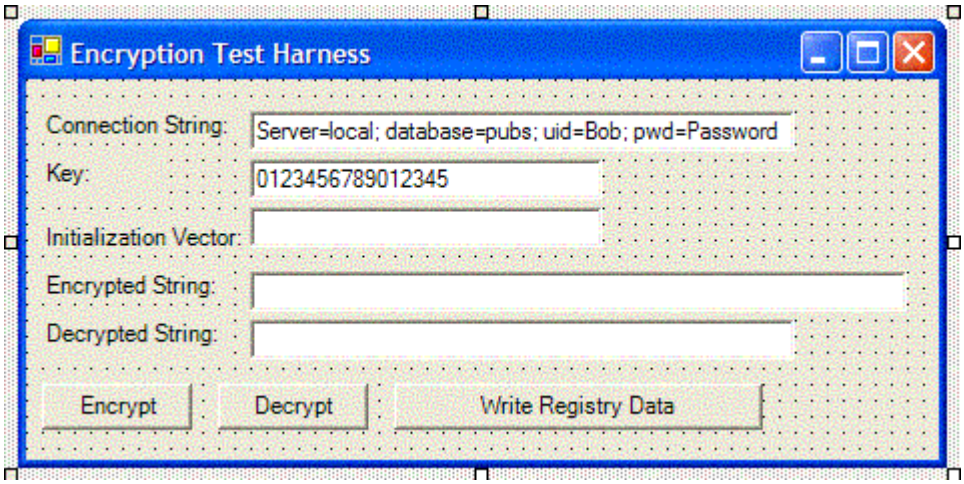

Abbildung 1
Dialogfeld "Encryption Test Harness"

5. Legen Sie die **Text**-Eigenschaft von **txtConnectionString** wie folgt fest:

```
"Server=local; database=pubs; uid=Bob; pwd=Password"
```

6. Legen Sie die **Text**-Eigenschaft von **txtKey** wie folgt fest:

```
"0123456789012345"
```

   Die Schlüssellänge beträgt 16 Byte, damit der 3DES-Verschlüsselungsalgorithmus verwendet werden kann.

7. Legen Sie die **Text**-Eigenschaft von **Form1** wie folgt fest:

```
"Encryption Test Harness"
```

8. Doppelklicken Sie auf die Schaltfläche **Encrypt**, um einen Ereignishandler für das Klickereignis der Schaltfläche zu erstellen, und fügen Sie dem Ereignishandler den nachstehenden Code hinzu.

```
try
{
  // Create the encryptor object, specifying 3DES as the
  // encryption algorithm
  Encryptor enc = new Encryptor(EncryptionAlgorithm.TripleDes);
  // Get the connection string as a byte array
  byte[] plainText = Encoding.ASCII.GetBytes(txtConnectionString.Text);
  byte[] key = Encoding.ASCII.GetBytes(txtKey.Text);

  // Perform the encryption
  byte[] cipherText = enc.Encrypt(plainText, key);
  // Store the intialization vector, as this will be required
  // for decryption
  txtInitializationVector.Text = Encoding.ASCII.GetString(enc.IV);

  // Display the encrypted string
  txtEncryptedString.Text = Convert.ToBase64String(cipherText);
}
catch(Exception ex)
{
  MessageBox.Show("Exception encrypting: " + ex.Message,
                  "Encryption Test  Harness");
}
```

9. Kehren Sie zu **Form1** zurück, wechseln Sie in den Entwurfsmodus, und doppelklicken Sie auf die Schaltfläche **Decrypt**, um einen Ereignishandler für das Klickereignis der Schaltfläche zu erstellen.
10. Fügen Sie dem Ereignishandler für das Klickereignis der Schaltfläche **Decrypt** den folgenden Code hinzu.

```
try
{
  // Set up the Decryptor object
  Decryptor dec = new Decryptor(EncryptionAlgorithm.TripleDes);

  // Set the Initialization Vector
  dec.IV = Encoding.ASCII.GetBytes(txtInitializationVector.Text);

  byte[] key = Encoding.ASCII.GetBytes(txtKey.Text);
  // Perform the decryption
  byte[] plainText =  dec.Decrypt(Convert.FromBase64String(
                                  txtEncryptedString.Text),
                                  key);

  // Display the decrypted string.
  txtDecryptedString.Text = Encoding.ASCII.GetString(plainText);
}
catch(Exception ex)
{
  MessageBox.Show("Exception decrypting. " + ex.Message,
                  "Encryption Test Harness");
}
```

11. Kehren Sie zu **Form1** zurück, wechseln Sie ggf. in den Entwurfsmodus, und doppelklicken Sie auf die Schaltfläche **Write Registry Data**, um einen Ereignishandler für das Klickereignis der Schaltfläche zu erstellen.
12. Fügen Sie dem Ereignishandler den folgenden Code hinzu.

```
// Create registry key and named values
RegistryKey rk = Registry.LocalMachine.OpenSubKey("Software",true);
rk = rk.CreateSubKey("TestApplication");

// Write encrypted string, initialization vector and key to the registry
rk.SetValue("connectionString",txtEncryptedString.Text);
rk.SetValue("initVector",Convert.ToBase64String(
             Encoding.ASCII.GetBytes(txtInitializationVector.Text)));
rk.SetValue("key",Convert.ToBase64String(Encoding.ASCII.GetBytes(
                                        txtKey.Text)));
MessageBox.Show("The data has been successfully written to the registry");
```

13. Führen Sie die Anwendung aus, und klicken Sie dann auf **Encrypt**.
    Die verschlüsselte Verbindungszeichenfolge wird im Feld **Encrypted String** angezeigt.
14. Klicken Sie auf **Decrypt**.
    Die ursprüngliche Verbindungszeichenfolge wird im Feld **Decrypted String** angezeigt.
15. Klicken Sie auf **Write Registry Data**.
16. Klicken Sie im Meldungsfenster auf **OK**.
17. Führen Sie **Regedit.exe** aus, und zeigen Sie den Inhalt des folgenden Schlüssels an.

```
HKLM\Software\TestApplication
```

    Vergewissern Sie sich, dass die verschlüsselten Werte für die benannten Werte **connectionString**, **initVector** und **key** vorhanden sind.
18. Schließen Sie den Registrierungseditor und die Anwendung Test Harness.

2. Erstellen einer ASP.NET-Webanwendung

In diesem Verfahren wird eine einfache ASP.NET-Webanwendung entwickelt, die die verschlüsselte Verbindungszeichenfolge aus der Registrierung abruft und diese entschlüsselt.

▶ **So erstellen Sie eine ASP.NET-Anwendung**

1. Erstellen Sie mit Visual C# eine neue ASP.NET-Webanwendung mit Namen **EncryptionWebApp**.
2. Fügen Sie einen Assemblyverweis auf die Assembly **Encryption.dll** hinzu.
   Zum Erstellen dieser Assembly führen Sie die unter "Vorgehensweise: Erstellen einer Verschlüsselungsbibliothek" im Abschnitt "Referenz" dieses Handbuchs genannten Schritte durch.
3. Öffnen Sie **WebForm1.aspx.cs**, und fügen Sie am Anfang der Datei unterhalb der vorhandenen **using**-Anweisungen die folgenden **using**-Anweisungen hinzu.

```
using Encryption;
using System.Text;
using Microsoft.Win32;
```

4. Fügen Sie **WebForm1.aspx** die in Tabelle 2 aufgeführten Steuerelemente hinzu.

Tabelle 2: Steuerelemente für **WebForm1.aspx**

Steuerelement	Text	ID
Bezeichnungsfeld		lblEncryptedString
Bezeichnungsfeld		lblDecryptedString
Schaltfläche	Get Connection String	btnGetConnectionString

5. Doppelklicken Sie auf die Schaltfläche **Get Connection String**, um einen Ereignishandler für das Klickereignis der Schaltfläche zu erstellen.
6. Fügen Sie dem Ereignishandler den folgenden Code hinzu.

```
RegistryKey rk = Registry.LocalMachine.OpenSubKey(
                                          @"Software\TestApplication",false);
lblEncryptedString.Text = (string)rk.GetValue("connectionString");

string initVector = (string)rk.GetValue("initVector");
string strKey = (string)rk.GetValue("key");

Decryptor dec = new Decryptor(EncryptionAlgorithm.TripleDes );
dec.IV = Convert.FromBase64String(initVector);

// Decrypt the string
byte[] plainText = dec.Decrypt(Convert.FromBase64String(
                               lblEncryptedString.Text),
                               Convert.FromBase64String(strKey));

lblDecryptedString.Text = Encoding.ASCII.GetString(plainText);
```

7. Klicken Sie im Menü **Erstellen** auf **Projektmappe erstellen**.
8. Klicken Sie im Projektmappen-Explorer mit der rechten Maustaste auf **WebForm1.aspx**, und klicken Sie dann auf **In Browser anzeigen**.
9. Klicken Sie auf **Get Connection String**.
   Die ver- und entschlüsselten Verbindungszeichenfolgen werden im Webformular angezeigt.

Referenzen

Weitere Informationen finden Sie im Abschnitt "Referenz" dieses Handbuchs unter "Vorgehensweise: Erstellen einer Verschlüsselungsbibliothek".