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

Nachstehend wird erläutert, wie die DPAPI von einer ASP.NET-Webanwendung oder einem ASP.NET-Webdienst zum Verschlüsseln von vertraulichen Daten verwendet wird.

Vorgehensweise: Verwenden von DPAPI (Computerspeicher) von ASP.NET aus

In Webanwendungen müssen häufig sicherheitsrelevante Daten wie Datenbank-Verbindungszeichenfolgen und Anmeldeinformationen von Dienstkonten in Konfigurationsdateien gespeichert werden. Aus Sicherheitsgründen sollten solche Informationen niemals im Klartext gespeichert und vor der Speicherung immer verschlüsselt werden.

Diese Vorgehensweise erläutert, wie die DPAPI von ASP.NET aus verwendet wird. Dies umfasst ASP.NET-Webanwendungen und Webdienste ebenso wie .NET Remoting-Komponenten, für die ASP.NET als Host fungiert.

- Der Code in dieser Vorgehensweise verweist über eine verwaltete Klassenbibliothek auf die DPAPI. Wie diese Klassenbibliothek angelegt wird, ist unter "Vorgehensweise: Erstellen einer DPAPI-Bibliothek" im Abschnitt "Referenz" dieses Handbuchs beschrieben.

Hinweise

- Die DPAPI kann entweder mit dem Computerspeicher oder mit dem Benutzerspeicher arbeiten (was ein geladenes Benutzerprofil voraussetzt). Standardmäßig verwendet die DPAPI den Benutzerspeicher, Sie können jedoch auch festlegen, dass der Computerspeicher verwendet werden soll, indem Sie das Flag CRYPTOPROTECT_LOCAL_MACHINE an die DPAPI-Funktionen übergeben.
- Mit der Verwendung des Benutzerprofils wird eine zusätzliche Sicherheitsschicht eingezogen, denn hiermit wird der Zugriff auf die geheimen Daten weiter eingeschränkt. Nur der Benutzer, der die Daten verschlüsselt, kann sie auch wieder entschlüsseln. Allerdings erfordert die Verwendung des Benutzerprofils zusätzlichen Entwicklungsaufwand, wenn die DPAPI von einer ASP.NET-Anwendung verwendet werden soll, denn Sie müssen explizite Schritte zum Laden und Entladen des Benutzerprofils unternehmen (da ASP.NET Benutzerprofile nicht automatisch lädt).

- Der Computerspeicheransatz (der in dieser Vorgehensweise verwendet wird) ist einfacher zu entwickeln, da er keine Benutzerprofilverwaltung voraussetzt. Wird hierbei jedoch kein zusätzlicher Entropieparameter verwendet, ist dieser Ansatz weniger sicher, da jeder Benutzer des Computers Daten entschlüsseln kann. (Unter Entropie wird in diesem Zusammenhang ein Zufallswert verstanden, der das Dechiffrieren der Geheimdaten schwieriger macht.) Das Problem bei der Verwendung eines zusätzlichen Entropieparameters besteht darin, dass dieser Wert von der Anwendung ebenfalls sicher gespeichert werden muss, woraus sich wiederum ein Schlüsselverwaltungsproblem ergibt.

  Hinweis: Wenn Sie die DPAPI mit dem Computerspeicher verwenden, ist die verschlüsselte Zeichenfolge für den jeweiligen Computer spezifisch, daher müssen die verschlüsselten Daten auf jedem Computer erzeugt werden. Sie können die verschlüsselten Daten nicht auf andere Computer in einer Farm oder in einem Cluster kopieren.

- Weitere Informationen über die Verwendung der DPAPI mit dem Benutzerspeicher von einer ASP.NET-Webanwendung aus (unter Verwendung einer Serviced Component in einer Enterprise Services-Anwendung) finden Sie unter "Vorgehensweise: Verwenden von DPAPI (Benutzerspeicher) von ASP.NET aus mit Enterprise Services" im Abschnitt "Referenz" dieses Handbuchs.

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 in dieser Vorgehensweise erläuterten Verfahren setzen zudem Kenntnisse des Erstellens von ASP.NET-Webanwendungen mit dem Entwicklungstools Microsoft Visual C#™ voraus.

Bevor Sie diese Vorgehensweise durcharbeiten, müssen Sie die unter "Vorgehensweise: Erstellen einer DPAPI-Bibliothek" erläuterten Schritte unternehmen, um die verwaltete Klassenbibliothek für die DPAPI zu erstellen, die vom Code in dieser Vorgehensweise verwendet wird.

Zusammenfassung

Diese Vorgehensweise enthält folgende Verfahren:

1. Erstellen einer ASP.NET-Clientwebanwendung
2. Testen der Anwendung
3. Ändern der Webanwendung zum Lesen einer verschlüsselten Verbindungszeichenfolge aus **Web.config**.

1. Erstellen einer ASP.NET-Clientwebanwendung

In diesem Verfahren wird eine ASP.NET-Clientwebanwendung erstellt, die die DPAPI-Klassenbibliothek zum Ver- und Entschlüsseln von in der Datei **Web.config** gespeicherten Daten aufruft.

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

1. Starten Sie Visual Studio .NET, und erstellen Sie in C# eine neue ASP.NET-Webanwendung mit Namen **DPAPIClientWeb**.
2. Fügen Sie einen Verweis auf die Assembly **DataProtector.dll** hinzu, die mit dem unter "Vorgehensweise: Erstellen einer DPAPI-Bibliothek" erläuterten Verfahren erstellt wurde.

3. Öffnen Sie **WebForm1.aspx**, und fügen Sie am Anfang der Datei unterhalb der vorhandenen **using**-Anweisungen die folgenden **using**-Anweisungen hinzu.

```
using System.Text;
using DataProtection;
```

4. Fügen Sie **WebForm1.aspx** die in Tabelle 1 aufgeführten Steuerelemente hinzu.

   Tabelle 1: Steuerelemente für **WebForm1.aspx**

Typ des Steuerelements	Text	ID
Schaltfläche	Encrypt	btnEncrypt
Schaltfläche	Decrypt	btnDecrypt
Textfeld		txtDataToEncrypt
Textfeld		txtEncryptedData
Textfeld		txtDecryptedData
Bezeichnungsfeld		lblError
Bezeichnungsfeld	Data To Encrypt	
Bezeichnungsfeld	Encrypted Data	
Bezeichnungsfeld	Decrypted Data	

   Das Webformular sollte ähnlich dem in Abbildung 1 dargestellten aussehen.

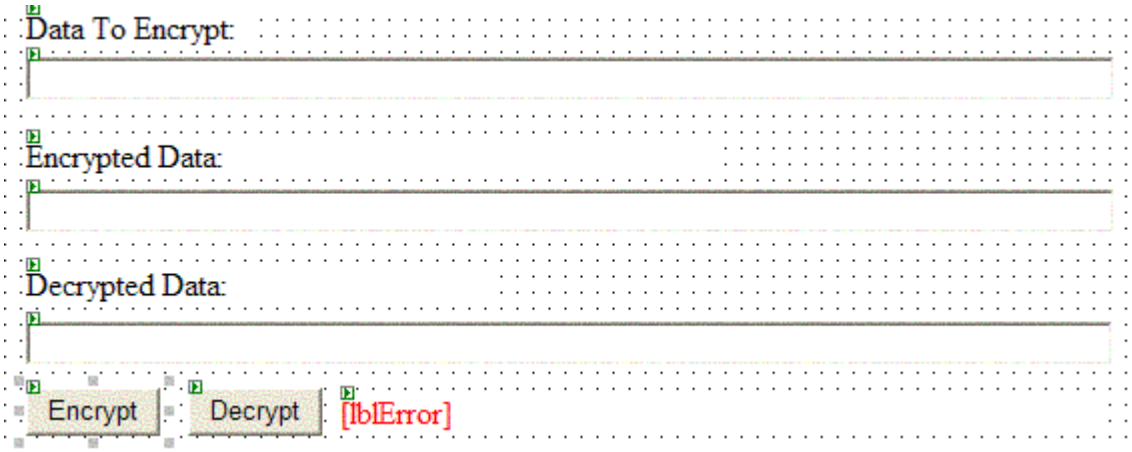

Abbildung 1
Webformular für DPAPIClientWeb

5. Doppelklicken Sie auf die Schaltfläche **Encrypt**, um einen Ereignishandler für das Klickereignis der Schaltfläche zu erstellen.

```
DataProtector dp = new DataProtector( DataProtector.Store.USE_MACHINE_STORE );
try
{
  byte[] dataToEncrypt = Encoding.ASCII.GetBytes(txtDataToEncrypt.Text);
  // Not passing optional entropy in this example
  // Could pass random value (stored by the application) for added security
  // when using DPAPI with the machine store.
  txtEncryptedData.Text =
               Convert.ToBase64String(dp.Encrypt(dataToEncrypt,null));
}
```

```
catch(Exception ex)
{
  lblError.ForeColor = Color.Red;
  lblError.Text = "Exception.<br>" + ex.Message;
  return;
}
lblError.Text = "";
```

6. Kehren Sie zum Webformular zurück, und doppelklicken Sie auf die Schaltfläche **Decrypt**. Fügen Sie dem Ereignishandler für das Klickereignis der Schaltfläche den folgenden Code hinzu.

```
DataProtector dp = new DataProtector(DataProtector.Store.USE_MACHINE_STORE);
try
{
  byte[] dataToDecrypt = Convert.FromBase64String(txtEncryptedData.Text);
  // Optional entropy parameter is null.
  // If entropy was used within the Encrypt method, the same entropy parameter
  // must be supplied here
  txtDecryptedData.Text =
                    Encoding.ASCII.GetString(dp.Decrypt(dataToDecrypt,null));
}
catch(Exception ex)
{
  lblError.ForeColor = Color.Red;
  lblError.Text = "Exception.<br>" + ex.Message;
  return;
}
lblError.Text = "";
```

7. Klicken Sie im Menü **Erstellen** auf **Projektmappe erstellen**.

2. Testen der Anwendung

In diesem Verfahren wird die Webanwendung getestet, um sicherzustellen, dass die Daten erfolgreich ver- und entschlüsselt werden.

▶ **So testen Sie die Anwendung**

1. Drücken Sie **STRG+F5**, um die Webanwendung auszuführen.
2. Geben Sie eine Zeichenfolge in das Textfeld **Data to Encrypt** ein, und klicken Sie dann auf **Encrypt**.

   Vergewissern Sie sich, dass die verschlüsselten Daten (Verschlüsselungsformat Base64) im Textfeld **Encrypted Data** angezeigt werden.
3. Klicken Sie auf die Schaltfläche **Decrypt**.

   Vergewissern Sie sich, dass die verschlüsselten Daten erfolgreich entschlüsselt wurden und im Textfeld **Decrypted Data** angezeigt werden.

3. Ändern der Webanwendung zum Lesen einer verschlüsselten Verbindungszeichenfolge aus Web.config.

Mit diesem Verfahren wird eine verschlüsselte Datenbank-Verbindungszeichenfolge in einem <**appSettings**>-Element in der Datei **Web.config** der Anwendung platziert. Anschließend fügen Sie Code zum Lesen und Entschlüsseln dieser Zeichenfolge aus der Konfigurationsdatei hinzu.

▶ **So ändern Sie die Webanwendung zum Lesen einer verschlüsselten Verbindungszeichenfolge aus Web.config.**

1. Kehren Sie zu Visual Studio .NET zurück, und zeigen Sie **WebForm1.aspx** im Entwurfsmodus an.
2. Fügen Sie dem Formular eine weitere Schaltfläche hinzu. Legen Sie die **Text**-Eigenschaft auf **Decrypt string from config file** und die **ID**-Eigenschaft auf **btnDecryptConfig** fest.
3. Doppelklicken Sie auf die Schaltfläche, um einen Ereignishandler für das Klickereignis der Schaltfläche zu erstellen.
4. Fügen Sie am Anfang der Datei unterhalb der vorhandenen **using**-Anweisungen die folgende **using**-Anweisung hinzu.

```
using System.Configuration;
```

5. Kehren Sie zum Ereignishandler **btnDecryptConfig_Click** zurück, und fügen Sie den folgenden Code hinzu, um eine Datenbank-Verbindungszeichenfolge aus dem Abschnitt <**appSettings**> der Datei **Web.config** abzurufen.

```
DataProtector dp = new DataProtector(DataProtector.Store.USE_MACHINE_STORE);
try
{
  string appSettingValue =
            ConfigurationSettings.AppSettings["connectionString"];
  byte[] dataToDecrypt = Convert.FromBase64String(appSettingValue);
  string connStr = Encoding.ASCII.GetString(
                                dp.Decrypt(dataToDecrypt,null));
  txtDecryptedData.Text = connStr;
}
catch(Exception ex)
{
  lblError.ForeColor = Color.Red;
  lblError.Text = "Exception.<br>" + ex.Message;
  return;
}
lblError.Text = "";
```

6. Klicken Sie im Menü **Erstellen** auf **Projektmappe erstellen**, um die Projekte neu zu erstellen.
7. Klicken Sie im Projektmappen-Explorer mit der rechten Maustaste auf **WebForm1.aspx**, und klicken Sie dann auf **In Browser anzeigen**.
8. Geben Sie eine Datenbank-Verbindungszeichenfolge wie die folgende in das Feld **Data to Encrypt** ein.

```
server=(local);Integrated Security=SSPI; database=Northwind
```

9. Klicken Sie auf die Schaltfläche **Encrypt**.
10. Markieren Sie den verschlüsselten Chiffrierungstext, und kopieren Sie ihn in die Zwischenablage.
11. Wechseln Sie zu Visual Studio.NET, öffnen Sie **Web.config**, und fügen Sie außerhalb des <**system.web**>-Elements das folgende <**appSettings**>-Element hinzu. Weisen Sie die verschlüsselte Verbindungszeichenfolge, die sich gegenwärtig in der Zwischenablage befindet, dem **value**-Attribut zu.

```
<appSettings>
   <add key="connectionString" value="encrypted connection string" />
</appSettings>
```

12. Speichern Sie die Datei **Web.config**.
13. Klicken Sie auf die Schaltfläche **Decrypt string from config file**, und vergewissern Sie sich, dass die verschlüsselte Datenbank-Verbindungszeichenfolge erfolgreich aus der Datei **Web.config** ausgelesen werden konnte und im Feld **Decrypted data** angezeigt wird.

Referenzen

Weitere Informationen über dieses Themenbereich finden Sie in den folgenden Vorgehensweisen im Abschnitt "Referenz" dieses Handbuchs:

- "Vorgehensweise: Erstellen einer DPAPI-Bibliothek"
- "Vorgehensweise: Verwenden von DPAPI (Benutzerspeicher) von ASP.NET aus mit Enterprise Services"