

Usando certificados en WCF de manera sencilla

Introducción

La seguridad tiene un papel importante en cualquier aplicación distribuida y Windows Communication Foundation (conocido como WCF), el nuevo marco de comunicación de Microsoft, implementa muchas normas de seguridad y tiene una amplia gama de características disponibles. Uno de los aspectos más importantes de la seguridad es la autenticación. WCF puede configurarse para utilizar muchos métodos de autenticación:

- Anónimo
- Nombre de usuario y contraseña
- Certificado
- Windows
- CardSpace

En este artículo le mostraré cómo configurar WCF con certificados para autenticar al servidor WCF mediante un enfoque alternativo.

Conceptos de fondo

Las secciones siguientes suponen que Ud. ya está familiarizado con muchos conceptos WCF y seguridad. Consulte la sección de recursos externos si desea revisar algunos de estos conceptos o para obtener más información.

El problema

El uso de certificados para autenticación no es nuevo, pero sigue siendo uno de la forma más común para autenticar a una persona/aplicación. WCF tiene una compatibilidad integrada para los certificados que cumplen los estándares de seguridad de servicios Web (WS-Security).

El problema con las configuraciones predeterminadas y ejemplos disponibles es que todos los certificados deben estar instalados en el almacén de certificados, que es básicamente una ubicación central donde Windows guarda todos los certificados (utilizados también para otras aplicaciones: Internet Explorer,...).

¿Por qué esta solución causa algunos problemas? La respuesta fácil es: porque no es fácil de configurar correctamente todos los certificados. Para más detalles:

Al implementar su servicio en el servidor debe instalar en el almacén de certificados todos los certificados utilizados (en diferentes lugares basadas en el uso de los certificados).

Esta operación debe ser ejecutada mediante un programa de instalación, un archivo de secuencia de comandos o un archivo por lotes. Por esta razón, es difícil implementar la aplicación mediante la instalación xcopy/ClickOnce.

Cada cliente también debe instalar el certificado utilizado para autenticarse siempre en el almacén de certificados. Esto es fácil si tienes un pequeño número de clientes pero muy difícil si debe configurar manualmente cada equipo (Además, para el cliente, puede utilizar una instalación de xcopy/ClickOnce).

Debe dar al proceso de ejecución (como ASP.NET) los permisos para leer la clave privada del certificado. Este paso generalmente requiere cambiar los permisos del sistema de archivos. Una vez más, esto requiere un archivo de secuencia de comandos o una instalación que no siempre es fácil.

Si utiliza un hosting compartido probablemente no es posible instalar certificados o cambiar los permisos de certificado. (Este fue el problema al que nos enfrentamos, al momento de implementar esta solución)

Como desarrollador me gusta tener cada proyecto aislado de los demás. Quiero ser capaz de probar diferentes configuraciones o aplicaciones, me gusta simplemente descargar la versión más reciente en el repositorio de código y ejecutarlo, sin ninguna configuración especial. Si uso el almacén de certificados, tengo que recordar siempre instalar o desinstalar los certificados cada vez.

En la siguiente página MSDN puede ver un ejemplo de configuración mediante certificados y una descripción de cómo instalar certificados mediante la solución clásica: [MSDN: certificado de seguridad de mensajes.](#)

Estas fueron las razones detrás de mi decisión para tratar de implementar un enfoque diferente el cual es descrito en las siguientes secciones.

La solución

Mi objetivo es encontrar una manera fácil de utilizar certificados sin usar almacén de certificados. WCF se puede extender fácilmente; en este artículo le mostraré cómo ampliar WCF para cargar los certificados de archivos.

Es sabido que almacenar certificados en el sistema de archivos es menos seguro, pero creo que teniendo el cuidado necesario, esta puede ser una alternativa útil. Consulte la sección desventajas para una discusión de los problemas de mi enfoque.

Considere que con mi solución, simplemente cambio como se cargan los certificados, todas las ventajas de utilizar WCF (estándares, código probado,...) siguen siendo válidas. Más importante es que se debe seguir utilizando la mayoría de los ajustes necesarios para utilizar certificados.

Detalles de implementación

Cargar un certificado desde un archivo es bastante fácil, simplemente debe utilizar la clase de **System.Security.Cryptography.X509Certificates.X509Certificate2**:

Vea la sección service model del archivo Web.Config del WCF service:

```
<system.serviceModel>
  <!--<serviceHostingEnvironment aspNetCompatibilityEnabled="true" />-->
  <bindings>
    <wsHttpBinding>
      <binding name="wsHttpEndpointBinding">
        <security>
          <message clientCredentialType="UserName" negotiateServiceCredential="true"
algorithmSuite="Basic128" establishSecurityContext="true" />
        </security>
      </binding>
    </wsHttpBinding>
  </bindings>
  <services>
    <service behaviorConfiguration="_WCFTest.WCFTestBehavior"
name="_WCFTest.WCFTestService">
      <endpoint address="" binding="wsHttpBinding"
bindingConfiguration="wsHttpEndpointBinding" bindingName="wsHttpEndpoint"
contract="_WCFTest.IWCFTestService" />
      <!--<endpoint address="mex" binding="mexHttpBinding"
contract="IMetadataExchange" />-->
    </service>
  </services>
</behaviors>
```

```

<serviceBehaviors>
  <behavior name="_WCFTTest.WCFTTestBehavior">
    <serviceThrottling maxConcurrentInstances="1000" maxConcurrentCalls="1000"
maxConcurrentSessions="1000" />
    <serviceMetadata httpGetEnabled="true" />
    <serviceDebug includeExceptionDetailInFaults="true" />
    <serviceAuthorization principalPermissionMode="UseAspNetRoles" />
    <serviceCredentials>
      <userNameAuthentication userNamePasswordValidationMode="MembershipProvider"
membershipProviderName="AspNetMembershipProvider" />
    </serviceCredentials>
  </behavior>
</serviceBehaviors>
</behaviors>
</system.serviceModel>

```

Este ejemplo además implementa Autenticación/Autorización usando ASP.NET Membership y Roles.

En el archivo .SVC preste atención a la cláusula Factory:

```

<%@ ServiceHost Language="VB" Debug="true" Service="_WCFTTEST.WCFTTESTService" Factory="_
WCFTTEST.WCFTTESTServiceFactory" CodeBehind="WCFTTEST.svc.vb" %>

```

En el archivo IWCFTTESTService.vb es donde se lee y carga el certificado:

```

Public Class WCFTTESTServiceFactory : Inherits ServiceHostFactory

  Private _baseAddressIndex As Integer =
CInt(ConfigurationManager.AppSettings.Get("BaseAddressIndex")) 'En el caso de Hosting
SoloWindows.com hay que poner este valor (2)
  Private _serverCertificate As X509Certificate2
  Private WithEvents _serviceHost As ServiceHost
  Private _archivo As String =
ConfigurationManager.AppSettings.Get("PathDelCertificado")

  Protected Overrides Function CreateServiceHost(ByVal serviceType As System.Type,
ByVal baseAddresses() As System.Uri) As ServiceHost
    Try
      _serverCertificate = New
X509Certificate2(AppDomain.CurrentDomain.SetupInformation.PrivateBinPath + "\" +
_archivo, "")
      _serviceHost = New ServiceHost(serviceType,
baseAddresses(_baseAddressIndex))
      AddHandler _serviceHost.Faulted, New EventHandler(AddressOf Me.Fallo)
      'Setear el certificado del servidor
      If Not _serverCertificate Is Nothing Then
serviceHost.Credentials.ServiceCertificate.Certificate = _serverCertificate
      'Retornar el host del servicio creado
      Return _serviceHost
    Catch ex As Exception
      Return New ServiceHost(serviceType, baseAddresses(_baseAddressIndex))
    End Try
  End Function

  Public Sub Fallo(ByVal sender As Object, ByVal e As System.EventArgs)
    'Error
  End Sub

  'Public Sub Cerrado(ByVal sender As Object, ByVal e As System.EventArgs) Handles
_serviceHost.Closed
  ' Interceptar evento
  'End Sub

```

```

    'Public Sub Cerrando(ByVal sender As Object, ByVal e As System.EventArgs) Handles
_serviceHost.Closing
        ' Interceptar evento
    'End Sub

    'Public Sub Abierto(ByVal sender As Object, ByVal e As System.EventArgs) Handles
_serviceHost.Opened
        ' Interceptar evento
    'End Sub

    'Public Sub Abriendo(ByVal sender As Object, ByVal e As System.EventArgs) Handles
_serviceHost.Opening
        ' Interceptar evento
    'End Sub

    'Public Sub MensajeDesconocidoRecibido(ByVal sender As Object, ByVal e As
System.EventArgs) Handles _serviceHost.UnknownMessageReceived
        ' Interceptar evento
    'End Sub
End Class

```

Como se muestra en la parte resaltada en verde, se carga el certificado utilizando unas keys de configuración. El primer parámetro es la ruta del archivo del certificado, el segundo parámetro es la contraseña utilizada para cifrar la clave privada (si existe).

```

<appSettings>
  <add key="PathDelCertificado" value="WCFTEST.pfx" />
  <add key="BaseAddressIndex" value="2" />
</appSettings>

```

Con la clase X509Certificate2 puede cargar dos tipos de archivos:

- .cer - Se utiliza para almacenar una clave pública
- .pfx - Se utiliza para almacenar una clave pública-privada (opcionalmente encriptado con una contraseña)

Puede obtener estos archivos de una entidad de certificación pública o crear sus certificados auto firmados mediante makecert.exe y pvk2pfx.exe, ambos disponibles como Visual Studio Tools. Aquí un ejemplo sobre cómo crear un certificado:

```
makecert -r -pe -n "CN=WCFTEST" -b 01/01/2007 -e 01/01/2030 -sky exchange WCFTEST.cer -sv WCFTEST.pvk
```

```
pvk2pfx.exe -pvk WCFTEST.pvk -spc WCFTEST.cer -pfx WCFTEST.pfx
```

El primer comando (makecert) genera una clave pública (en este caso WCFTEST.cer) y una clave privada (en este caso WCFTEST.pvk). El segundo comando (pvk2pfx) combina los 2 archivos en un .pfx único (en este caso WCFTEST.pfx). Al ejecutar makecert y pvk2pfx, puede insertar una contraseña para cifrar la clave privada.

Servicio de autenticación

Esta sección describe cómo configurar el servidor con el certificado utilizado para autenticar el servicio.

Debe generar el certificado de servicio y ponerlo en un directorio seguro. Por lo general se puede especificar el certificado para el servicio dentro de la sección de System.ServiceModel del archivo config como este:

```

<behaviors>
  <serviceBehaviors>
    <behavior name="serviceCredentialBehavior">

```

```

    <serviceCredentials>
      <serviceCertificate findValue="Contoso.com"
        storeLocation="LocalMachine"
        storeName="My"
        x509FindType="FindBySubjectName" />
    </serviceCredentials>
  </behavior>
</serviceBehaviors>
</behaviors>

```

Esta configuración básicamente le indica a WCF en dónde encontrar el certificado dentro del almacén de certificados. Para cargar el archivo de certificado, por las restricciones mencionadas anteriormente, no se puede simplemente cambiar la configuración. Para cargar el certificado se debe establecer la propiedad de ServiceHost.Credentials.ServiceCertificate.Certificate como se muestra en el código resaltado en rojo más arriba.

Nota: Cuando cree el cliente proxy con svcutil verá una sección de identidad dentro de la sección del endpoint:

```

<identity>
  <certificate encodedValue="...." />
</identity>

```

Esta sección debe crearse cada vez que cambia el certificado de servicio, ya que contiene la identificación pública del certificado generada en tiempo de diseño. Este valor se utiliza por el cliente para estar seguro para hablar con el servicio que se espera.

A continuación se muestra una parte del archivo de configuración del cliente

```

<system.serviceModel>
  <bindings>
    <wsHttpBinding>
      <binding name="wsHttpEndpoint_IWCFTESTService" closeTimeout="00:10:00"
        openTimeout="00:10:00" receiveTimeout="00:10:00" sendTimeout="00:10:00"
        bypassProxyOnLocal="false" transactionFlow="false"
        hostNameComparisonMode="StrongWildcard"
        maxBufferPoolSize="5242880" maxReceivedMessageSize="655360"
        messageEncoding="Text" textEncoding="utf-8" useDefaultWebProxy="true"
        allowCookies="false">
        <readerQuotas maxDepth="32" maxStringContentLength="81920"
        maxArrayLength="163840"
          maxBytesPerRead="40960" maxNameTableCharCount="163840" />
        <reliableSession ordered="true" inactivityTimeout="00:10:00"
          enabled="false" />
        <security mode="Message">
          <!--<transport clientCredentialType="Windows" proxyCredentialType="None"
            realm="" />-->
          <message clientCredentialType="UserName" negotiateServiceCredential="true"
            algorithmSuite="Basic128" establishSecurityContext="true" />
        </security>
      </binding>
    </wsHttpBinding>
  </bindings>
  <client>
    <endpoint address="http://www.WCFTEST.com/WCFTESTService.svc"
      binding="wsHttpBinding" behaviorConfiguration="CertificadoTemporalBehavior"
      bindingConfiguration="wsHttpEndpoint_I50ConsultoresService"
      contract=" WCFTEST.IWCFTESTService" name="wsHttpEndpoint_IWCFTESTService">
      <identity>
        <!--<dns value="Server"/>-->
        <certificate
        encodedValue="AwAAAAEAAAAUAAAAOqIK4KIAJALt4MbWqCyPD4ghcEIgAAAAAQAAABgCAAawggIUMIIBfaADA
        gECAhCmBz4rkoH2nEVt7nKjiNTSMA0GCSqGSIB3DQEBBAUAMB4xHDAaBgNVBAMTEzUwQ29uc3VsdG9yZXNTZXJ2

```

```

ZXIwHhcNMDcwMTAxMDIwMDAwWhcNMzAwMTAxMDIwMDAwWjAeMRwwGgYDVQQDEXMlMENvbnNlbHRvcmlvZmVydmlvYmIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDk8dOK0xJ5lWwv0E+nr7pHYgl2QmxYR4wD/oa1UiF7YQwvle
WaeTkML7WwczLmovxEfYodJM6EjJI5KeACb4cVYGW6GjvWSOohNmuP2X35ulK55APqTqbXb6sbMzoA2K+ykjoET
GX0WqNMPof1m4FXVxcJC3SCZQhPkD7fVLGz3QIDAQABo1MwUTBPBgNVHQEESDBGgBDPF7J8EXAYBcoOI3J5HbX
oSAwHjEcMBoGA1UEAxMTNTBDb25zdWx0b3Jlc1NlcnZlcoIQpgc+K5KB9pxFbe5yo4jU0jANBkqkqkiG9w0BAQQ
FAAOBgQCUCOVct4U+UaHb9QDpAKOvCfxyPilHoiaLWxcuTqFIZL7ZKVRT4UeGbG1FMY9U0beUeoLJE9DJhhw2W1
FeWMfLUCc8zZtkNQJlaotg8hEqC0SRMKUDcDtlmFdFRBZhJOXEBw6ICfzGvRC9uYPZeV31YtqK1gvPkPeNkauad
D0zsQ==" />
</identity>
</endpoint>
</client>
<behaviors>
  <endpointBehaviors>
    <behavior name="CertificadoTemporalBehavior">
      <clientCredentials>
        <serviceCertificate>
          <!--Setear certificateValidationMode a "None" para no chequear si el
certificado del servidor es de confianza, asi podemos utilizar certificados nuestros.
          ATENCION: Hay que setear esta propiedad a "ChainTrust" o
"PeerOrChainTrust" cuando se use un certificado prolijo-->
          <authentication revocationMode="NoCheck"
certificateValidationMode="None"/>
        </serviceCertificate>
      </clientCredentials>
    </behavior>
  </endpointBehaviors>
</behaviors>
<diagnostics>
  <!--Enable Message Logging here
  log all messages received or sent at the transport or service model levels-->
  <messageLogging logEntireMessage="true"
    maxMessagesToLog="300"
    logMessagesAtServiceLevel="true"
    logMalformedMessages="true"
    logMessagesAtTransportLevel="true" />
</diagnostics>
</system.serviceModel>
<system.web>
  <membership defaultProvider="AspNetMembershipProvider">
    <providers>
      <clear/>
      <add name="AspNetMembershipProvider" connectionStringName="LYA.
ConnectionString" passwordFormat="Hashed" enablePasswordRetrieval="false"
enablePasswordReset="true" applicationName="WCFTEST"
type="System.Web.Security.SqlMembershipProvider"/>
    </providers>
  </membership>
  <roleManager enabled="true" defaultProvider="AspNetSqlRoleProvider">
    <providers>
      <clear/>
      <add name="AspNetSqlRoleProvider" type="System.Web.Security.SqlRoleProvider"
connectionStringName="LYA.ConnectionString" applicationName="WCFTEST"/>
    </providers>
  </roleManager>
</system.web>
<system.diagnostics>
  <sources>
    <source name="System.ServiceModel" switchValue="Information,ActivityTracing"
propagateActivity="true">
      <listeners>
        <add name="xml" />
      </listeners>
    </source>
    <source name="System.ServiceModel.MessageLogging">

```

```

        <listeners>
            <add name="xml" />
        </listeners>
    </source>
</sources>
<sharedListeners>
    <add initializeData="C:\WCFTESTCliente.svclog"
type="System.Diagnostics.XmlWriterTraceListener"
    name="xml" />
</sharedListeners>
<trace autoflush="true" />
</system.diagnostics>

```

Finalmente parte del código de cliente para consumir el servicio:

```

Public Function ObtenerBaseDeDatos() As WCFTEST.IWCFTESTServiceClient
    Try
        Dim _wcf As New WCFTEST.IWCFTESTServiceClient
        With _wcf
            .ClientCredentials.UserName.UserName =
PropiedadesComunes.Usuario.UserName
            .ClientCredentials.UserName.Password =
PropiedadesComunes.Usuario.Password
            .Open()
        End With
        Return _wcf
    Catch ex As Exception
    End Try
End Function

Public Function GuardarCliente(ByVal cliente As Cliente) As Long
    Try
        'Variable que representa el proxy del WCF
        Using _db As WCFTEST.IWCFTESTServiceClient = ObtenerBaseDeDatos()
            Return _db.GuardarCliente(cliente)
        End Using
    Catch ex As Exception
        If ExceptionPolicy.HandleException(ex, PoliticaDeCapaDeAccesoADatos) Then Throw
        Return 0
    End Try
End Function

```

Desventajas

El almacenamiento de los certificados en archivos es más fácil, pero es menos seguro debido a que un usuario malicioso puede robar la clave privada. La clave pública puede compartirse sin problemas, pero si alguien roba la clave privada verá comprometida su seguridad. Con la clave privada un usuario malicioso puede descifrar los mensajes o suplantar a su servicio o un cliente.

Creo que en cualquier caso, si alguien puede tener acceso a su sistema de archivo probablemente tenga otros problemas más graves de seguridad. Tenga en cuenta también que si alguien tiene acceso a su sistema de archivo, él o ella puede probablemente (depende del tipo de ataque) también tener acceso al almacén de certificados.

Mi sugerencia es evaluar correctamente la aplicación y los requisitos de seguridad. En muchos casos, creo que guardar certificado en un archivo es bastante seguro y puede ser útil.

Referencias Externas

<http://msdn.microsoft.com/en-us/netframework/aa663324.aspx>

<http://msdn.microsoft.com/en-us/library/ms731049.aspx>

www.yasistemas.com



Comunidad de Usuarios Microsoft Uruguay



2009

L&A SISTEMAS

www.lyasistemas.com

513-76-13

www.lyasistemas.com