How to: Use Transport Security and Message Credentials

.NET Framework (current version)

Securing a service with both transport and message credentials uses the best of both Transport and Message security modes in Windows Communication Foundation (WCF). In sum, transport-layer security provides integrity and confidentiality, while message-layer security provides a variety of credentials that are not possible with strict transport security mechanisms. This topic shows the basic steps for implementing transport with message credentials using the WSHttpBinding and NetTcpBinding bindings. For more information about setting the security mode, see How to: Set the Security Mode.

When setting the security mode to `TransportWithMessageCredential`, the transport determines the actual mechanism that provides the transport-level security. For HTTP, the mechanism is Secure Sockets Layer (SSL) over HTTP (HTTPS); for TCP, it is SSL over TCP or Windows.

If the transport is HTTP (using the WSHttpBinding), SSL over HTTP provides the transport-level security. In that case, you must configure the computer hosting the service with an SSL certificate bound to a port, as shown later in this topic.

If the transport is TCP (using the NetTcpBinding), by default the transport-level security provided is Windows security, or SSL over TCP. When using SSL over TCP, you must specify the certificate using the SetCertificate method, as shown later in this topic.

To use the WSHttpBinding with a certificate for transport security (in code)

1. Use the HttpCfg.exe tool to bind an SSL certificate to a port on the machine. For more information, seeHow to: Configure a Port with an SSL Certificate.
2. Create an instance of the WSHttpBinding class and set the Mode property to TransportWithMessageCredential.
3. Set the ClientCredentialType property to an appropriate value. (For more information, seeSelecting a Credential Type.) The following code uses the Certificate value.
4. Create an instance of the Uri class with an appropriate base address. Note that the address must use the "HTTPS" scheme and must contain the actual name of the machine and the port number that the SSL certificate is bound to. (Alternatively, you can set the base address in configuration.)
5. Add a service endpoint using the AddServiceEndpoint method.
6. Create the instance of the ServiceHost and call the Open method, as shown in the following code.

C#

```
            WSHttpBinding b = new WSHttpBinding();
            b.Security.Mode = SecurityMode.TransportWithMessageCredential;
            b.Security.Message.ClientCredentialType =
MessageCredentialType.Certificate;

            // The SSL certificate is bound to port 8006 using the HttpCfg.exe tool.
            Uri httpsAddress = new Uri("https://localMachineName:8006/base");
            ServiceHost sh = new ServiceHost(typeof(Calculator), httpsAddress);
            sh.AddServiceEndpoint(typeof(ICalculator), b, "HttpsCalculator");
            sh.Open();
```

```
            Console.WriteLine("Listening");
            Console.ReadLine();
```

To use the NetTcpBinding with a certificate for transport security (in code)

1. Create an instance of the NetTcpBinding class and set the Mode property to TransportWithMessageCredential.
2. Set the ClientCredentialType to an appropriate value. The following code uses the Certificate value.
3. Create an instance of the Uri class with an appropriate base address. Note that the address must use the "net.tcp" scheme. (Alternatively, you can set the base address in configuration.)
4. Create the instance of the ServiceHost class.
5. Use the SetCertificate method of the X509CertificateRecipientServiceCredential class to explicitly set the X.509 certificate for the service.
6. Add a service endpoint using the AddServiceEndpoint method.
7. Call the Open method, as shown in the following code.

C#

```
            NetTcpBinding b = new
NetTcpBinding(SecurityMode.TransportWithMessageCredential);
            b.Security.Message.ClientCredentialType =
MessageCredentialType.Certificate;
            Uri netTcpAdddress = new Uri("net.tcp://baseAddress");
            ServiceHost sh = new ServiceHost(typeof(Calculator), netTcpAdddress);
            sh.Credentials.ServiceCertificate.SetCertificate(
                StoreLocation.LocalMachine, StoreName.My,
                X509FindType.FindByIssuerName, "Contoso.com");
            sh.AddServiceEndpoint(typeof(ICalculator), b, "TcpCalculator");
            sh.Open();
            Console.WriteLine("Listening");
            Console.ReadLine();
```

To use the NetTcpBinding with Windows for transport security (in code)

1. Create an instance of the NetTcpBinding class and set the Mode property to TransportWithMessageCredential.
2. Set the transport security to use Windows by setting the ClientCredentialType to Windows. (Note that this is the default.)
3. Set the ClientCredentialType to an appropriate value. The following code uses the Certificate value.
4. Create an instance of the Uri class with an appropriate base address. Note that the address must use the "net.tcp" scheme. (Alternatively, you can set the base address in configuration.)
5. Create the instance of the ServiceHost class.
6. Use the SetCertificate method of the X509CertificateRecipientServiceCredential class to explicitly set the X.509 certificate for the service.

7. Add a service endpoint using the AddServiceEndpoint method.

8. Call the Open method, as shown in the following code.

C#

```
            NetTcpBinding b = new
NetTcpBinding(SecurityMode.TransportWithMessageCredential);
            b.Security.Transport.ClientCredentialType =
TcpClientCredentialType.Windows;
            b.Security.Message.ClientCredentialType =
MessageCredentialType.Certificate;
            Uri netTcpAdddress = new Uri("net.tcp://Tcp");
            ServiceHost sh = new ServiceHost(typeof(Calculator), netTcpAdddress);
            sh.Credentials.ServiceCertificate.SetCertificate(
                StoreLocation.LocalMachine, StoreName.My,
                X509FindType.FindByIssuerName, "Contoso.com");
            sh.AddServiceEndpoint(typeof(ICalculator), b, "TcpCalculator");
            sh.Open();
            Console.WriteLine("Listening");
            Console.ReadLine();
```

Using Configuration

To use the WSHttpBinding

1. Configure the computer with an SSL certificate bound to a port. (For more information, seeHow to: Configure a Port with an SSL Certificate). You do not need to set a `<transport>` element value with this configuration.

2. Specify the client credential type for the message-level security. The following example sets the `clientCredentialType` attribute of the `<message>` element to `UserName`.

```
<wsHttpBinding>
<binding name="WsHttpBinding_ICalculator">
        <security mode="TransportWithMessageCredential" >
           <message clientCredentialType="UserName" />
        </security>
</binding>
</wsHttpBinding>
```

To use the NetTcpBinding with a certificate for transport security

1. For SSL over TCP, you must explicitly specify the certificate in the `<behaviors>` element. The following example specifies a certificate by its issuer in the default store location (local machine and personal stores).

```
<behaviors>
 <serviceBehaviors>
   <behavior name="mySvcBehavior">
       <serviceCredentials>
```

```
          <serviceCertificate findValue="contoso.com"
                              x509FindType="FindByIssuerName" />
        </serviceCredentials>
    </behavior>
  </serviceBehaviors>
</behaviors>
```

2. Add a <netTcpBinding> to the bindings section
3. Add a binding element, and set the `name` attribute to an appropriate value.
4. Add a `<security>` element, and set the `mode` attribute to `TransportWithMessageCredential`.
5. Add a `<message>` element, and set the `clientCredentialType` attribute to an appropriate value.

```
<bindings>
<netTcpBinding>
  <binding name="myTcpBinding">
    <security mode="TransportWithMessageCredential" >
       <message clientCredentialType="Windows" />
    </security>
  </binding>
</netTcpBinding>
</bindings>
```

To use the NetTcpBinding with Windows for transport security

1. Add a <netTcpBinding> to the bindings section,
2. Add a `<binding>` element and set the `name` attribute to an appropriate value.
3. Add a `<security>` element, and set the `mode` attribute to `TransportWithMessageCredential`.
4. Add a `<transport>` element and set the `clientCredentialType` attribute to `Windows`.
5. Add a `<message>` element and set the `clientCredentialType` attribute to an appropriate value. The following code sets the value to a certificate.

```
<bindings>
<netTcpBinding>
  <binding name="myTcpBinding">
    <security mode="TransportWithMessageCredential" >
       <transport clientCredentialType="Windows" />
       <message clientCredentialType="Certificate" />
    </security>
  </binding>
</netTcpBinding>
</bindings>
```

See Also

How to: Set the Security Mode
Securing Services
Securing Services and Clients