Message and Transport Security

Contents

Objectives

- Understand the differences between message security and transport security in WCF.
- Understand how to use message and transport security together.
- Understand the benefits and tradeoffs involved in using each security type.
- Understand how to decide when to use each option.

Overview

When working with WCF or Web services, securing communication between the client and the service is very important. *Transfer security* is concerned with guaranteeing the integrity and confidentiality of WCF service messages as they flow from application to application across the network. Use encryption to enforce confidentiality and protect your messages from eavesdropping. Use integrity checks, such as a signature-based checksum, to protect your message from tampering.

In WCF, transfer security is also responsible for providing authentication. In the context of WCF, *authentication* refers to mutual authentication, where clients are not only uniquely identified to the service, but the service is also uniquely identified to the client.

Transfer security in WCF is achieved through the use of either transport security or message security.

Transport Security

When using transport security, the user credentials and claims are passed by using the transport layer. In other words, user credentials are transport-dependent, which allows fewer authentication options compared to message security. Each transport protocol (TCP, IPC, MSMQ, or HTTP) has its own mechanism for passing credentials and handling message protection. The most common approach for this is to use Secure Sockets Layer (SSL) for encrypting and signing the contents of the packets sent over Secure HTTP (HTTPS).

Transport security is used to provide point-to-point security between the two endpoints (service and client). If there are intermediary systems between client and the service, each intermediate point must forward the message over a new SSL connection.

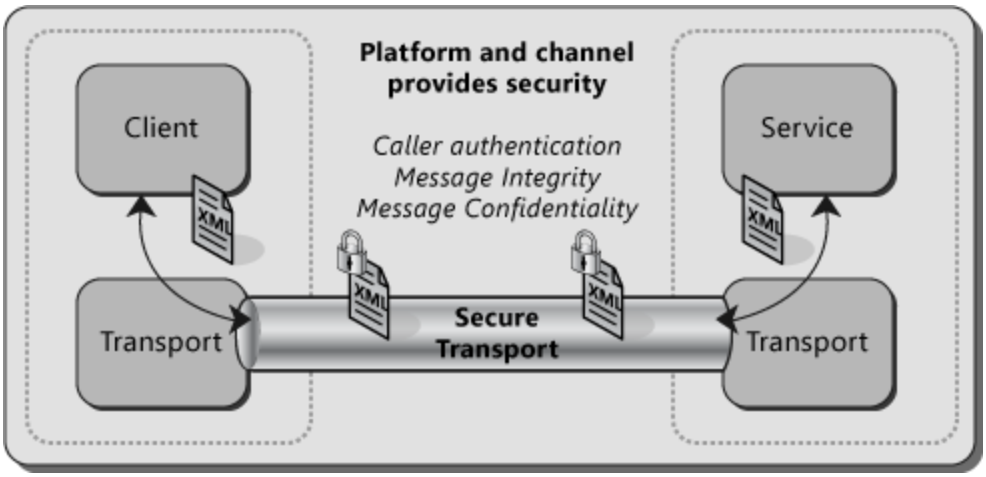

Figure 1

Transport Security

Use transport security in the following scenarios:

- You are sending a message directly from your application to a WCF service and the message will not be routed through intermediate systems.
- Both the service and the client are located in an intranet.

Using transport security offers the following advantages:

- It provides interoperability, meaning that communicating parties do not need to understand WS-Security specifications.
- It may result in better performance.
- Hardware accelerators can be used to further improve the performance.

Using transport security has the following disadvantages:

- Security is applied on a point-to-point basis, with no provision for multiple hops or routing through intermediate application nodes.
- It supports a limited set of credentials and claims compared to message security.
- It is transport-dependent upon the underlying platform, transport mechanism, and security service provider, such as NTLM or Kerberos.

Message Security

When using *message security*, the user credentials and claims are encapsulated in every message using the WS-Security specification to secure messages. This option gives the most flexibility from an authentication perspective. You can use any type of security credentials you want, largely independent of transport, as long as both the client and service agree.

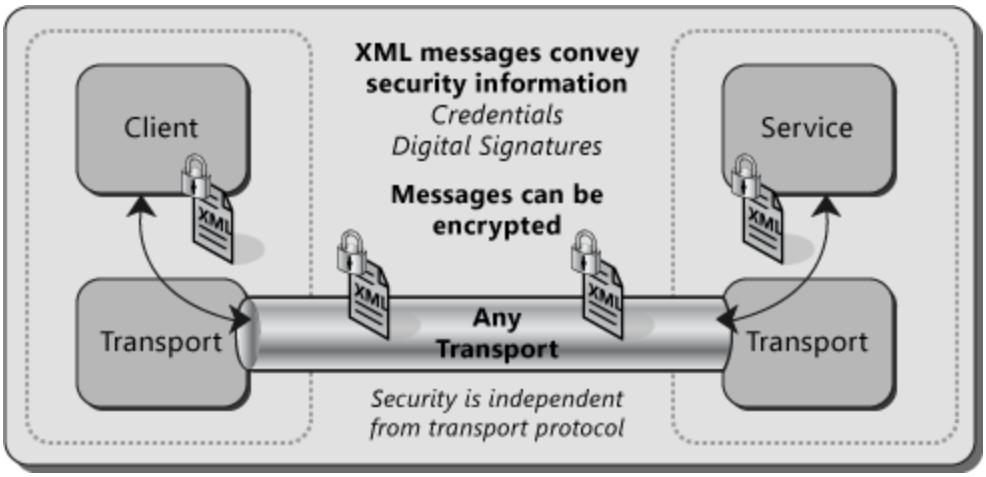

Figure 2

Message Security

Use message security in the following scenarios:

- You are sending a message to a WCF service, and the message is likely to be forwarded to other WCF services or may be routed through intermediate systems.
- Your WCF clients are accessing the WCF service over the Internet and messages may be routed through intermediate systems.

Using message security offers the following advantages:

- It provides end-to-end security. Because message security directly encrypts and signs the message, having intermediaries does not break the security.
- It allows partial or selective message encryption and signing, thus improving overall application performance.
- Message security is transport-independent and therefore can be used with any transport protocol.
- It supports a wide set of credentials and claims, including the issue token that enables federated security.

Using message security has following disadvantages:

- This option may reduce performance compared to transport security because each individual message is encrypted and signed.
- It does not support interoperability with older ASMX clients, as it requires both the client and service to support WS-Security specifications.

Transfer Security Modes

In WCF, you have two primary choices for providing transfer security: either you provide the transfer security on the transport level, or on the message level. Each option has its own advantages and disadvantages. The following table details the security modes available across most of the standard bindings.

Mode	Description

None	No security is provided; you should not use this option.
Transport	Mutual authentication and message protection are provided at the transport level.
Message	Mutual authentication and message protection are provided at the message level.
Both	Mutual authentication and message protection are provided at both the transport and message level. This is far more than is needed for most scenarios.
TransportWithMessageCredential	Client authentication is provided at the message level, and message protection and service authentication are provided at the transport level.
TransportCredentialOnly	Mutual authentication is provided at the transport level, but no message protection is provided. This option is available only on **BasicHttpBinding**.

Transport Security in WCF

In WCF, transport security depends on the binding and subsequent transport being used. Each protocol (TCP, HTTP, MSMQ, NamedPipes) has its own mechanism for passing credentials and handling message protection. Typically, you can use transport security when your client is deployed within an intranet, as it provides point-to-point security and better performance compared to message security. Note the following considerations:

- With transport security, the service credentials are negotiated by default.
- Transport security is available on all of the bindings except for **wsDualHttpBinding**.
- When using HTTP bindings, the WCF service typically is hosted in Internet information Services (IIS) and the transport security is provided by SSL. The SSL certificate is used to provide the message protection.
- With **netTcpBinding**, when using Windows authentication, the binding uses the service's Windows token to provide message protection. When using non-Windows authentication such as certificate authentication, you have to configure a service certificate as service credentials. The binding uses the service certificate for message protection.

Use the <**Security mode**> attribute to configure transport security on your binding. The following example shows how a **wsHttpBinding** binding is configured to use transport security:

```
...
<bindings>
  <wsHttpBinding>
    <binding name="wsHttpEndpointBinding">
      <security mode="Transport">
      </security>
    </binding>
  </wsHttpBinding>
</bindings>
```

...

Intranet Scenarios

The following are the authentication types and bindings that can be used in a typical intranet scenario:

- **Windows authentication with netTcpBinding**. By default, **netTcpBinding** uses Windows authentication and transport security. It uses the service account's Windows identity token to provide message protection. The credentials are negotiated with the Security Support Provider Interface (SSPI).
- **Certificate authentication with netTcpBinding**. By default, **netTcpBinding** uses transport security, which means you will have to configure the client credentials to use a certificate. To provide message protection at the transport level, you will have to configure a service certificate as service credentials. The certificate will negotiate a session key and service public key during the handshake, which will allow you to encrypt the content with the service certificate public key and sign the content with the private session key.

Note:

In an intranet scenario, it is recommended that you use **netTcpBinding** unless you have a specific requirement to use other bindings such as **wsHttpBinding**.

By default, **netTcpBinding** uses binary encoding and transport security, which delivers better performance.

Internet Scenarios

The following are the authentication types and bindings that can be used in a typical Internet scenario:

- **Basic authentication with basicHttpBinding**. By default, **basicHttpBinding** does not support any security, so you will need to configure the binding to use transport security. This is a good option when you want to support interoperability with non-WCF or non-Windows clients. In this scenario, you need to install a SSL certificate on IIS and then configure the virtual directory to require SSL. SSL will then negotiate a session key and service public key during the handshake, which will allow you to encrypt the content with the service certificate public key and sign the content with the private session key.
- **Certificate authentication with wsHttpBinding**. By default, **wsHttpBinding** uses message security and Windows authentication, so you will have to configure the binding to use transport security and configure the client credentials to use the certificate. To provide message protection at the transport level, install an SSL certificate on IIS and configure the virtual directory to require SSL.

Note:

In an Internet scenario, you can only use the **HttpBinding** option.

Message Security in WCF

Message security uses the WS-Security specification to secure messages. The specification describes enhancements to Simple Object Access Protocol (SOAP) messaging to ensure confidentiality, integrity, and authentication at the SOAP message level (instead of the transport level). Typically, you can use transport security when your client is deployed over the Internet, as it provides end-to-end security.
With transport security, the service credentials are negotiated by default, but you can configure the message security to avoid service credential negotiation if you want to restrict clients from accessing your service. This is especially important when you are in partner scenario where your service is exposed to a number of clients. When you configure message security to not negotiate credentials, you have to make sure that the service credentials are available out-of-band to the client application.
Message security is available on all of the bindings except
for **netNamedPipeBinding** and **MSmqIntegrationBinding**.
When using Windows authentication, message security uses the service's Windows token to provide message security. When using non-Windows authentication such as username, certificate, or issue token authentication, you have to configure a service certificate as service credentials. Message security uses the service certificate for message protection.
Use the <**Security mode**> attribute to configure message security on your binding. The following example shows **netTcpBinding** configured to use message security:

```
…
<bindings>
  <wsHttpBinding>
    <binding name="netTcpEndpointBinding">
      <security mode="message">
      </security>
    </binding>
  </wsHttpBinding>
</bindings>
…
```

By default, message security encrypts and signs the messages. Although it is not recommended, with message security you can lower the protection level or disable it based on your requirements.

Protection Level

You can use the **[ServiceContract(ProtectionLevel)]** attribute to specify message security protection levels on the interface or operation level. The available protection level options are:

- **None**. Use **None** to turn off signing and encryption on the operation or interface.
- **Sign**. Use **Sign** to sign the interface or operation but not encrypt it.
- **EncryptAndSign**. Use **EncryptAndSign** to both encrypt and sign the interface or operation.

The following code snippet creates an interface with the protection level set to **Sign**:

```
[ServiceContract(ProtectionLevel=ProtectionLevel.Sign]
public interface IService
{
string GetData(int value);
}
```

The following code snippet specifies an operation with the protection level set to **Sign**:

```
[OperationContract(ProtectionLevel=ProtectionLevel.Sign]
string GetData(int value);
```

Intranet Scenarios

Message security is not the best choice in an intranet scenario, but if your requirements force you to use message security, the following authentication types and bindings can be used in a typical intranet scenario:

- **Windows authentication with netTcpBinding**. By default, **netTcpBinding** uses Windows authentication and transport security. You will have to configure the binding to use message security. The binding uses the service account's Windows identity token to provide message protection. The credentials are negotiated with SSPI.
- **Certificate authentication with netTcpBinding**. You will have to configure the binding to use message security and configure the client credentials to use the certificate. To provide message protection at the message level, you will have to configure a service certificate as the service credentials. The certificate will negotiate a session key and service public key during the handshake, which will allow you to encrypt the content with the service certificate public key and sign the content with the private session key.
- **Username authentication with netTcpBinding**. You will have to configure the binding to use message security and configure the client credentials to use username authentication. To provide message protection at the message level, you need to install and configure a service certificate as service credentials.

Note:

In an intranet scenario, it is recommended that you use **netTcpBinding** unless you have a specific requirement to use other bindings such as **wsHttpBinding**.

By default, **netTcpBinding** uses binary encoding and transport security, which may improve the performance of your service.

Internet Scenarios

Message security is the preferred option for Internet scenarios. The following are the authentication types and bindings that can be used in a typical Internet scenario:

- **Basic authentication with basicHttpBinding**. By default, **basicHttpBinding** does not support any security, so you will have to configure the binding to use message security. Using this option does not allow you to support interoperability. In this scenario, you need to install and configure a service certificate as service credentials. The certificate will negotiate a session key and service public key during the handshake, which will allow you to encrypt the content with the service certificate public key and sign the content with the private session key.
- **Certificate authentication with wsHttpBinding**. By default, **wsHttpBinding** uses message security and Windows authentication, so you will have to configure the client credentials to use the certificate. To provide message protection at the message level, install and configure a service certificate as service credentials.

Note:

In an Internet scenario, you can only use the **HttpBinding** option.