MyID
Version 11.5

Securing Websites and Web Services

Copyright

Conventions Used in this Document

1. Lists:

    ◆ Numbered lists are used to show the steps involved in completing a task when the order is important

    ◆ Bulleted lists are used when the order is unimportant or to show alternatives

2. **Bold** is used for menu items and for labels.

    For example:

    ◆ "Record a valid email address in **'From' email address**"

    ◆ Select **Save** from the **File** menu

3. *Italic* is used for emphasis and to indicate references to other sections within the current document:

    For example:

    ◆ "Copy the file *before* starting the installation"

    ◆ "See *Issuing a Card* for further information"

4. ***Bold and italic*** are used to identify the titles of other documents.

    For example: "See the ***Release Notes*** for further information."

    Unless otherwise explicitly stated, all referenced documentation is available on the product media.

5. A `fixed width` font is used where the identification of spaces is important, including filenames, example SQL queries and any entries made directly into configuration files or the database.

6. **Notes** are used to provide further information, including any prerequisites or configuration additional to the standard specifications.

    For example:

    Note: This issue only occurs if updating from a previous version.

7. Warnings are used to indicate where failure to follow a particular instruction may result in either loss of data or the need to manually configure elements of the system.

    For example:

Warning:	You must take a backup of your database before making any changes to it.

Contents

1 Introduction

This document provides information that will help you secure your MyID® websites and web services.

MyID consists of a website and a number of web services hosted in Internet Information Services (IIS). IIS is part of Windows and provides security features to meet privacy and authentication requirements for MyID websites and web services. However, IIS needs to be configured to meet these requirements. The actual configuration needed depends on your installation of MyID.

IIS is a large product, created by Microsoft, and as such full documentation is available from Microsoft. The purpose of this guide is to assist configuring the security aspects of IIS to configure SSL and authentication features of IIS. Information in this guide is based on IIS8.5 and IIS10.0.

Where your MyID installation consists of multiple web servers, this configuration must be repeated for all MyID web servers in the MyID installation.

1.1 Change history

Version	Description
IMP1966-01	Released with MyID 11.0.
IMP1966-02	Released with MyID 11.1.
IMP1966-03	Released with MyID 11.2.
IMP1966-04	Released with MyID 11.3.
IMP1966-05	Released with MyID 11.4.
IMP1966-06	Released with MyID 11.5.

2 Configuring SSL/TLS (HTTPS)

Standard SSL/TLS (one-way SSL/TLS) provides privacy but not authentication. This means that it stops an adversary on the network from examining network traffic between the client and the server.

SSL must be configured and 'required' for all websites and web services associated with MyID (that is, clients must be forced to connect with HTTPS and should be unable to connect with HTTP).

To configure SSL/TLS:

1. Enroll a server certificate.

    a) In IIS manager, at the web server level, select **Server Certificates**.

    b) This opens a window that allows enrollment of the server certificates that represent the identity of the web server.
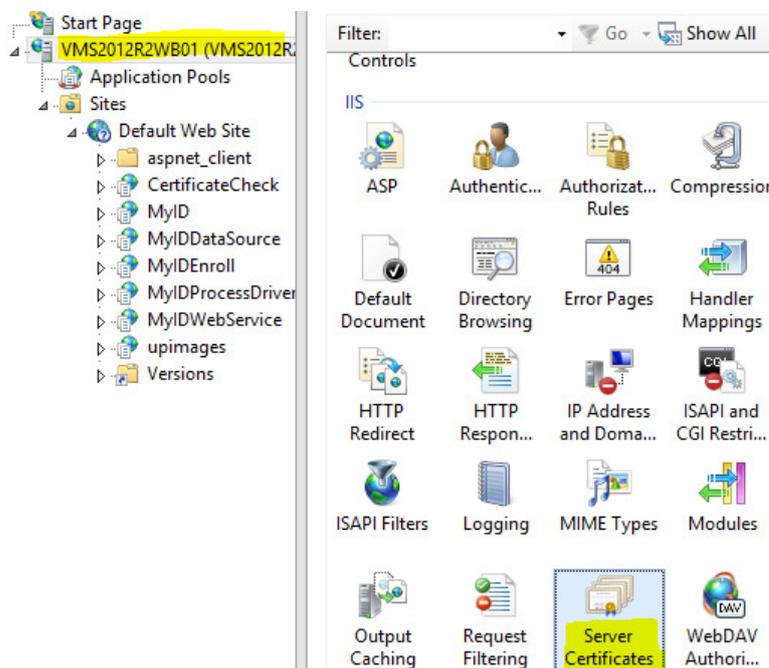
2. Bind the certificate to the website.

    a) In IIS manager, at the web site level, select **Bindings**.
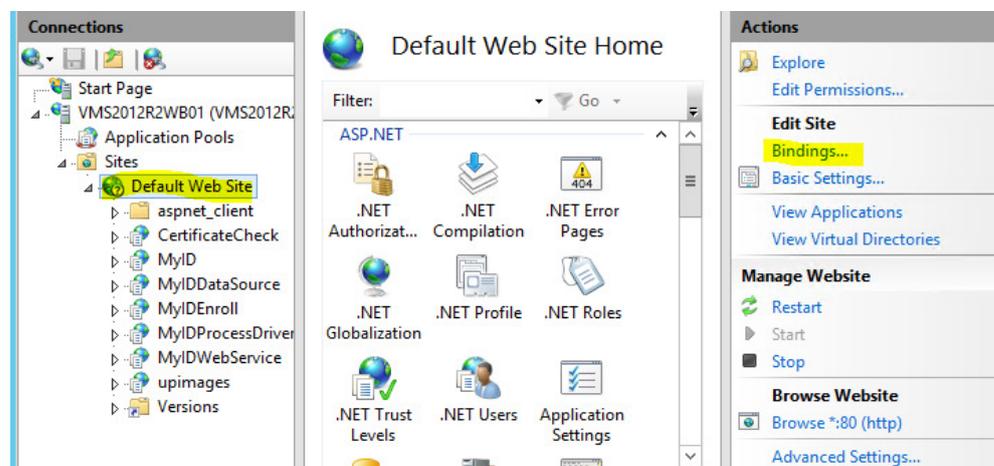
    b) This opens a dialog showing the current bindings. If https is not on the list, click **Add**.
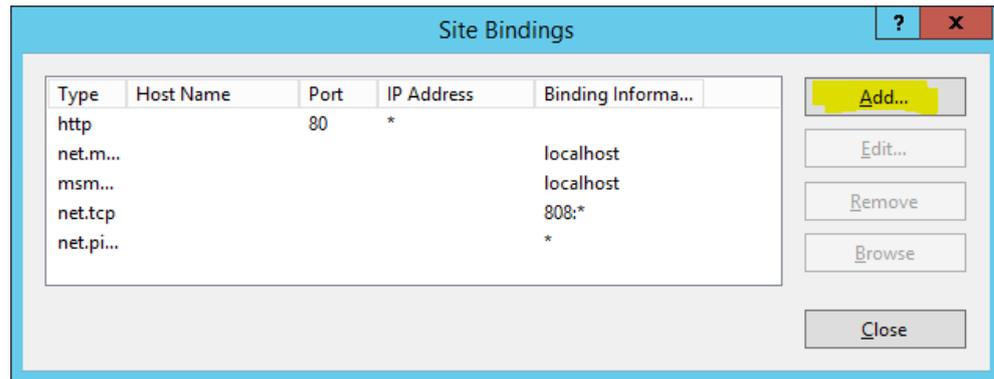
c) Select **https** as the **Type**, select the server SSL certificate, then click **OK**.
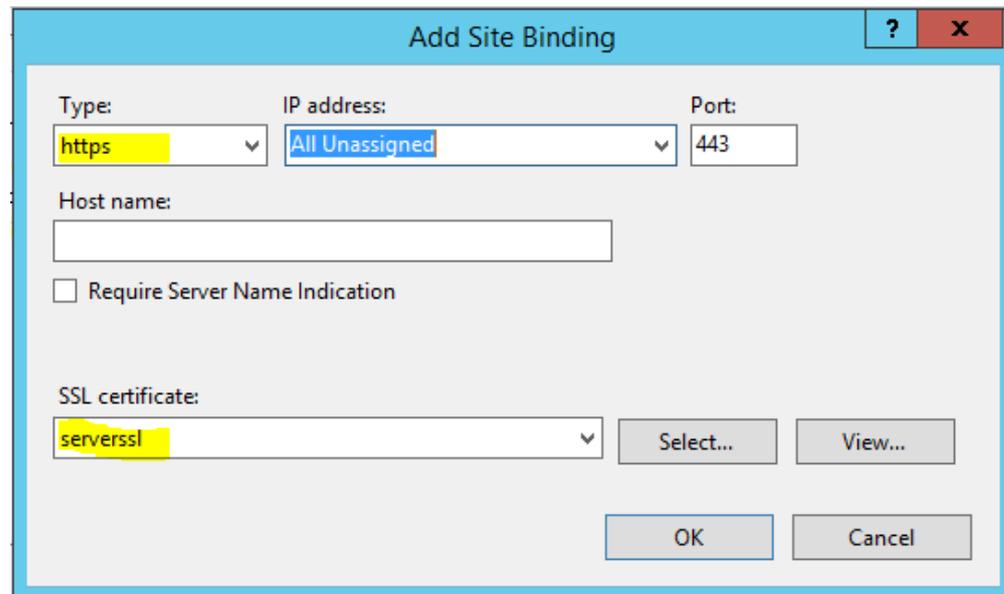
3. Configure the websites and web services to require SSL/TLS.

   Note: A PowerShell script is supplied to enable SSL/TLS for the web services. See the *Configuring MyID for 2-way SSL/TLS* section in the ***Installation and Configuration Guide***.

   At this point the web server has been SSL/TLS enabled – clients can choose to connect to HTTPS, although they may still connect to HTTP. In order to protect clients from accidentally connecting using HTTP (and therefore not getting the benefit of the privacy that https provides), we must set IIS to require SSL.

   This can be done individually for each virtual directory – but it is more efficient to require SSL for the entire website.

   Note: If the MyID web server is hosting other applications that do not support SSL/TLS, it may not be possible to enforce SSL/TLS for the entire web server, and you may need to set it for each virtual directory.

   a) In IIS manager, select the level to require SSL

      You are recommended to set this at the website level (so that it applies to the entire website).

   b) Select **SSL Settings**.
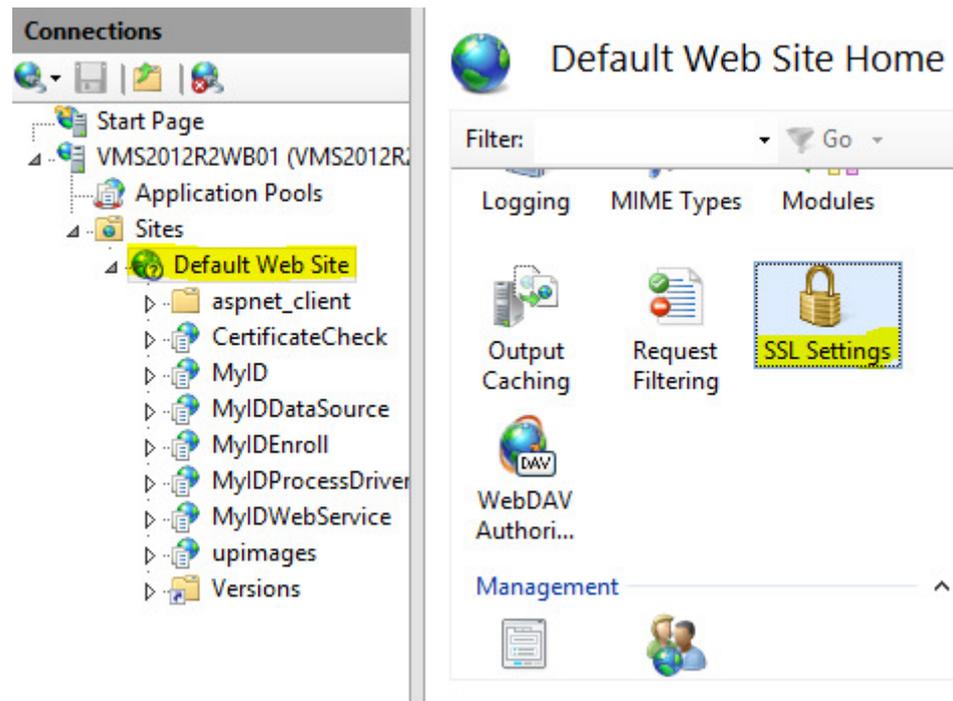
c) Select the **Require SSL** option, and click **Apply**.

    This enforces that clients may only connect using HTTPS – HTTP will be disabled.
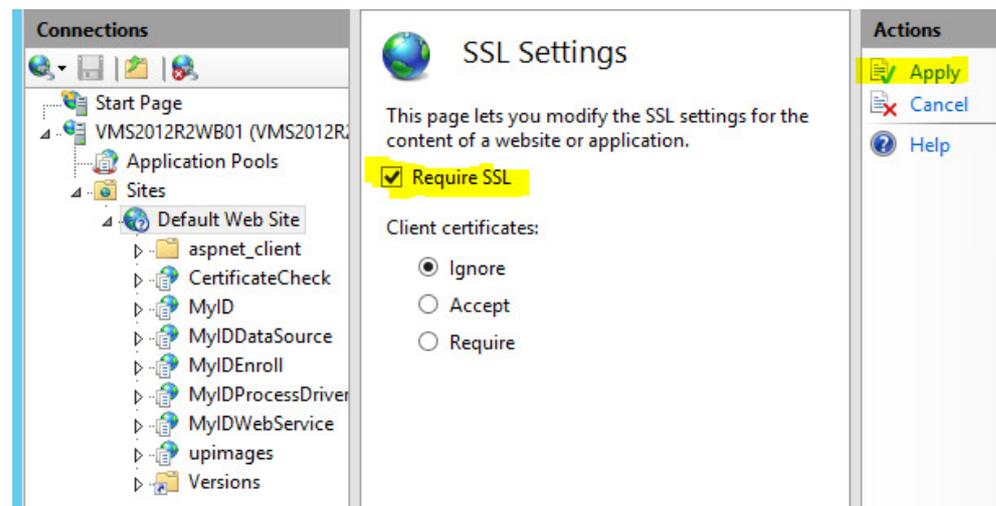
Since the web server certificate will expire, it is important to put a plan in place for renewing/replacing this certificate before it expires to ensure continuity of service.

Ensure that all clients use HTTPS (rather than HTTP) to connect to the MyID website and web services, since connecting through HTTP will no longer work.

For WCF web services, review the `Web.config` as described in section *5, Additional Configuration for WCF Web Services*.

3   Identifying IIS Authentication Options

Virtual directories/applications may be ASP, ASP.NET or WCF. It is important to identify the WCF virtual directories/applications as these require additional configuration as described in this document.

3.1   Client website and web services

Some virtual directories installed as part of MyID are intended to be consumed by end clients (for example, operators accessing MyID using MyID Desktop, the Self-Service App, the Self-Service Kiosk, or through mobile devices).

The client website and web services comprises:

Virtual Directory name	Type	Number of Instances
MyID	ASP website	1
upimages	ASP website	1
MyIDDataSource	ASP.NET web service *	Multiple (see below)
MyIDProcessDriver	ASP.NET web service *	Multiple (see below)

Multiple instances of `MyIDDataSource` and `MyIDProcessDriver` are created – instances at the root level, in version-numbered subfolders within the main folder.

8. `MyIDDataSource` and `MyIDProcessDriver` exist at the root of the Default Web Site.

9. Multiple copies of `MyIDDataSource` and `MyIDProcessDriver` exist in version-numbered subfolders.
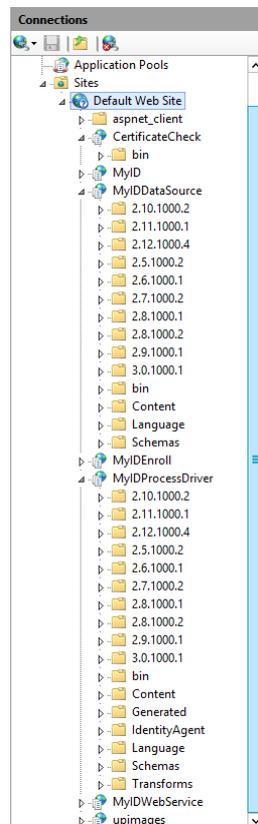
For `MyIDDataSource` and `MyIDProcessDriver`, since there are multiple instances, it is important that any authentication settings in IIS are applied to all instances of these web services, including the version-numbered instances.

The MyID application builds in authentication features (for example, smart card or security phrase logon) to protect access to these client websites and web services.

Therefore the options are:

10. Anonymous Access – IIS will let any client connect without authentication – the MyID application will take care of authentication. The client website and web services listed above default to anonymous access after installation.

11. Two-way SSL/TLS – the MyID application will take care of user authentication – but before this stage is reached, IIS will ensure that the client has authenticated with an appropriate client certificate.

   - IIS Client Certificate Mapping Authentication one-to-one mapping can be used for this, although with many clients this can become unmanageable.

   - IIS Client Certificate Mapping Authentication many-to-one mapping can be used – this can be configured to allow all certificates issued from a particular CA to connect.

   - The *Web Service Architecture Installation and Configuration* and *Installation and Configuration Guide* documents contain details on configuring MyID Desktop to use a particular client certificate to connect.

   - It is necessary to issue appropriate client certificates to the clients that are to be allowed to connect to MyID.

Note: If you select two-way SSL/TLS authentication for the `MyIDDataSource` web service, this is not supported for the following folder:

`MyIDDataSource/Content/img/workflow-icons`

When configuring two-way SSL/TLS for `MyIDDataSource`, browse to the `Content/img/workflow-icons` folder and configure that folder not to require two-way SSL/TLS.
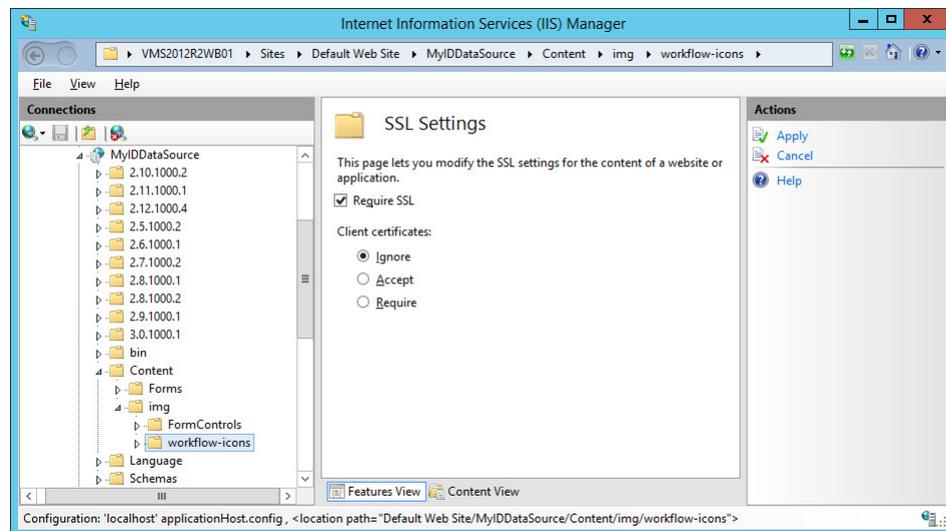
MyID also allows you to use Integrated Windows Logon, where the user's Windows credentials are used to authenticate the user to MyID – see the *Administration Guide* for more information.

3.2   Server-to-server web services

Some web services are intended to be consumed by other services. These web services do not have authentication built into them at the application level – they are shipped with anonymous access disabled, and you must configure the appropriate authentication mechanism in IIS.

These server to server web services include:

Virtual Directory name	Description	Type
MyIDEnroll	Lifecycle API (allow external systems to import and modify data).	ASP.NET web service
MyIDWebService	MI Reports (allow external systems to read data).	ASP.NET web service
DeviceManagementAPI	Device Management API (allow external systems to request and manage devices). This is not part of the default MyID installation – it is available as a separate patch.	WCF web service
CredentialWebService	Credential Web service (allow external systems to request and manage mobile/VSC credentials) This is not part of the default MyID installation – it is available as a separate patch.	WCF web service
DCNotificationListener	Listens for notifications to update derived credentials. This is not part of the default MyID installation – it is available as a separate patch.	WCF web service

You must review these web services and determine the appropriate authentication level. It is important to realize that IIS is providing the authentication – there is no additional application level authentication.

Appropriate options for each of these includes:

12. No access allowed – if the feature is not required, then the safest option is to disable all authentication mechanisms, therefore disabling access to the virtual directory.

13. IIS Client Certificate Mapping Authentication one-to-one mapping – this requires that a specified client certificate must be used to connect to the virtual directory. It is necessary to first enroll this certificate for the required clients. This is the recommended option for authentication for server to server web services

14. Windows authentication – this is appropriate only for cases where the client to connect is part of the same Windows domain hierarchy.

15. Anonymous authentication – this is only appropriate for development or test systems – production systems must *never* allow anonymous access to the server to server web services.

Note: You can configure IP Address and Domain Restrictions addition to the above authentication options.

4 Configuring Authentication in IIS

This section discusses configuring authentication. It is important to realize that IIS allows multiple authentication mechanisms to be enabled concurrently – offering the client a choice of authentication mechanism. When configuring a new mechanism it is therefore important to disable other mechanisms that were previously enabled.

4.1 Configuring for no access

To configure a virtual directory for no access:

1. In Internet Information Services (IIS) Manager, select the virtual directory in IIS to configure.

2. Select **Authentication**.

3. Ensure all mechanisms are set to **Disabled**.

4. If IIS Client Certificate Mapping Authentication has been previously enabled, disable it as described in sections *4.4.1*, *Configuring IIS Client Certificate Mapping Authentication one-to-one mapping* or *4.4.2*, *Configuring IIS Client Certificate Mapping Authentication many-to-one mapping*.

4.2 Configuring anonymous access

Note: The client web services and website are already configured for anonymous access when you install MyID.

> **Warning:** Server-to-server web services must never be configured for anonymous access for production systems – the configuration described here is for non-production testing purposes only.

To configure a virtual directory for anonymous access:

1. In Internet Information Services (IIS) Manager, select the virtual directory in IIS to configure.

2. Select **Authentication**.

3. Set **Anonymous Access** to **Enabled** and all other authentication methods to **Disabled**.

4. Select **Anonymous Authentication** and click **Edit**.
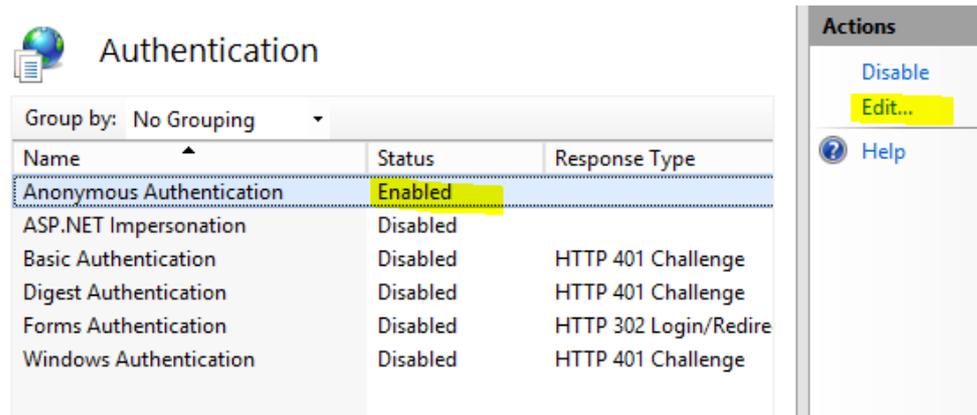
5. Select **Application pool identity**.

6. Click **OK**.

7. If IIS Client Certificate Mapping Authentication has been previously enabled, disable it as described in sections *4.4.1*, *Configuring IIS Client Certificate Mapping Authentication one-to-one mapping* or *4.4.2*, *Configuring IIS Client Certificate Mapping Authentication many-to-one mapping*.

4.3   Configuring Windows authentication

Note: Configuring Integrated Windows Logon for the client website and web services is described in the *Administration Guide*. The instructions in this section of this document are specific to server-to-server web services – *do not* follow these instructions for the client website and web services.

To configure a virtual directory for anonymous access:

1. In Internet Information Services (IIS) Manager, select the virtual directory in IIS to configure.

2. Select **Authentication**.

3. Set **Windows Authentication** to **Enabled**.

4. Ensure all other authentication methods are set to **Disabled**.
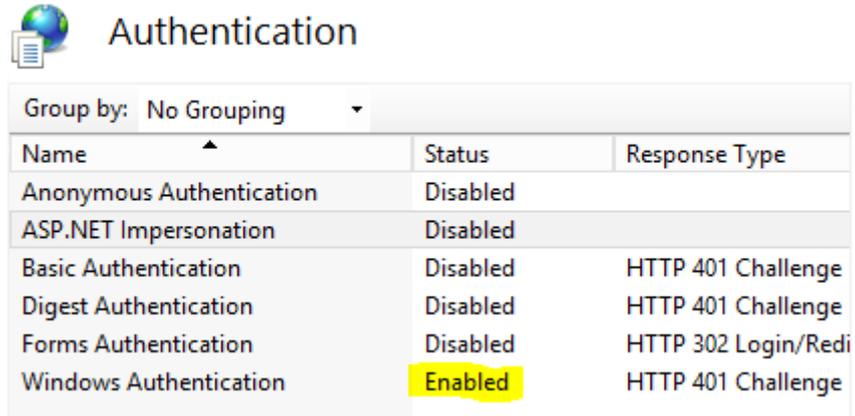
Authentication

Group by: No Grouping

Name	Status	Response Type
Anonymous Authentication	Disabled	
ASP.NET Impersonation	Disabled	
Basic Authentication	Disabled	HTTP 401 Challenge
Digest Authentication	Disabled	HTTP 401 Challenge
Forms Authentication	Disabled	HTTP 302 Login/Redi
Windows Authentication	Enabled	HTTP 401 Challenge

5. If IIS Client Certificate Mapping Authentication has been previously enabled, disable it as described in sections *4.4.1, Configuring IIS Client Certificate Mapping Authentication one-to-one mapping* or *4.4.2, Configuring IIS Client Certificate Mapping Authentication many-to-one mapping*.

6. For WCF web services, see section *5, Additional Configuration for WCF Web Services*.

7. The application calling the web service must connect to the web service using domain credentials that grant access to the components hosted by the web service.

4.4 Configuring IIS client certificates

IIS has a several different mechanisms available for specifying client SSL certificate requirements:

16. Client Certificate Mapping Authentication – this maps client certificates to associated domain accounts. This guide does not discuss how to configure Client Certificate Mapping Authentication, except to highlight that it is a feature that is distinct from IIS Client Certificate Mapping Authentication. For the purposes of this guide, Client Certificate Mapping Authentication is expected to be set to disabled (its default state).

17. IIS Client Certificate Mapping Authentication one-to-one mapping – configure a list of distinct client certificates that are allowed to authenticate, and map them to a Windows user account. A typical use case is to configure a client certificate that is allowed to connect to the MyIDEnroll web service, which maps to the MyID IIS user account.

18. IIS Client Certificate Mapping Authentication many-to-one mapping – configure rules to ensure that any client certificate that meets the rules will be mapped to the MyID IIS user account. A typical use case is for configuring clients that have been issued client certificates by a specific CA, where the subject belongs to a particular OU, the ability to access MyID.

IIS Client Certificate Mapping Authentication is an optional feature of IIS – you must ensure this feature is installed for it to be available. Verify this feature is enabled in Server Manager – in **Add Roles and Features**, on the **Server Roles** page, under **Web Server (IIS) > Web Server > Security**, select **IIS Client Certificate Mapping Authentication**.

Note: The user interface for configuring IIS Client Certificate Mapping Authentication has some quirks, which are important to understand in order to configure correctly.

19. The Configuration Editor option is used to configure IIS Client Certificate Mapping Authentication – in the IIS Manager it is possible to access the Configuration Editor at any level of the tree (server, website or virtual directory). However the IIS Client Certificate Mapping Authentication feature will work properly only when it is configured at the website level. This means that it is possible to configure one-to-one or many-to-one mappings for the Default web site to determine which client certificates may authenticate, and then configure individual virtual directories to require client certificates, but it is a limitation of IIS that it is not possible to configure different multiple virtual directories/application that require different client certificates to connect.

20. If you have a requirement for accepting different client certificates to connect to different virtual directories, you must:

   ◆ Create an additional web site in IIS – this will be assigned on a different port.

   ◆ Configure IIS Client Certificate Mapping Authentication for the new website – determining which client certificates may connect.

   ◆ Create a virtual directory/application in the new website to represent the feature, pointed at the same file location of the original version. Disable other authentication mechanisms (to force authentication to go through IIS Client Certificate Mapping Authentication).

   ◆ Disable or remove access to the original virtual directory/application that has been duplicated – you must not create a new locked down copy while still leaving open another route that bypasses the new IIS Client Certificate Mapping Authentication rules.

4.4.1 Configuring IIS Client Certificate Mapping Authentication one-to-one mapping

To configure one-to-one mapping for IIS Client Certificate Mapping Authentication:

1. In Internet Information Services (IIS) Manager, select the **Default Web Site**.

   Alternatively, select the web site you created as described in section *4.4*, *Configuring IIS client certificates*.

2. Select **Configuration Editor**.
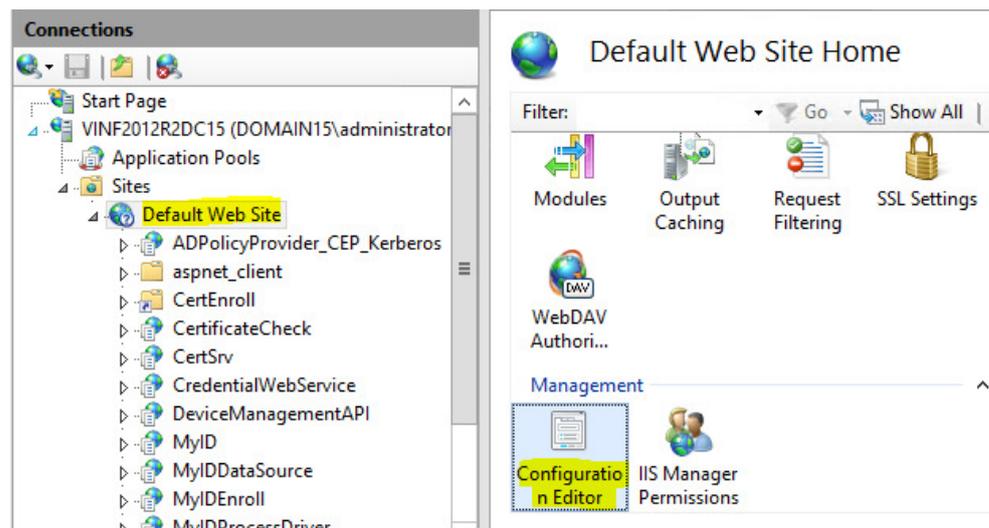
3. From the **Section** drop-down list, select the following:

   system.webserver > security > authentication > iisClientCertificateMappingAuthentication

4. From the **From** drop-down list, select **ApplicationHost.config**.

The following screen shows the (default) disabled state where this feature is disabled (False).
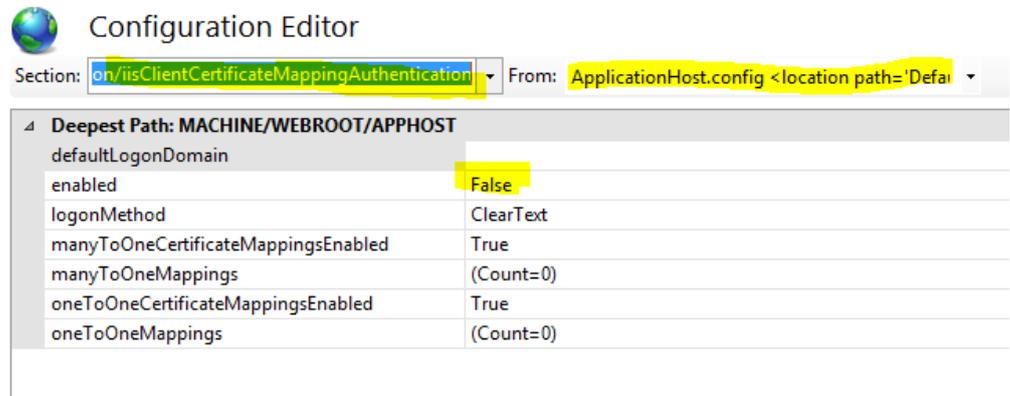
5. Set the following:

   ◆ **enabled** – **True**

   ◆ **manyToOneCertificateMappingsEnabled** – **False**

   ◆ **oneToOneCertificateMappingsEnabled** – **True**

6. Select the **oneToOneMappings** option and click the browse button.

7. In the Collection Editor dialog, click **Add**.

8. In the **certificate** box, enter the Base64 certificate.

   Note: This must be processed so that it fits on a single line, without any whitespace and without PEM headers.

   For more information, see:

   http://www.iis.net/learn/manage/configuring-security/configuring-one-to-one-client-certificate-mappings

9. Set **enabled** to **True**.

10. In the **userName** box, type the MyID IIS user account name, and type the account password in the **password** box.

    This is the Windows user account to which the certificate will be mapped.
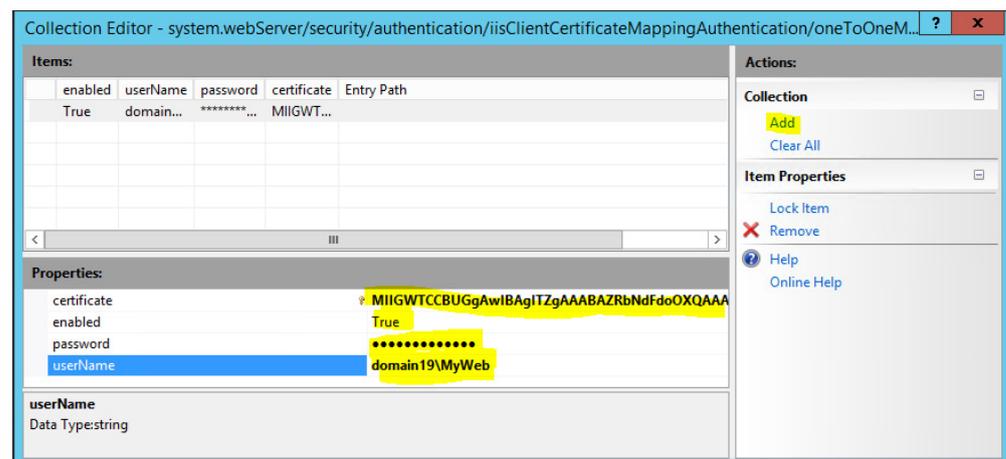
11. Close the dialog.

    The **oneToOneMappings** will now be updated to show how many certificates are mapped.

---
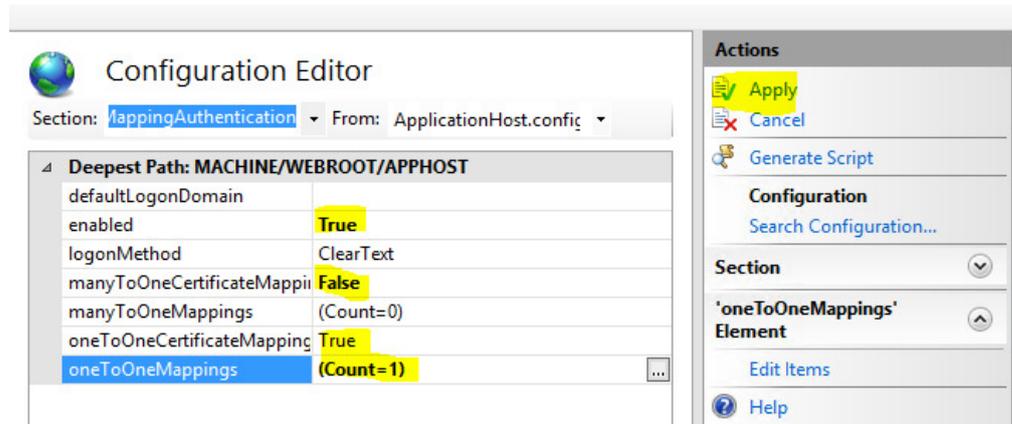
12. Click **Apply** to save the changes.

13. In the tree, select the virtual directory that is to use this rule for client certificate authentication, and select **SSL Settings**.
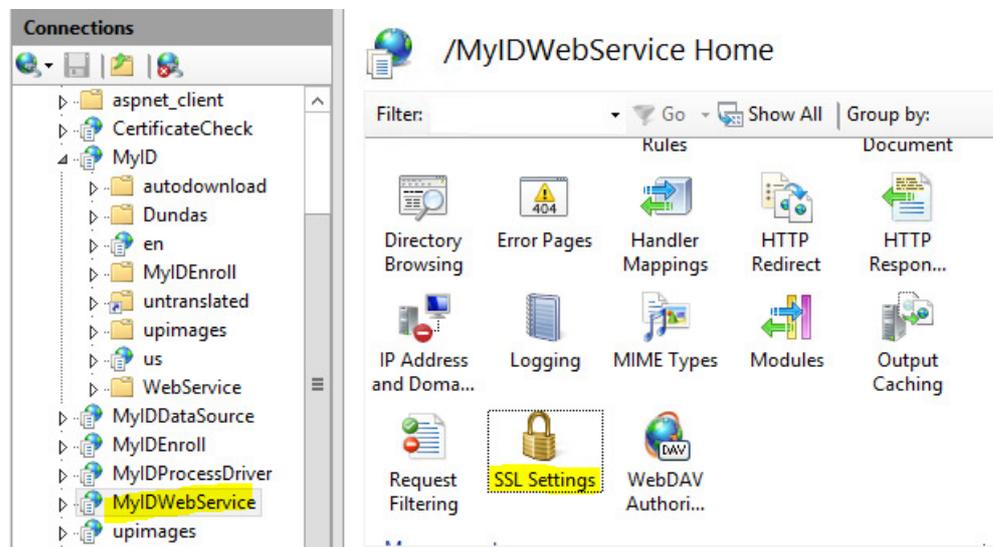
14. Click the **Require SSL** option, then in the **Client certificates** list select the **Require** option and click **Apply**.

15. In the tree, select the virtual directory, and select **Authentication**.

   ◆ For websites or ASP.NET web services, disable all authentication mechanisms.

   ◆ For WCF web services, you must have **Anonymous Access** enabled; without this option, IIS Client Certificate Mapping Authentication will not work.
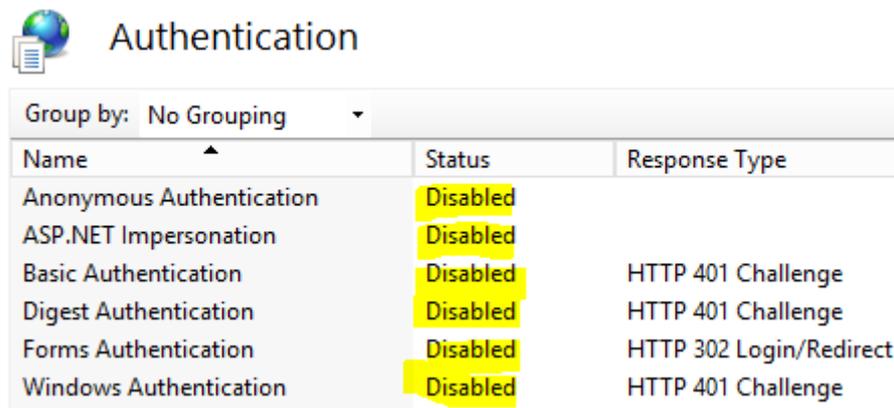
To confirm that the web service is configured correctly, Test that the web service is inaccessible except when the configured client certificate is used to authenticate.

If testing with a browser it is advisable to connect with a private browsing option, since this enforces that a new session is authenticated; otherwise an authentication made before IIS was reconfigured may still be granting access.

Since the client certificate(s) will expire, ensure there is a plan in place for renewing/replacing them before they expire, to ensure continuity of service.

For WCF web services, review the `web.config` settings for the web service; see section *5*, *Additional Configuration for WCF Web Services*.

4.4.2 Configuring IIS Client Certificate Mapping Authentication many-to-one mapping

This is identical to configuring one-to-one mapping, as described in section *4.4.1*, *Configuring IIS Client Certificate Mapping Authentication one-to-one mapping*, except for the following:

21. Set **oneToOneMappingEnabled** to **False**.

22. Set **manyToOneMappingsEnabled** to **True**.

23. Configure **manyToOneMappings** – typically a rule will be configured to map issuer DN to a Windows user account.

    Further information on configuring **manyToOneMappings** rules is available on the Microsoft web site in Knowledge Base article 2026113 *Configuring Many-to-One Client Certificate Mappings for Internet Information Services (IIS) 7.0 and 7.5*.

5 Additional Configuration for WCF Web Services

Some web services are WCF web services rather than ASP.NET web services. WCF web services can be identified as they contain an `.svc` file rather than an `.asmx` file.

These WCF web services need additional configuration within their `web.config` file to allow for SSL, and for authentication to be configured.

For WCF web services, both the IIS settings in section *4, Configuring Authentication in IIS* section and the `web.config` settings described in this section must be consistently applied for the type of authentication being configured. If the IIS settings and the `web.config` settings are inconsistent (for example, IIS is set up for a different authentication type than `web.config`), then depending on the exact configuration the web service may either not function at all, or may grant unintended access.

Note: Do *not* make these changes on web services which are not WCF web services.

There are two main types of binding used in WCF web services – `basicHttpBinding` and `webHttpBinding`. The example `web.config` excerpts below show `basicHttpBinding`. If your web service uses `webHttpBinding`, a `webHttpBinding` node will be present instead of a `basicHttpBinding` node.

Full documentation for configuring the WCF `web.config` files is supplied by Microsoft:

https://docs.microsoft.com/en-us/dotnet/framework/configure-apps/file-schema/wcf/

Warning	A patch that updates the WCF web service may overwrite the `web.config` file. Therefore after editing the `web.config` file, take a backup of it. After installing a MyID patch that modifies the WCF web service, you may need to restore the `web.config` file manually to re-enable the SSL/authentication settings.

5.1 Web.config settings to enable SSL

WCF patches shipped by Intercede from 2016 onwards already contain a `Web.config` that is configured to allow both non-SSL and SSL connections. These `Web.config` files contain two `add` nodes (one for the `http` scheme and one for the `https` scheme) under the `system.serviceModel/protocolMapping` section as follows:

```
<system.serviceModel>
  <protocolMapping>
    <add scheme="http" binding="basicHttpBinding"
bindingConfiguration="httpBinding" />
    <add scheme="https" binding="basicHttpBinding"
bindingConfiguration="httpsBinding" />
  </protocolMapping>
```

If your `Web.config` file contains an entry that matches the above pattern, no changes are required to `Web.config` to enable SSL. In this configuration, the web service will function with both SSL and non-SSL connections, and you can disable non-SSL connections by setting the **Require SSL** option in IIS.

If your `Web.config` file does not contain a `system.serviceModel/protocolMapping` for the `scheme="https"` then configure it as follows:

1. Edit the `Web.config` file for the WCF web service.

   Note: Web services that use `webHttpBinding` are shipped as HTTP enabled in the `web.config` file – if your `web.config` file does not include a `basicHttpBinding` node, skip this section.

2. Locate the `system.serviceModel/protocolMapping` section.

    a) To enable SSL connections ensure you have an `add` node under `protocolMapping` as follows:

```
<add scheme="https" binding="basicHttpBinding"
bindingConfiguration="httpsBinding" />
```

    b) To also enable non SSL connections, ensure you have another `add` node under `protocolMapping` as follows:

```
<add scheme="http" binding="basicHttpBinding"
bindingConfiguration="httpBinding" />
```

The value of the `bindingConfiguration` attribute must match the value of the `bindingConfiguration` attribute on the following node:

```
system.serviceModel/services/service/endpoint
```

(described below) and the value of the `name` attribute on the following node:

```
bindings/basicHttpsBindings/binding
```

3. Locate the `system.serviceModel/services/service/endpoint` section and set the `bindingConfiguration` attribute and `binding` attribute:

    a) To enable SSL (recommended), ensure that you have an `endpoint` node underneath the `service` node that has `binding` and `bindingConfiguration` attributes as follows:

```
<endpoint address="" binding="basicHttpBinding"
bindingConfiguration="httpsBinding" contract="serviceContract"
/>
```

    b) If you also want to enable non-SSL connections (not recommended), you can have another `endpoint` node underneath the `service` node as follows:

```
<endpoint address="" binding="basicHttpBinding"
bindingConfiguration="httpBinding" contract="do not modify" />
```

    c) In the above `endpoint` nodes:

        ▫ The value of the `bindingConfiguration` attribute must match the value of the `bindingConfiguration` attribute on the following node (described previously):

```
system.serviceModel/protocolMapping/add
```

        ▫ The value of the `contract` attribute will be a value that is already set in your `web.config` file that is specific to the web service you are configuring – do *not* change the value of the `contract` attribute – leave it at the value it is already set to in your `web.config` file.

4. Locate the `system.serviceModel/bindings/basicHttpBinding/binding` section.

    a) For each `bindings/basicHttpsBindings/binding` node, the value of the name attribute must match the value of the `bindingConfiguration` attribute on the following nodes:

```
services/service/endpoint
```

```
system.serviceModel/protocolMapping/add
```

    b) Depending on your configuration you may have one or two `binding` nodes underneath `bindings/basicHttpBinding`.

c) To enable SSL (recommended) ensure you have a binding node underneath `bindings/basicHttpBindings` as follows:

```
<bindings>
    <basicHttpBinding>
        <binding name="httpsBinding">
            <security mode="Transport">
```

Note that there will be a `transport` node underneath the `security` node that has a `clientCredentialType` attribute that defines the authentication type. The `transport` node must be present, but subsequent sections of this guide describe how to set the `clientCredentialType` attribute depending on the desired authentication mechanism.

If you also want to enable access without SSL (not recommended), ensure you have another `binding` node as follows:

```
<binding name="httpsBinding">
    <security mode="TransportCredentialOnly">
```

This must also have a `transport` subnode as described above.

Note: The above settings must be consistent with the **Require SSL** setting in IIS manager; that is, if IIS manager is configured for the WCF service to use SSL, then the `web.config` file for that WCF service must also be configured for SSL bindings and behaviors, otherwise the web service will not function.

5.2 Web.config settings to enable Windows authentication

To configure the service for Windows authentication:

1. Edit the `Web.config` file for the WCF web service.

2. Locate the binding security section.

   This is either:

   `system.serviceModel/bindings/basicHttpBinding/binding/security`

   or:

   `system.serviceModel/bindings/webHttpBinding/binding/security`

3. Under the `system.serviceModel/bindings/security/transport` nodes, change the value of the `clientCredentialType` attribute to `Windows`.

```
<system.serviceModel>
  <bindings>
    <basicHttpBinding>
      <binding name="httpsBinding">
        <security mode="Transport">
          <transport clientCredentialType="Windows" />
        </security>
```

There may be multiple `security/transport` nodes (a node for `httpBinding` and a node for `httpsBinding`) – if so, modify the `clientCredentialType` attribute on all of them.

Note: In the above example, the `name` and `mode` attributes on your system may be different values to what is shown above:

 The name attribute must match the `bindingConfiguration` attribute on the `system.serviceModel/protocolMapping/add` node.

 When mapping to an `httpsBinding`, the `mode` attribute must contain the value `Transport`. When mapping to an `httpBinding`, the `mode` attribute must contain the value `TransportCredentialOnly`.

Note: The above settings must be consistent with the **Require SSL** setting in IIS manager; that is, if IIS manager is configured for the WCF service to use Windows authentication, the `Web.config` file must also be configured for Windows authentication, as described above.

5.3 Web.config settings to enable anonymous authentication

Warning:	Anonymous authentication is to be used for test systems only. Do not use anonymous authentication on production systems.

To configure the service for anonymous authentication:

1. Edit the `Web.config` file for the WCF web service.

2. Locate the binding security section.

   This is either:

   ```
   system.serviceModel/bindings/basicHttpBinding/binding/security
   ```

   or:

   ```
   system.serviceModel/bindings/webHttpBinding/binding/security
   ```

3. Under the `system.serviceModel/bindings/security/transport` nodes, change the value of the `clientCredentialType` attribute to `None`.

   ```
   <system.serviceModel>
     <bindings>
       <basicHttpBinding>
         <binding name=" httpsBinding">
           <security mode="Transport">
             <transport clientCredentialType="None" />
           </security>
   ```

   There may be multiple `security/transport` nodes (a node for `httpBinding` and a node for `httpsBinding`) – if so, modify the `clientCredentialType` attribute on all of them.

   Note: In the above example, the `name` and `mode` attributes on your system may be different values to what is shown above:

   ◆ The name attribute must match the `bindingConfiguration` attribute on the `system.serviceModel/protocolMapping/add` node

   ◆ When mapping to an `httpsBinding`, the `mode` attribute must contain the value `Transport`. When mapping to an `httpBinding`, the `mode` attribute must contain the value `TransportCredentialOnly`.

   Note: The above settings must be consistent with the authentication setting in IIS manager; that is, if IIS manager is configured for the WCF service to use anonymous authentication, the `Web.config` file must also be configured for anonymous authentication, as described above.

5.4 Web.config settings to enable two-way SSL

IIS Client Certificate Mapping Authentication does not operate correctly with WCF web services – its standard behavior is to allow any certificate trusted by the server to connect.

To secure server-to-server WCF web services, you can use the Service Authorization Manager, a separate module available from Intercede customer support. This module allows you to configure a WCF application/virtual directory to require two-way SSL/TLS, requiring authentication with a whitelisted client SSL/TLS certificate.

For more information, contact customer support, quoting reference SUP-215.

5.5   Preventing the publishing of WSDL

You can control whether the WSDL for the web service is made available to the caller by modifying the `system.ServiceModel/behaviours/serviceMetadata` node.

Locate the `system.ServiceModel/behaviours/serviceMetadata` section.

24. The `httpsGetEnabled` attribute controls whether the WSDL is made available over https.

25. The `httpGetEnabled` attribute controls whether the WSDL is made available over http.

You can set these to `false` to disable WSDL, or `true` to enable WSDL.