MyID Professional

Version 11.4

Web Service Architecture

Installation and Configuration Guide

Lutterworth Hall, St Mary's Road, Lutterworth, Leicestershire, LE17 4PS, UK
www.intercede.com | info@intercede.com | @intercedemyid | +44 (0)1455 558111

Document reference: IMP1986-02 December 2019

Copyright

Conventions Used in this Document

- 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

- **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

- *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"

- ***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 installation media.

- 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.

- **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.

- 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

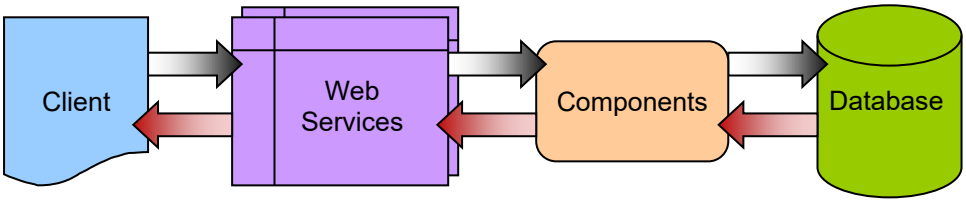
This document describes the MyID® Web Service Architecture. The web services installed on your web server allow MyID end-user applications to communicate with your MyID system. For example, the web services allow you to use the following client applications:

- MyID Desktop
- MyID Self-Service App

The following web services are provided:

- MyID Process Driver Web Service – allows a client application to communicate with the MyID application server to carry out card and identity management.
- MyID Data Source Web Service – provides form definitions and device configuration data to a client application. This is a read-only web service with no security restrictions.
- Certificate Check Web Service – allows a client application to check the status of its certificates using the Windows API. This is an optional web service.

1.1 Overview



The client passes requests through HTTP or HTTPS to the MyID Data Source and MyID Process Driver web services; both services are required for full operation. The web services communicate using DCOM with the MyID components on the application server; these components provide the business logic and communicate with the MyID database. Responses are returned to the client through the MyID web services.

The web services, components and database may be on separate servers, or on the same server. The two web services must be installed on the same server.

1.2 Prerequisites

The MyID Web Service Architecture is provided either as a stand-alone update or as part of the main MyID product installation program.

For MyID versions supported and any patch prerequisites, see the ***Installation and Configuration Guide*** provided on the MyID product media or the ***readme.html*** document provided with the software update.

In addition:

- You need to have .NET 4.8 installed on the server on which the web services are installed.
- Your client applications must be able to communicate over HTTPS to the web server on which you have installed the MyID web services.

  You *must* set up SSL/TLS on this connection.
- The MyID web services must be able to communicate with the MyID components using DCOM. If the web services reside on a separate server to the MyID application server, you must set up the appropriate COM+ proxies.

  See section *2.2.3*, *Setting up the COM+ proxies* for detail.

1.3 Change history

Version	Description
INT1986-01	First release for MyID Professional.
INT1986-02	Released with MyID 11.4.

2 Installing the Web Services

The web services are provided as part of the main MyID product installation program.

To install the web services as part of the MyID installation, on the Select MyID Server Roles and Features page, select the **Web Services Server\MyID Client Web Service** option.

Note: If you are installing the web services on a separate server to the main MyID web server, you must configure the web services with the location of the MyID web server. See section *2.2*, *Setting up the MyID web services on a standalone server*.

See the ***Installation and Configuration Guide*** for details of the installation procedure.

2.1 Web service configuration

To ensure maximum compatibility with MyID clients, multiple versions of the MyID web services are installed – each client automatically uses the most appropriate version of the web services.

Accordingly, to ensure that the web services are configured correctly, you may need to edit the `myid.config` file in multiple locations.

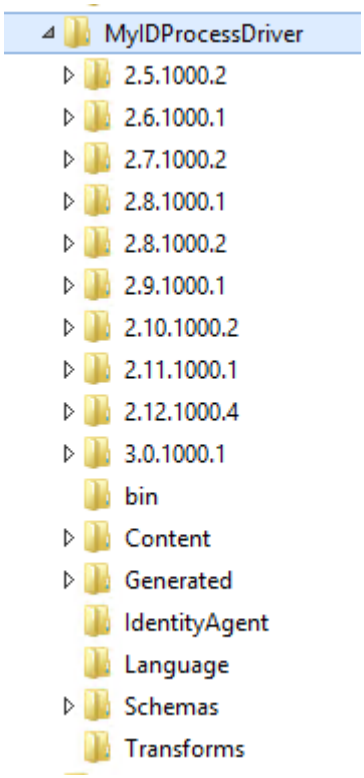
By default, the MyID web services are installed to the following folder:

```
C:\Program Files (x86)\Intercede\MyID\SSP\
```

In the root of this folder are folders for each of the individual web services:

- `MyIDDataSource` – contains a `myid.config` file.
- `MyIDProcessDriver` – contains a `myid.config` file.

Within MyIDDataSource and MyIDProcessDriver, there are subfolders that contain the version specific files; for example:



If you make a change to the `myid.config` file in the root of the web service folder, the change will be inherited by all the versions unless that version has the same setting explicitly set.

2.2 Setting up the MyID web services on a standalone server

You may want to set up your MyID web services on a different server to the MyID application or web servers; in this case, you must carry out some additional configuration.

2.2.1 Configuring the server

For a standalone web services server, follow the instructions in the MyID ***Installation and Configuration Guide*** for preparing a system for a web server.

Note, however, that a standalone web services server does not need all of the role services that a web server needs. You must have the following role services:

- Static Content
- Default Document
- ASP.NET
- .NET Extensibility
- ISAPI Extensions
- ISAPI Filters
- Request Filtering
- IIS Management Console

You are also recommended to have the following:

- HTTP Logging

2.2.2 Installing .NET framework

You must install .NET Framework 4.8 on the server.

2.2.3 Setting up the COM+ proxies

If the web services are on a different server to the MyID application server components, you must export the MyID COM+ proxies to the server on which the web services run. This allows the web services to communicate with the MyID COM+ components on the application server.

To do this, you need the `.msi` files in the `Components\Export` folder on the MyID application server. By default, this is:

```
C:\Program Files (x86)\Intercede\MyID\Components\Export
```

You need to install the following proxies:

- APDUCardServer
- Edefice_BOL
- Edefice_CS
- ExpiringItems

Different web services require different proxies; see the table below for details.

To run the COM+ proxy installers, either:

- From the MyID web server, browse to a share on the MyID application server and run the `.msi` installers directly. For example, browse to:

  ```
  \\<server>\C$\Program Files (x86)\Intercede\MyID\Components\Export
  ```

  where `<server>` is the name of your MyID application server and `C$` is a share of the root of the `C:` drive. Run the `.msi` files directly.

  Note: If you experience any problems, make sure you have added the application server to the list of Trusted Sites on the web server.

  or:

- Copy the `.msi` files to the MyID web server and run the installers from there.

Note: If you are using multiple servers for your web services in conjunction with a load balancer, you must ensure that you set up session affinity on your servers.

2.2.4 Installing the MyID web service components

You must install the web services on the server using the supplied installation program. This installer creates the virtual directories and the application pool for the web services.

2.2.5 Setting the location of the web server

If the web services server is not the same server as the web server, you must edit the `myid.config` file in the `MyIDProcessDriver` folder. Add the following line:

```
<add key="WebServer" value="https://myserver"/>
```

Where `myserver` is the domain name of your MyID server. You do not need to include the MyID virtual directory.

Note: The case of `WebServer` is important.

2.2.6 Troubleshooting

If you have an existing server which has .NET 4.8 and IIS already installed and the site is not working as expected, try running the following statement at the Windows command line:

```
C:\Windows\Microsoft.NET\Framework\v4.0.30319\ aspnet_regiis.exe -i
```

This command ensures that .NET 4 is registered with IIS.

2.3 Configuring the MyID web services for Integrated Windows Logon

If you set up the MyID server to use Integrated Windows Logon, some applications using the web services can use the cardholder's currently logged-on Windows identity to authenticate to MyID without having to enter passphrases or use a smart card.

See the ***Administration Guide*** for details of setting up Integrated Windows Logon.

In addition to the procedures in the MyID documentation, you must also set up the authentication in IIS.

A PowerShell script called `ConfigureWindowsAuthentication.ps1` has been provided; this is installed on the MyID web server in the `Utilities` folder.

The script takes the following optional parameters:

- `webSiteName` – This is the name of the web site that is hosting the MyID web service. By default, this is:

  `Default Web Site`

- `installationPath` – This is the folder where MyID was installed. By default, this is:

  `C:\Program Files (x86)\Intercede\MyID`

The script ensures that Anonymous Authentication is set for MyIDDataSource and MyIDProcessDriver, and that Windows Authentication is enabled for the `WindowsAuth.asmx` web service.

Note: If you upgrade your MyID web services, you may have to run this PowerShell script again.

2.4 Configuring the MyID web services for 2-way SSL/TLS

See the *Configuring MyID for 2-way SSL/TLS* section in the ***Installation and Configuration Guide***.

2.5 Security for self-service operations

MyID has implemented a series of security features where, amongst other security considerations, it is no longer possible to determine a username from just a serial number. This limitation prevents some self-service operations.

The issue may present with an error similar to:

```
This logon mechanism isn't available with the current configuration.
501107
```

To configure MyID to allow the previous behavior, edit the `myid.config` file in the `MyIDProcessDriver` folder. Set the value of the key `PreventStartWorkflowWithPassphraseByDevice` to `false` to disable this feature.

2.6 Checking the status of the web services

You can use the `IsAlive` API method on the web service to confirm that the web services are running and reachable. For example, you may want to check that the web services are running before launching the Self-Service App.

To check the status, call the following method:

```
https://<server>/MyIDProcessDriver/ProcessDriver.asmx/IsAlive
```

where:

- `<server>` is the server name of your web services server.

This method returns the Boolean value `true` if the web services are running; for example:

```
<boolean xmlns="https://www.intercede.com/myid">true</boolean>
```

2.7 Specifying the target user

The user identifier passed to the MyID server is based on the Windows logon name of the user. This is then matched against the SAM Account Name stored for the user in the MyID database.

You can change how the system handles the user identifier in the following ways:

- Set the `/un` option on the command line of the Self-Service App to the logon name you want to use.
- To change the identifier that is passed to the web services, set the Windows environment variable `MYID_USERNAME` to the identifier you want to use. This value is used instead of the Windows logon name for all users on the PC.

  Note: This environment variable has no effect if you launch the Self-Service App using a hyperlink. To specify a different logon name, you must use the `/un` command line option instead.
- To change which MyID field the identifier is matched against, alter the `ws_LogonJobs` view in the MyID database to change the definition of the `UserIdentifier` field to point to a different field. This allows you to compare the user identifier to a field other than the SAM Account Name for the user.

  Note: Any installation of a MyID update may affect the `ws_LogonJobs` view in the MyID database; after you update MyID, you must check the `ws_LogonJobs` view in the database and, if necessary, re-apply any customizations.

Note: In addition to the Windows logon name, MyID also passes the User Principal Name from the client and attempts to match this against the UPN stored for the user in the MyID database; however, if you use the `/un` command line option or the `MYID_USERNAME` environment variable to override the Windows logon name taken from the client, MyID does *not* pass the User Principal Name from the client.

2.7.1 Case sensitivity

- When MyID matches the User Principal Name from the client against the UPN stored in the database, it carries out a case-sensitive match.
- When MyID matches the Windows logon name against the SAM Account Name stored in the database, it carries out a case-sensitive match.
- When MyID matches the username provided by the `/un` command line option or the `MYID_USERNAME` environment variable against the SAM Account Name stored in the database, it carries out a case-*in*sensitive match.