STSOrgSync

Installation Guide

**Version**: 1.0.0

**Date**: 25.11.2015

**Author**: BSG

Content

[1 Introduction 3](#_Toc467007587)

[2 Pre-requisites 3](#_Toc467007588)

[2.1 The FOCES certificate 3](#_Toc467007589)

[2.2 The service-agreements 3](#_Toc467007590)

[3 Installation 4](#_Toc467007591)

[3.1 Installing the client certificate 4](#_Toc467007592)

[3.2 Configuration 6](#_Toc467007593)

[3.3 Database setup 7](#_Toc467007594)

[3.4 Installing the Windows Service 7](#_Toc467007595)

[3.5 Setting up SSL 8](#_Toc467007596)

[4 Certificate renewal 9](#_Toc467007597)

[4.1 Renewal of KOMBITs certificates 9](#_Toc467007598)

[5 Logging and monitoring 10](#_Toc467007599)

[5.1 Logging 10](#_Toc467007600)

[5.1.1 Windows Registry setting 10](#_Toc467007601)

[5.1.2 Log.config file 10](#_Toc467007602)

[5.2 Monitoring 10](#_Toc467007603)

# Introduction

STSOrgSync is distributed both as an API (in various flavors) as well as an actual pre-compiled Windows Service, which can be deployed on a Windows Server 2012 R2 server.

This document covers all variants of using STSOrgSync, as the requirements and installation procedure is almost the same.

When using the SDK API, the Windows Service does not need to be installed (as the software being build using the SDK takes the place of the Windows Service), but everything else is the same.

# Pre-requisites

Before STSOrgSync can be taken into use, a FOCES certificate must be procured, and all relevant service-agreements must be in place.

## The FOCES certificate

KOMBIT has based their security infrastructure on the Danish national OCES certificates. For system-to-system integration, a certificate of FOCES (also called Funktions-OCES) variant is required.

FOCES certificates is required in a wide range of use-cases when integrating with public services in Denmark. This document does not cover the procedure for procuring a FOCES certificate, as this can be done by the municipalities certificate-administrator at the NemID self-service portal.

<https://www.medarbejdersignatur.dk/produkter/nemid_medarbejdersignatur/nemid_selvbetjening>

## The service-agreements

Once KOMBIT delivers Administrationmodulet, management of service agreements can be performed online in a self-service portal. Until then, service agreements are managed through word-forms, that are filled out and e-mailed to KMD.

KOMBIT has supplied a form for this purpose, which can be downloaded here

<https://share-komm.kombit.dk/P024/Delte%20dokumenter/Serviceaftale%20anmodning.docx>

A version of this form, filled out with all the required details (except the name and CVR number of the municipality) is distributed with STSOrgSync.

The form must be e-mailed to [sts-support@kmd.dk](mailto:sts-support@kmd.dk) together with the public part of the FOCES certificate (a file ending with .cer or .crt).

The service agreement will ensure that security tokens can be retrieved from the Token Service, which is part of the security infrastructure of Støttesystemerne.

Finally, a similar agreement must be created on serviceplatformen.dk, allowing access to the proxy services on Serviceplatformen.

KOMBIT has created a step-by-step guide called “Kogebogen” for this purpose. It also covers the word-form mentioned above.

<https://share-komm.kombit.dk/P013/Delte%20dokumenter/Kogebog%20V_5.pdf>

# Installation

It is possible to compile STSOrgSync using Visual Studio and then manually install the software, or just install the latest pre-compiled version by running STSOrgSyncSetup.exe which is distributed with the product.

When running STSOrgSyncSetup.exe, the following steps are performed

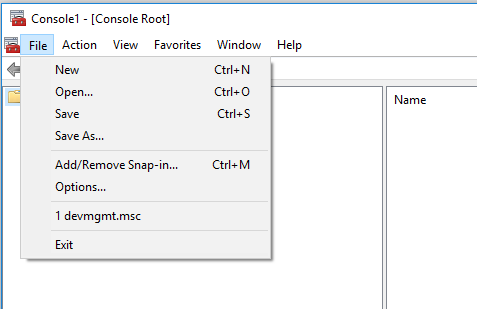
* The software is installed in [Program Files]\Digital Identity\STSOrgSync
* All certificates that must be trusted are installed in the correct keystores on the Windows machine
* A Windows Service called STSOrgSync is installed (but not started) on the machine
* The default configuration is loaded into Windows Registry

After the above installation, customization of the installation is required, as described below.

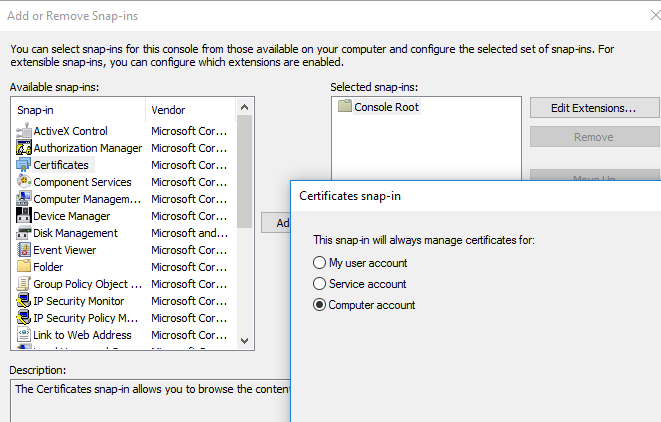
## Installing the client certificate

The previously mentioned FOCES certificate must be installed on the machine where the STSOrgSync software is to run. The certificate must be installed under “Local Computer” -> “Person”. There are many ways to do this, one is using mmc, which is described below

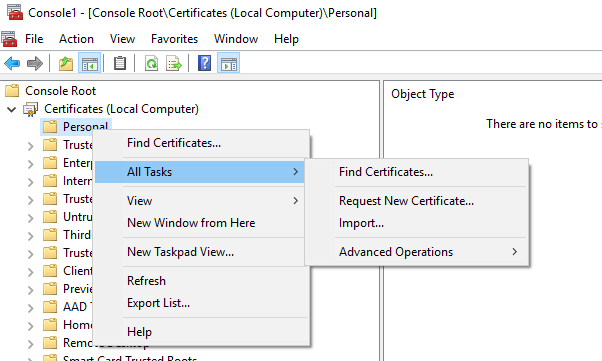
1. Start mmc.exe (click Windows->Run and enter mmc.exe)
2. Click on “File” and pick “Add/remove snap-in”



1. Select the “Certificate” snap-in and pick “Computer Account” followed by “Local Computer”



1. Go to “Personal”, right-click and pick “Import…”



1. Change the file-type to “all files (\*.\*)” and browse to the file containing the FOCES certificate, enter the password for the keystore and follow the steps in the wizard. Note that the file extension for the FOCES certificate is either .pfx or .p12 (and not .cer)
2. Once installed, it is possible to right-click on the certificate and select “manage private keys”. The system account that is going to run STSOrgSync must be given full access to the private keys.
3. Finally, we will need the “thumbprint” of the FOCES certificate just installed. So double-click on the certificate and go to the “Details” pane. Here scroll to Thumbprint value, and copy it… it will look something like this (we will need it in the configuration section)

‎09 19 ed 32 cf 87 58 a0 02 b3 9c 10 35 2b e7 dc cc f1 22 2a

If the certificate is not correctly installed, security errors relating to certificates will appear in the log when running STSOrgSync.

## Configuration

STSOrgSync uses the Windows Registry to store all configuration. The key where configuration is stored is

HKEY\_LOCAL\_MACHINE\SOFTWARE\Wow6432Node\Digital Identity\STSOrgSync

The STSOrgSync distribution comes with a config.reg file, which contains a set of default keys that can be used to bootstrap the configuration. Simply double-click on the file, and all the keys will be created – note that if the STSOrgSyncSetup.exe installer was used, these values are automatically loaded into registry.

Below is a list of all keys, and what each is used for.

Note that changes to registry does not take effect until STSOrgSync is restarted

|  |  |
| --- | --- |
| Key | Description |
| ClientCertThumbprint | This is the thumbprint value found when installing the FOCES certificate. It is used by STSOrgSync to locate the correct certificate when calling Organisation.  **Example value**  ‎09 19 ed 32 cf 87 58 a0 02 b3 9c 10 35 2b e7 dc cc f1 22 2a |
| DBConnectionString | When using the ServiceLayer and SchedulingLayer, this must be filled out, and be a valid database connection string to a MS SQL server. The user account that runs STSOrgSync must be allowed to create tables.  Alternatively, the value SQLITE can be used in this configuration field, in which case a local light-weight SQL server is used.  **Example value**  Data Source=52.209.14.128;database=STSOrgSync |
| DisableRevocationCheck | This true/false flag is used to control whether STSOrgSync performs revocation check on certificates.  When running in the test-environment it is recommended to set this flag to true, as the service to check revocation against is not easily accessible in the test-environment of DanID/Nets. |
| LogRequestResponse | This true/false flag is used to tell STSOrgSync if it should log the full request/response payloads when calling Organisation. It is recommended to set this value to false in production, as the payloads are quite big. |
| Municipality | This is the CVR number of the municipality. It is used to indicate the correct service-agreement to use. |
| OrganisationUUID | This is the UUID of the Organisation object that KOMBIT has created for the municipality. This value must be retrieved from KOMBIT, and should never change. |
| Environment | This flag is used to set which environment of Organisation to call against. The allowed values are   * **TEST**. This value will configure STSOrgSync to call against the test environment on the Serviceplatform. * **PROD**. This value will configure STSOrgSync to call against the production environment on the Serviceplatform. |
| UseSSL | This is a true/false flag that is used to indicate to the service, whether it is protected by SSL or not.  SSL configuration is not performed inside the service (see chapter 3.5 for details), but the service needs to know if it should except an SSL connection or not. |

## Database setup

If the Service API or SQL API are used, then a database is needed to run STSOrgSync. The database must be a Microsoft SQL Server, and the user account running STSOrgSync must be allowed to create tables on the database.

STSOrgSync uses very little space, as the SQL Server is used only as a queue for outgoing data, and data is removed once it has been synchronized with Organisation.

Note that it is possible to configure STSOrgSync to use a local SQL server by setting the connection string to the value SQLITE in the configuration above. If this setting is used, it is not possible to use the SQL API, but the Service API will function fine. The primary drawback of using the local SQL server, is that a harddrive crash will cause whatever registrations might be waiting in the queue to be lost.

## Installing the Windows Service

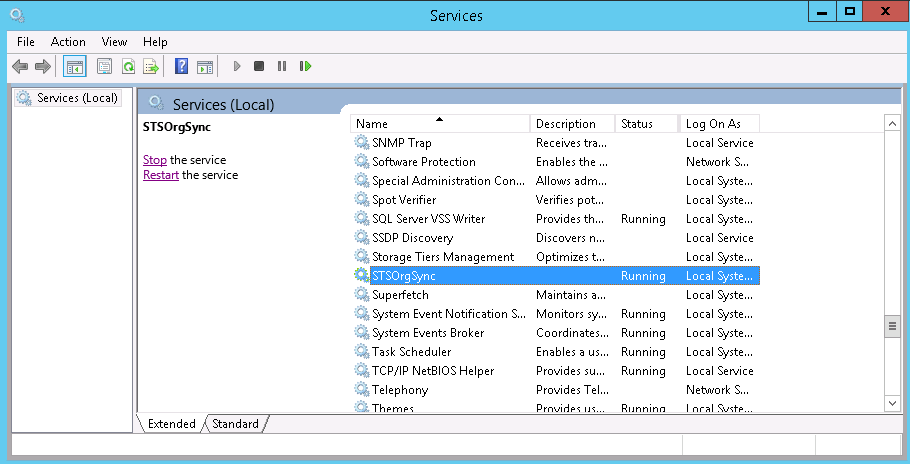
STSOrgSync contains a Windows Service project, which is also distributed as pre-compiled binary ready to installed.

To install the service, simply execute the following command from Powershell

WindowsService.exe --install

If the software is installed using the STSOrgSyncSetup.exe installer, the service is automatically installed.

Once installed, the service will be visible in the Services overview in Windows, as shown below



Note that the service is not configured to start automatically – if this is required, just right-click on the service and go to properties.

The service defaults to running as Local System, and if the service should run as a dedicated user, this should be changed using the Services GUI above.

It is recommended (but not required) that this system user has local administrative rights on the server where the service is running. If not, the user must manually be given full access rights to the following

* The certificates used (mmc.exe can be used to manage access rights to private keys)
* The folder where STSOrgSync is installed (write access)
* The port (9010) which the service binds to (netsh can be used to manage access rights to specific ports)

## Setting up SSL

Configuring the Windows Service for using SSL on the service endpoints is optional, though recommended.

The implementation of STSOrgSync uses the Open Web Interface for .NET (OWIN) for accepting connections, which in turn uses the network configuration on the server for handling things like SSL.

SSL for a given service is configured using Powershell, and requires that a valid SSL certificate is installed on the server. Getting an SSL certificate is outside the scope of this document, but both self-signed and purchased certificates can be used.

The following steps must be performed

1. If the service is running, stop it while configuring SSL
2. Install the SSL certificate using mmc (the certificate should go to Local Machine / Personal, and must contain the private key)
3. Remember to set the registry key UseSSL to true (see chapter 3.2)
4. Execute the following powershell command as Administrator

netsh http add sslcert ipport=0.0.0.0:9010 certhash=[thumbprint] appid= '{67998776-7e22-4753-be18-12951a4e15d3}'

where **thumbprint** is the thumbprint of the SSL certificate

example

netsh http add sslcert ipport=0.0.0.0:9010 certhash=9c4b3797bbca9c233f690ec4b276871a6182b4e1 appid= '{67998776-7e22-4753-be18-12951a4e15d3}'

1. Restart the STSOrgSync service

After these steps, SSL has been enabled, and the service can be accessed on https rather than http.

# Certificate renewal

Certificates expire, and FOCES certificate have a validity period of 3 years, so at least once every 3 years, the certificates used by STSOrgSync needs to be renewed.

The renewal process itself is handled through the same OCES self-service portal that is used for ordering certificates, and the certificate does not actually need to be renewed, simply ordering a new certificate is also a valid approach.

The exact details on how to update the certificate in KOMBITs self-service portal 3 years from now is just plain guess-work, but it will likely work like this

1. Log onto the self-service portal
2. Locate the STSOrgSync it-system
3. Add the new certificate to the it-system (do not remove the old one yet)
4. Update the certificate in STSOrgSync (details below)
5. Go back to the self-service portal and remove the old certificate

By keeping the old certificate while performing step 4, we ensure that the system still works while we are updating the certificate, and we can even perform step 3 several days before we do step 4+5.

Updating the certificate in STSOrgSync is the same as installing the original certificate, meaning we will have to perform the following steps

1. Install the new certificate using mmc (or a similar tool), as described in chapter 3.1
2. Make sure to update Windows Registry with the new thumbprint
3. Restart STSOrgSync
4. Check the logs, and see that everything is running as it should

## Renewal of KOMBITs certificates

As a special case of the above scenario, the certificates used by KOMBIT also expires at some point, which means that the certificates configured in “Trusted People” also needs to be updated.

Exactly how KOMBIT intends to distribute the new certificates when the old expire, and how they intend to inform their customers, is currently unknown.

# Logging and monitoring

As STSOrgSync will run as a background task, it is vital that the process is monitored so any blocking issues can be resolved.

For that purpose, STSOrgSync will log all relevant information to a log file, which can be inspected, both to monitor the health of the process, but also to locate the cause of any errors that may occur.

Besides the log file, STSOrgSync also does internal monitoring, and the state of this monitoring is exposed through the APIs. For the Windows Service, this means a REST endpoint that can be monitored, and for the SDK, it is an operation that can be called in the SDK API.

## Logging

STSOrgSync uses log4net for logging, and a set of configuration parameters to control how much information is logged. Note that any change in configuration requires STSOrgSync to be restarted, as the configuration is read at startup only.

### Windows Registry setting

The setting ‘LogRequestResponse’ in Windows Registry is by default set to ‘false’, but during development, and when trying to track down a problem, it might be useful to enable logging of the full SOAP requests being sent to Organisation.

Set this registry setting to true, and restart STSOrgSync. After that, the log file will grow at a rapid rate, as each request/response takes up 10+ kb of data, and synchronization of a single object might require 5-10 sets of request/response.

Remember to disable this setting (and restart STSOrgSync) when done debugging.

### Log.config file

The log4net settings are stored in a file called Log.config. If this file does not exist, nothing will be logged to the log-file. STSOrgSync comes with a pre-configured Log.config file, which has file rollover enabled, so no more than 10 logfiles of 100kb each are stored. These settings are easily changed in Log.config.

Log.config also contains the path and filename of the logfile, which by default is stored in

C:\logs\STSOrgSyncer\system.log

## Monitoring

The self-monitoring code performs the following checks when invoked

* Can the Token Service be accessed (it attempts to retrieve a token)?
* Can the Organisation services be accessed (it attempts to retrieve the schema definition from each of the Organisation services)?
* Can the configured database be accessed (simple read validation is performed)?

The last check is only performed if a database is configured (not required when using the SDK API).

The result is a Health object, returned either as a .NET object instance (SDK API) or a JSON structure (Service API). Note that those using the SQL API will have to use the Service API for monitoring.

The resulting structure looks like this

GET **/**api**/**health HTTP**/**1.1

**{**

"STSstatus"**:** **true**

"ServiceStatus"**:** **true**

"DBStatus"**:** **true**

**}**