



Kofax SignDoc Standard Installation Guide

Version: 3.3.0

Date: 2023-06-21

KOFAX

© 2014–2023 Kofax. All rights reserved.

Kofax is a trademark of Kofax, Inc., registered in the U.S. and/or other countries. All other trademarks are the property of their respective owners. No part of this publication may be reproduced, stored, or transmitted in any form without the prior written permission of Kofax.

Table of Contents

Preface	5
Product documentation.....	5
Offline documentation.....	6
Training.....	7
Getting help with Kofax products.....	7
Chapter 1: Introduction	9
General overview.....	9
Chapter 2: Base installation	10
Installation as a Windows service.....	10
Definitions.....	10
General prerequisites.....	10
Quick start.....	11
Content of the SignDoc Standard ZIP archive.....	12
Production setup.....	12
Advanced configuration.....	15
Configuration backup.....	17
View service details.....	18
Logging.....	18
Prepare Microsoft SQL database.....	19
Monitor application using JMX.....	19
Installation on Tomcat without using the provided service installer.....	23
Installation on other JEE-compliant application servers.....	24
Device Connector - Certificate provider plug-in.....	24
Uninstall SignDoc Standard.....	24
Install and configure Microsoft SQL Server.....	25
Database installation.....	25
Database engine configuration.....	25
Create accounts and users.....	27
Deployment on Linux.....	27
Installation in Docker environment.....	27
Docker examples.....	29
Base image.....	30
High-availability image.....	31
High-availability with HTTPS.....	31

Configure with environment variables (Windows only).....	32
Chapter 3: Advanced installation.....	35
Hardening a SignDoc installation.....	35
Content Security Policy (CSP).....	36
Authentication LDAP.....	37
Windows Authentication.....	39
Chapter 4: SignDoc Authentication Module (SAM).....	40
Overview.....	40
Installation.....	40
SAM configuration with application.yml.....	41
SignDoc Standard configuration.....	42
SSO configuration in Manage Client and Administration Center.....	43
Step by step example.....	44
Chapter 5: Upgrade SignDoc Standard.....	46
Upgrade from SignDoc Standard 2.2.1.....	46
Upgrade from SignDoc Standard 1.3.1 or earlier versions.....	47
Upgrade troubleshooting.....	47
Chapter 6: Database migration.....	48
Database migration with Flyway.....	48
Load balancing considerations.....	48
Chapter 7: Troubleshooting.....	50
General.....	50
Email settings.....	50
Apache Tomcat.....	51
Appendix A: Glossary.....	52

Preface

This guide contains information that administrators need to install or upgrade Kofax SignDoc Standard.

For information on supported operating systems and other system requirements, see the [Kofax SignDoc Technical Specifications](#) document.

This document is updated regularly, and we recommend that you review it carefully to ensure success with your Kofax SignDoc product.

Product documentation

The full documentation set for SignDoc Standard is available at the following location:

<https://docshield.kofax.com/Portal/Products/SD/3.3.0-t85kv64y7c/SD.htm>

In addition to this guide, the documentation set includes the following items:

Release notes

- *Kofax SignDoc Release Notes*

Technical specifications

- *Kofax SignDoc Technical Specifications*

Guides

- *Kofax SignDoc Standard Administrator's Guide*
- *Kofax SignDoc Standard Developer's Guide*

Help

- *Kofax SignDoc Standard Help*
- *Kofax SignDoc Standard Administration Center Help*
- *Kofax SignDoc Assistant App Help*
- *Signing Documents with Kofax SignDoc Help*
- *Kofax SignDoc Device Connector Help*

Software development kit

- *Kofax SignDoc Browser Capture Help*
- *Kofax SignDoc SDK API Documentation (C)*
- *Kofax SignDoc SDK API Documentation (C++)*
- *Kofax SignDoc SDK API Documentation (.NET with exceptions)*


- *Kofax SignDoc SDK API Documentation (.NET without exceptions)*
- *Kofax SignDoc SDK API Documentation (Java)*

Offline documentation

Customers who require offline documentation can download the English documentation package `KofaxSignDocDocumentation_3.3.0_EN.zip` from the [Kofax Fulfillment](#) site. The .zip file includes the `help` directory where you can find the SignDoc help files and the `print` directory with the SignDoc guides.

Kofax SignDoc Standard help

The following steps describe how to make the English offline help accessible in SignDoc Standard (Administration Center, Manage Client, and Signing Client) by copying the help to the internal web server (Tomcat) for the installation.

 Before proceeding, you must install SignDoc Standard in the directory `<INSTALLDIR>` and set the `<SERVICE_EXTERNAL_HOST_URL>` as described in this guide.

1. From the Kofax Fulfillment site, download `KofaxSignDocDocumentation_3.3.0_EN.zip`.
2. Extract the contents of the .zip file to any directory `<EXTRACTDIR>`.
3. Copy `<EXTRACTDIR>/help` to the directory `<INSTALLDIR>/service/webapp`.
4. Start SignDoc Standard and configure the help links in the System settings of the Administration Center.
 - a. Manage Client
Open the subcategory Client > Manage and edit "Manage Client online help URL" by entering the URL `<SERVICE_EXTERNAL_HOST_URL>/help/Standard/index.html`.
 - b. Signing Client
Open subcategory Client > Signing and edit "Signing Client online help URL" by entering the URL `<SERVICE_EXTERNAL_HOST_URL>/help/StandardSigningDocuments/index.html`.
 - c. Administration Center
Open subcategory Client/Administration and edit "Administration Center online help URL" by entering the URL `<SERVICE_EXTERNAL_HOST_URL>/help/StandardAdministrationCenter/index.html`.
5. Test the configured links by clicking the Help link in the header of Administration Center, Manage Client and Signing Client. Each help system should display in a new browser tab.

Kofax SignDoc Standard guides

From the directory `<EXTRACTDIR>/print`, you can access the following guides:

- *Kofax SignDoc Standard Administrator's Guide*
`KofaxSignDocStandardAdministratorsGuide_EN.pdf`
- *Kofax SignDoc Standard Developer's Guide*
`KofaxSignDocStandardDevelopersGuide_EN.pdf`
- *Kofax SignDoc Standard Installation Guide*

KofaxSignDocStandardInstallationGuide_EN.pdf

SignDoc Software Developer Kit documentation

According to the functionality and the programming language, the offline documentation .zip file contains documentation for the SignDoc Software Developer Kit.

To open and use the SignDoc Software Developer Kit documentation, follow these steps:

1. From the Kofax Fulfillment site, download `KofaxSignDocDocumentation_3.3.0_EN.zip`.
2. Extract the contents of the .zip file to any directory <EXTRACTDIR>.
3. Navigate to <EXTRACTDIR>/help to access the SignDoc Software Developer Kit documentation.


Training

Kofax offers both classroom and online training to help you make the most of your product. To learn more about training courses and schedules, visit the [Kofax Education Portal](#) on the Kofax website.

Getting help with Kofax products

The [Kofax Knowledge Portal](#) repository contains articles that are updated on a regular basis to keep you informed about Kofax products. We encourage you to use the Knowledge Portal to obtain answers to your product questions.

To access the Kofax Knowledge Portal, go to <https://knowledge.kofax.com>.

 The Kofax Knowledge Portal is optimized for use with Google Chrome, Mozilla Firefox, or Microsoft Edge.

The Kofax Knowledge Portal provides:

- Powerful search capabilities to help you quickly locate the information you need.
Type your search terms or phrase into the **Search** box, and then click the search icon.
- Product information, configuration details and documentation, including release news.
To locate articles, go to the Knowledge Portal home page and select the applicable Solution Family for your product, or click the View All Products button.

From the Knowledge Portal home page, you can:

- Access the Kofax Community (for all customers).
On the Resources menu, click the **Community** link.
- Access the Kofax Customer Portal (for eligible customers).
Go to the [Support Portal Information](#) page and click **Log in to the Customer Portal**.
- Access the Kofax Partner Portal (for eligible partners).
Go to the [Support Portal Information](#) page and click **Log in to the Partner Portal**.

- Access Kofax support commitments, lifecycle policies, electronic fulfillment details, and self-service tools.
Go to the [Support Details](#) page and select the appropriate article.

Chapter 1

Introduction

This guide includes instructions for installing and upgrading Kofax SignDoc Standard, along with information on authentication and database installation, configuration, and migration.

General overview

Operating systems

SignDoc Standard can be installed on Windows and Linux operating systems with a 64-bit architecture.

SignDoc Standard application

The application consists of two WAR (web application archive) files that are deployed into a web application server. The `sdweb.war` file contains Native Libraries used for licensing and PDF handling. The configuration files are stored outside of the WAR files in a directory referenced by the environment variable `SIGNDOC_HOME`.

REST interface

SignDoc Standard can interact with the system via the REST API that supports almost all aspects of the application. It is possible to create and schedule signing packages with one REST request. For details, see the [Kofax SignDoc product documentation](#) page.

Web application server

SignDoc Standard runs on Apache Tomcat, the standard web application server used within the industry. For supported versions and prerequisites, see the [Kofax SignDoc Technical Specifications](#) document.

Chapter 2

Base installation

This guide explains how to install Kofax SignDoc Standard on a Windows 64-bit operating system. Sample screens in this guide may vary from your installation, depending on your version of Windows. For an installation on Linux systems, see the [Installation in Docker environment](#).

The installed system always contains these components:

- Database (Microsoft SQL Server)
- Application Server (Apache Tomcat and SignDoc Standard)

SignDoc Standard is run as a J2EE compatible application in the Application Server. Apache Tomcat and Microsoft SQL Server are usually installed on different computers (nodes) for various reasons, but they can also be installed on the same computer (node).

Reverse proxy / Load balancing

This installation guide does not cover the setup of a reverse proxy or load balancing.

SSL setup

This installation guide does not consider/discuss an SSL configuration, which usually depends on local IT regulations and is effectively transparent to SignDoc Standard.

Installation as a Windows service

This section provides information on installing SignDoc Standard as a Windows service.

Definitions

- <INSTALLDIR> is the directory of the unpacked `signdoc_standard*-tomcat.zip` file.
- <CIRRUS_HOME> is the home directory of the web application (cirrus).

General prerequisites

Before starting the installation, you must verify that your system meets the requirements in the [Kofax SignDoc Technical Specifications](#) document.

Make sure that the Windows PowerShell `powershell.exe` is in the system path.

If you plan to upgrade an existing SignDoc Standard installation, see [Upgrade SignDoc Standard](#).

i We recommend installing SignDoc Standard behind a reverse proxy. If the reverse proxy is also used for load balance requests, they can be handled as stateless, such as round robin.

Quick start

Getting a simple local accessible SignDoc Standard installation running can be achieved in less than 5 minutes. The local installation can serve as a base for a production-ready setup.

Quick start goals

- Install SignDoc Standard as a Windows service.
- Database: preconfigured for a local file-based H2 database.

⚠ The data stored in the local file-based H2 database cannot be migrated to a production database based on Microsoft SQL Server.

- SMTP configuration: preconfigured for localhost with port 1025 (no authentication or encryption).
 - Works with [MailHog](#) (email testing tool).
 - MailHog can be stopped and deleted at any time. After a working SMTP server is configured, MailHog is not needed.

Quick start prerequisites

- 8 GB RAM
- Run MailHog after you download it from the GitHub website.
If this is not possible or not desired, an SMTP server must be configured first. See [Configure SMTP server connection](#).
- Access MailHog: `http://localhost:8025` (if applicable).
- Enable startup email feature. See "Startup email" in the [Production setup](#) section.

i MailHog is only needed for the quick start scenario. When using startup email with a valid SMTP server, make sure to use a real email address.

Quick start procedure

1. Double-check that [General prerequisites](#) are fulfilled.
2. Extract `signdoc-standard*-tomcat.zip` in a new directory `<INSTALLDIR>`.
Example:
`C:\Program Files\signdoc-standard-3.3.0`
3. Double-click `<INSTALLDIR>\service_up.cmd`.
4. Wait approximately 1 minute on first start.
5. A SignDoc Standard startup email should be sent to the specified startup email recipient while starting up.
6. Open SignDoc Standard: `http://localhost:6611`.

Content of the SignDoc Standard ZIP archive

The relevant and configurable content of SignDoc Standard consists of 3 files:

- Configuration file
- Script to install and configure the SignDoc Standard Windows service
- Script to deregister the SignDoc Standard Windows service

Tools

- `<INSTALLDIR>\service_up.cmd` installs, applies configuration, and restarts the SignDoc Standard Windows service.
- `<INSTALLDIR>\service_remove.cmd` stops and deregisters the SignDoc Standard Windows service. No files are deleted.
- `<INSTALLDIR>\service_configuration.properties` is the configuration file of the SignDoc Standard Windows service. This file can be edited with a standard text editor. The syntax and usage is described in the file.

Directories

- `<INSTALLDIR>\signdoc_home` is the consolidated default `<CIRRUS_HOME>` directory.

Production setup

The following sections describe basic tasks for a production setup.

Goals for production

- Configure SMTP server connection
- Configure database connection
- Configure network settings
- Configure reverse proxy setup (optional)
- Advanced configuration (optional)
- Kofax TotalAgility integration (optional)
- LDAP integration (optional)

Prerequisites for production setup

- Application server
 - Minimum 8 GB RAM. See "Tune Java memory settings" in the [Advanced configuration](#) section.
 - Minimum 2 GB free disk space
- Database
 - Installed Microsoft SQL Server with a database for SignDoc Standard and a database user with database owner (dbo) credentials for this database. See [Prepare Microsoft SQL database](#).
- SignDoc Standard
 - Install SignDoc Standard as described in [Quick start](#).

Procedure for production

The following topics do not depend on each other and can be executed independently.
What is common for all settings: The settings must be applied by executing `<INSTALLDIR>\service_up.cmd`.

Configure SMTP server connection

A valid and trustworthy SMTP connection is required to be able to send emails.

1. Open the `<INSTALLDIR>\service_configuration.properties` file in a text editor.
2. Navigate to # [EMAIL CONFIGURATION].
3. Amend the settings with the configuration parameters of a valid SMTP server.
See also the commented examples at the bottom of `service_configuration.properties`.

SMTP TLS example

```
mail.smtp.host=email-smtp.us-east-1.amazonaws.com
mail.smtp.port=587
mail.smtp.user=<Access key ID>
mail.smtp.from=dont_reply@mydomain.com
mail.smtp.password=<Secret access key>
mail.smtp.starttls.enable=true
mail.smtp.starttls.required=true
mail.smtp.ssl.checkserveridentity=false
```

Startup email

It can be useful to send a startup email to a predefined address whenever the SignDoc Standard server is starting. SignDoc Standard can be configured for this purpose. The startup email contains information about configuration, the environment, and start parameters for the SignDoc Standard application.

To enable the startup email

1. Open `<INSTALLDIR>\service_configuration.properties` in a text editor.
2. Navigate to # [EMAIL CONFIGURATION].
3. Uncomment the line starting with `#cirrus.startup.email`.
4. Set a valid email address.

Example

```
cirrus.startup.email=ksdadmin@localhost
```

Configure JDBC Server connection



- For production purposes, only Microsoft SQL Server is supported.
- Confirm that `jdbc.url` is not enclosed in single quotation marks.

To configure JDBC Server connection

1. Open `<INSTALLDIR>\service_configuration.properties` in a text editor.

2. Navigate to # [DATABASE CONFIGURATION]
3. Amend the setting with the configuration parameters and credentials of the JDBC connection.
See also the commented examples at the bottom of `service_configuration.properties`.

Microsoft SQL Server example

```
jdbc.url=jdbc:sqlserver://my-mssql-server:1433;databaseName=signdoc
jdbc.username=signdoc
jdbc.password=2beChanged!
```

Configure network settings

The following network settings must be configured.

- SERVICE_HTTP_PORT
- SERVICE_EXTERNAL_HOST_URL

To configure SERVICE_HTTP_PORT

1. Open `<INSTALLDIR>\service_configuration.properties` in a text editor.
2. Navigate to SERVICE_HTTP_PORT.
3. Set the preferred port number.

Example

```
SERVICE_HTTP_PORT=6611
```

For SERVICE_EXTERNAL_HOST_URL a production service must be accessible via the official domain name, so it can be accessed from other computers. See "Configure reverse proxy setup" in [Production setup](#).

HTTPS/TLS support

While it is possible to use TLS with SignDoc directly, it is generally recommended to use a reverse proxy to offload the TLS connections. This reduces the load and provides more flexibility for hosting and maintaining the SignDoc application.

To enable HTTPS/TLS, update the configuration:

1. Edit `<INSTALLDIR>\service_configuration.properties`.
Use `https://` for the SERVICE_EXTERNAL_HOST_URL setting.

Example

```
https://localhost:${SERVICE_HTTP_PORT}
```

2. Edit `<INSTALLDIR>_conf_templates\server.xml`.
 - Comment the default `http` connector (as described in the documentation notes within the file).
 - Uncomment and configure the `https` connector (as described in the documentation notes within the file).
 - By default, the `https` connector is using a self-signed certificate that can only be used for test purposes.
 - To use an individual and trustworthy certificate `keystoreFile`, `keystorePass`, `keyAlias` must be adjusted.

- We recommend using a PKCS#12 certificate store (*.pfx, *.p12) that contains a private key as well as all required certificates.

3. Apply the configuration and run `service_up.cmd` to restart the service.

Configure reverse proxy setup

In a reverse proxy scenario it is important to configure the application URLs correctly.

1. Open `<INSTALLDIR>\service_configuration.properties` in a text editor.
2. Navigate to `SERVICE_EXTERNAL_CONTEXT_URL`.
This is the context URL that is used to access the application. This URL must be reachable from anywhere and is part of the signing links that are sent via email.
3. Change the values if required.

Example

```
SERVICE_EXTERNAL_CONTEXT_URL=https://signdoc.mydomain.com
```


Kofax TotalAgility integration

The Kofax TotalAgility connection is individually defined per account/tenant in the SignDoc Standard Manage Client or SignDoc Standard Administration Center. For details, see "Plug-ins" topic in the [SignDoc Standard Administration Center Help](#) and "KTA state change plug-in" section in the [SignDoc Standard Administrator's Guide](#)

LDAP integration

LDAP (or Active Directory) can be used to authenticate SignDoc Standard users in the Manage Client.

Configure LDAP settings in `<INSTALLDIR>\service_configuration.properties` (section # LDAP integration) as described in the [Authentication LDAP](#) chapter.

 We recommend that you perform the configuration in `<INSTALLDIR>\service_configuration.properties` instead of `cirrus.properties`.

See also [Advanced configuration - Option 2](#).

Advanced configuration

To configure more features for SignDoc Standard, you have two options:

Option 1: Configuration using `service_configuration.properties` (generally recommended)


1. Edit `<INSTALLDIR>\service_configuration.properties`.
2. Amend `service_configuration.properties` with additional settings.

Option 2: Configuration using `cirrus.properties` (backward compatible and for special configurations)

1. Edit `<INSTALLDIR>_conf_templates\cirrus.properties`.

The default location is `<INSTALLDIR>\bin\signdoc_home\conf\cirrus.properties`.

2. Apply the configuration by running `<INSTALLDIR>\service_up.cmd`.

 The configuration defined in `<INSTALLDIR>\service_configuration.properties` takes precedence over settings with the same name in `cirrus.properties`.

Special configuration settings

The following configuration setting should always be configured in `cirrus.properties` and not in `service_configuration.properties`:


- `ldap.user.search.filter`

Example

```
ldap.user.search.filter=(&(objectClass=person)(|(cn=John Doe)
(mail=john.doe@example.com)))
```

- `jdbc.password`

The `jdbc.password` setting must consist only of ASCII characters.

 If characters outside the ASCII character set are being used, the server cannot start up.

This restriction does not apply if the password is declared as an encrypted configuration property.

Encrypted configuration properties

It is possible to create symmetrically encrypted configuration data. This is a pragmatic approach for SignDoc Standard to protect sensitive configuration data. By default, the data is encrypted using an AES-256 key. Configuration is applied using `service_up.cmd`.

1. Delete or comment the `property_name` from `service_configuration.properties`; otherwise, the encrypted value is not used.
2. Double-click the command `service/speh.cmd`.
3. In the input field enter the configuration to encrypt (such as the jdbc password). The next dialog box displays the encrypted data.
4. The next dialog box displays the configuration entry that must be added to `<INSTALLDIR>_conf_templates\cirrus.properties`. You must replace it with the correct property name.
5. Restart the service with `service_up.cmd`.

Here is a step-by-step example for configuration property `jdbc.password`.

1. Delete or comment the `jdbc.password` setting from `<INSTALLDIR>\service_configuration.properties`.
2. Double-click the command `service/speh.cmd`.
3. Enter "1234" as the password.
The next dialog box shows `2682e444e7935f2af0b9e20a4266e29b`.

The next dialog box shows

```
.encrypted_string_256=2682e444e7935f2af0b9e20a4266e29b.
```

4. Add `jdbc.password.encrypted_string_256=2682e444e7935f2af0b9e20a4266e29b` to `<INSTALLDIR>_conf_templates\cirrus.properties`.
5. Restart the service with `service_up.cmd`.

Control database migrations

If SignDoc Standard is run in a clustered environment, it makes sense to disable the automatic database migrations. See [Database migration](#).

1. Open `<INSTALLDIR>\service_configuration.properties` file in a text editor.
2. Navigate to # Database migrations section.
3. Set `cirrus.migrations.enabled` to `false`.

Example

```
# disable automatic migrations
cirrus.migrations.enabled=false
```

Tune Java memory settings

1. Open `<INSTALLDIR>_conf_templates\SignDocStandard.xml` in a text editor.
2. Look for the following lines and change the values to suit your needs:

```
<!-- minimum Java HEAP -->
<argument>-Xms1024m</argument>
```

```
<!-- maximum Java HEAP -->
<argument>-Xmx2048m</argument>
```

3. After revising these values, you must execute `service_up.cmd` to apply the values.

Use SignDoc Standard in a clustered environment

SignDoc Standard is supported in a clustered environment. A typical use case is load balancing.

The application server is stateless and can therefore be used with simple round-robin load balancing.

If multiple instances should be installed on the same operating system, you must ensure to use a different HTTP/TCP port for each instance. See "Configure network settings" in the [Production setup](#) section. The default HTTP/TCP port for SignDoc Standard is 6611.

i If the `server.xml` file must be changed, it should be done in the file `<INSTALLDIR>_conf_templates\server.xml`. Execute `service_up.cmd` to apply the change. See "Tools" in [Content of the SignDoc Standard ZIP archive](#) section.

Configuration backup

For a backup of the SignDoc Standard instance configuration it is sufficient to back up the files and directories listed below. Such a backup set can be applied 1:1 to a new SignDoc Standard installation

(such as for additional instances in a cluster). If a SignDoc Standard update is done, make sure to check [Upgrade SignDoc Standard](#) first. It is also possible to simply back up the complete directory.

Required

- `<INSTALLDIR>\service_configuration.properties`
Main configuration file of the SignDoc Service.

Optional

- `<INSTALLDIR>_conf_templates`
Contains the potentially modified configuration templates `cirrus.properties`, `logging.properties`, `server.xml`. If none of these files were modified, there is no need to back them up.
- `<INSTALLDIR>\signdoc_home\conf`
Contains basic configuration. If these files were not customized manually, there is no need to back them up.
- `<INSTALLDIR>\signdoc_home\fonts`
Contains the font configuration. If these files were not customized manually, there is no need to back them up.
- `<INSTALLDIR>\signdoc_home\db`
If existing, this directory contains the file-based default database. If this database is not used, there is no need to back up the files.

View service details

To view detailed service information, double-click:

```
<INSTALLDIR>\service\bin\SignDocStandard.exe
```

We do not recommend changing the settings with this tool, since they are being overwritten whenever `service_up.cmd` is executed. See "Tools" in [Content of the SignDoc Standard ZIP archive](#) section and [Advanced configuration](#).

Logging

Logging is configured in the SignDoc Administration Center.

There is also a configuration file (`<INSTALLDIR>\signdoc_home\conf\signdoc-logger.properties`), that is only considered if configuration options in the Administration Center are not changed. We recommend using the configuration options of the Administration Center since configuration changes are applied without having to restart the service.

The SignDoc Standard Windows service uses the file `<INSTALLDIR>\signdoc_home\conf\tomcat_logging.properties` for the logging configuration file of the Tomcat application server. Consult the Tomcat configuration if changes are needed.

Logging configuration options

The following configuration options can be set:

- `signdoc.logger.handler.enabled`

- `signdoc.logger.level`
- `signdoc.logger.custom_levels`
- `signdoc.logger.handler.console.enabled`
- `signdoc.logger.handler.file.enabled`
- `signdoc.logger.handler.logfile`
- `signdoc.logger.handler.date.format`
- `signdoc.logger.handler.logfile.maxsize`
- `signdoc.logger.handler.logfile.maxnumber`

For more details, see the [Kofax SignDoc Standard Administrator's Guide](#).

Prepare Microsoft SQL database

To be able to use SignDoc Standard for production usage, you must set up a database and database user that can be used by SignDoc Standard. This can either be achieved using the GUI tools provided by MS-SQL or with a T-SQL script.

Example T-SQL script

```
CREATE DATABASE signdoc
GO
USE signdoc
GO
CREATE LOGIN signdoc WITHPASSWORD='2beChanged!'
GO
CREATE USER signdoc FOR LOGIN signdoc
GO
ALTER ROLE db_owner ADD MEMBER signdoc
GO
```

Monitor application using JMX

You can monitor SignDoc using the provided JMX metrics. Besides the standard metric provided by the JMV, SignDoc offers metrics that are grouped below the SignDoc MBean node. By default, the JMX metrics can be queried by a local process running on the same server using the same user. If remote access is required, the `jmxremote.port` must be secured for production use. See [Enabling the Ready-to-Use Management](#).

Configuration

The JMX configuration options (find all options in the links below) are added without the `-D` prefix to `service_configuration.properties`. After restarting the service with `service_up.cmd`, the settings are applied.

The basic configuration options are:

- `com.sun.management.jmxremote=[true|false]`
Must be set to true to enable the remote JMX management functionality.
- `com.sun.management.jmxremote.port=<port_number>`
Defines the port JMX uses. If a firewall is used, make sure that this port is accessible.
- `com.sun.management.jmxremote.ssl=[true|false]`

Specifies if SSL should be used to encrypt the data. If this setting is set to true, further configuration options are required. For related information, see the links at the end of this section.

- `com.sun.management.jmxremote.authenticate=[true|false]`
Specifies if the user must authenticate to get authorization. If set to true, further configuration options are required. For related information, see the links at the end of this section.
- `java.rmi.server.hostname=<IP or hostname>`
The setting `java.rmi.server.hostname` restricts the access to the specified IP Address or hostname:
 - `java.rmi.server.hostname=127.0.0.1`
Makes the JMX port accessible only from localhost and not from a remote computer.
 - `java.rmi.server.hostname=<HOSTNAME>`
Makes the JMX port accessible only from any remote computer that can resolve and access the HOSTNAME.

Example

The following sample shows how to open an unsecured JMX port that is accessible from 127.0.0.1 (localhost) only. To secure the port, see [Enabling the Ready-to-Use Management](#).

```
com.sun.management.jmxremote=true
com.sun.management.jmxremote.port=1099
com.sun.management.jmxremote.ssl=false
com.sun.management.jmxremote.authenticate=false
java.rmi.server.hostname=127.0.0.1
```

SignDoc metrics

SignDoc offers metrics and operations below the domain. Supported SignDoc metrics can be found under SignDoc > Status > Global.

Attributes

Attribute	Description
Accounts	The list of all account IDs.
ActiveSigningSessions	The number of all started, but unfinished signing sessions across all accounts.
SysAdmins	The list of all system administrators.
TotalAccounts	The number of all accounts (regardless of status).
TotalDocuments	The number of all documents in the system.
TotalPackages	The number of all signing packages in the system (regardless of status).
TotalUsers	The number of all users in the system across all accounts.

Operations

```
getSignDocAccountStatus(String accountID)
```

This operation returns a map with account-specific information, as listed in the following table.

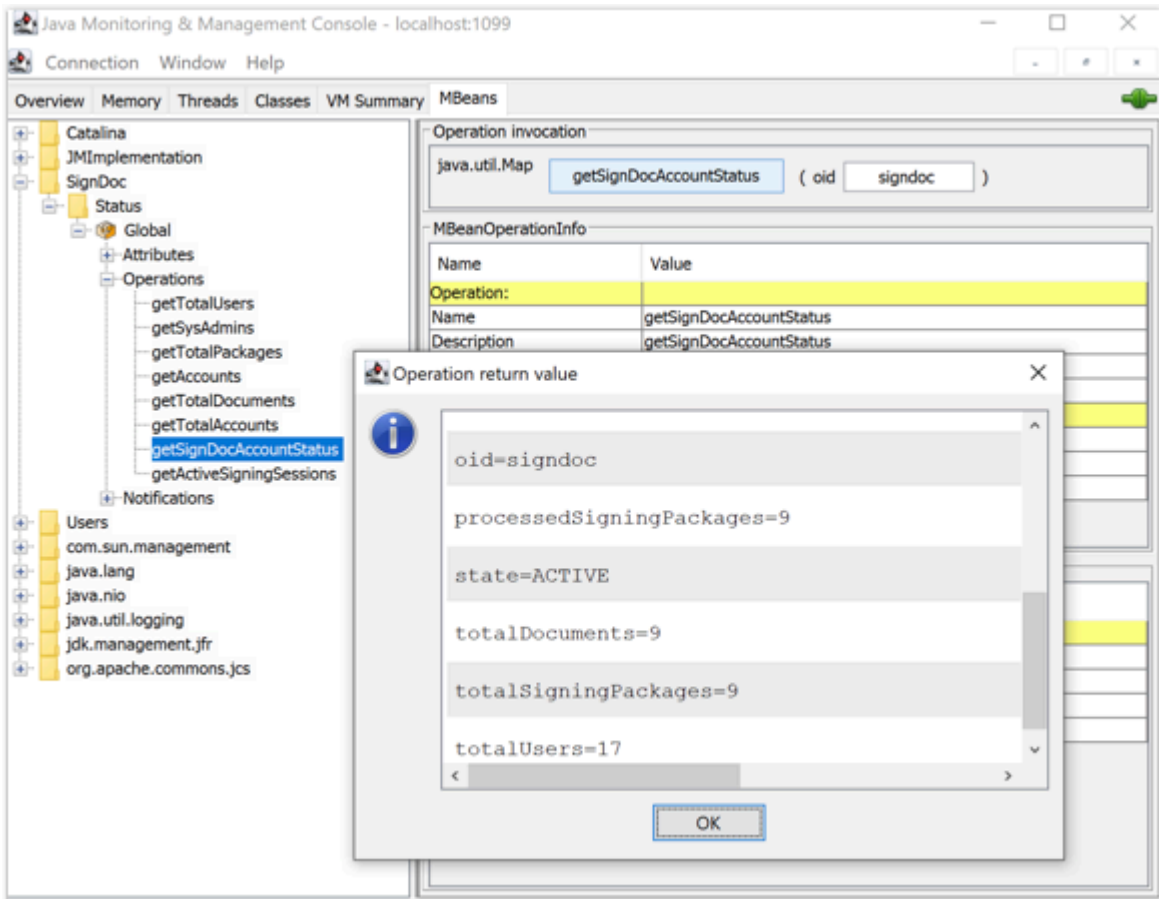
Attribute	Description
activeSigningSessions	The number of all started, but unfinished signing sessions of this account.
admins	The list of all account administrators.
contactInformation	Contact information of the account. Empty if not set.
lastRenewalTimestamp	Last time the license was renewed.
maxSigningPackages	Maximum number of allowed signing packages. -1 means unlimited.
maxUsers	Maximum number of allowed users. -1 means unlimited.
name	The account name.
oid	The account ID.
processedSigningPackages	The number of already processed signing packages.
state	The account state. Must be ACTIVE to be usable.
totalDocuments	The number of all documents in the account.
totalSigningPackages	The number of all signing packages in the system (regardless of status).
totalUsers	The number of all users in the system across all accounts.

If JConsole is used, it typically looks similar to the following screen.

The screenshot shows the Java Monitoring & Management Console window for localhost:1099. The 'MBeans' tab is active, and the tree view on the left shows the hierarchy: Catalina > JMIImplementation > SignDoc > Status > Global > Attributes. The 'Attributes' MBean is selected, and its values are displayed in a table on the right.

Name	Value
Accounts	[signdoc, screenshots, codededu...]
ActiveSigningSessions	0
SysAdmins	[ksdadmin]
TotalAccounts	4
TotalDocuments	9
TotalPackages	9
TotalUsers	29

Below the table is a 'Refresh' button.



For related information, see the following topics in the Help Center on the Oracle website:

- Monitoring and Management Using JMX Technology
- Enabling Ready-to-Use Management
- Using JConsole

JConsole Monitoring application is part of OpenJDK. See Adoptium website.

Installation on Tomcat without using the provided service installer

SignDoc Standard can be installed on an existing Tomcat server.

See the [Kofax SignDoc Technical Specifications](#) document for information about supported versions of Tomcat server and Java Runtime server you must use for the installation.

Follow these steps:

1. Extract the `cirrus.war` file manually in `%CATALINA_HOME%\webapps` so that there is a `cirrus` context directory.

2. Copy the following .jar files from the `service\lib` directory to the `lib` directory of the Tomcat directory (usually `CATALINA_HOME`):
 - `splm2jni-*.jar`
 - `SPSignDoc-*.jar`
3. Make sure that the directory with the native libraries `service\lib\native\Win64` is in the `PATH` of the Tomcat process.
4. Make sure that there are no older SignDoc native libraries in the `PATH` of the Tomcat process.
5. Copy the `signdoc_home` directory to the preferred location. This location must be readable and writable for the Tomcat process.
6. Copy `_conf_templates\cirrus.properties` to the `signdoc_home/conf` directory and configure it as desired.
7. To allow big file uploads it might be required to adjust the `maxPostSize` attribute of the `<Connector>` element of the Tomcats `server.xml`.
8. Start the Tomcat service with the following system properties.
Required properties:

```
-DCIRRUS_HOME=<path_to_signdoc_home_directory>  
-DSERVICE_EXTERNAL_HOST_URL=<generally_accessible_url_to_root_context>
```

Example:

```
-DCIRRUS_HOME="c:/signdoc_home"  
-DSERVICE_EXTERNAL_HOST_URL=http://mysigndocserver.example.com
```

Installation on other JEE-compliant application servers

Installation on other JEE-compliant servers is not supported.

Device Connector - Certificate provider plug-in

To use a certificate provider plug-in with SignDoc Standard, it must be recompiled due to improved encryption standards.

The following RSA parameters are required:

- Padding algorithm: OAEP
- Message digest: SHA-1
- Mask generation function: MGF1

Uninstall SignDoc Standard

To uninstall SignDoc Standard

1. Double-click `<INSTALLDIR>\service_remove.cmd`. Follow the prompts and wait until the Windows service "SignDoc Standard" is stopped and deregistered.
2. Delete the installation directory `<INSTALLDIR>`.

Install and configure Microsoft SQL Server

For production purposes, SignDoc Standard requires a database server to be able to store application data. Currently Microsoft SQL Server is supported. While installing the database server, use the suggested default settings unless noted otherwise.

In this guide Microsoft SQL Server 2019 is used as database service. For supported Microsoft SQL Server versions, see the [Kofax SignDoc Technical Specifications](#).

Database installation

This section describes an example of a database installation using Microsoft SQL Server 2019 Express.

To install Microsoft SQL Server 2019 Express

1. Download the latest Microsoft SQL Server 2019 Express version.
2. Run the installation with administrator rights (if required).
The **SQL Server Installation Center** window appears.
3. In the left navigation panel, click **Installation** and select **New SQL Server stand-alone installation or add features to an existing installation**.
After following the installation steps, the **SQL Server 2019 Setup** window appears.
4. In the left navigation panel, click **Feature Selection** and select all available features.
5. In the **SQL Server 2019 Setup** window, click **Instance Configuration**.
6. Then click **Server Configuration**.
7. Click the **Collation** tab.
8. Select a collation that is case-insensitive and also accent-sensitive (such as Latin1_General_CI_AS, which is the default for Microsoft SQL Server). This ensures that no duplicates are stored for particular data such as email addresses and object identifiers.
If the collation is not suitable, the application does not start. An `IllegalStateException` ("The database collation '<collationName>' is not suitable for the application.") is written to the log file in that case.

Database engine configuration

This section describes an example of a database engine configuration for Microsoft SQL Server 2019 Express.

To configure the database engine with Microsoft SQL Server 2019 Express


1. In the **SQL Server 2019 Setup** window, on the left navigation, click **Database Engine Configuration**.
2. Open the **Server Configuration** tab.
 - a. Select the **Mixed Mode** button.

For this guide, **2beChanged!** is used.

1. To create a new login, click **Security** in the **Object Explorer**, then right-click **Logins** and select **New Login**.

The **Login Properties** window appears.

2. Go to the **General** pane and enter the applicable information.

 Ensure that the check box is not selected for "User must change password at next login."

3. Go to **User Mapping** pane and select database **signdoc** and role membership **db_owner**.

Create accounts and users

Open the Administration Center to create accounts and users:

```
http[s]://<server>/cirrus/admin-center
```

In this guide this is:

```
http://<SERVICE_EXTERNAL_HOST_URL><SERVICE_HTTP_PORT>/cirrus/admin-center
```

See `service_configuration.properties` in [Production setup](#).

Example:

When `<SERVICE_EXTERNAL_HOST_URL>` is `localhost` and `SERVICE_HTTP_PORT` is `6611` the URL is:

```
http://localhost:6611/cirrus/admin-center
```


To create accounts and users, see the [Kofax SignDoc Standard Administration Center Help](#).

Deployment on Linux

If SignDoc is deployed under Linux, we recommend that you use Docker. SignDoc provides several sample Docker files (in the `docker` directory). These Docker files contain documentation and describe the basic installation procedure for a Linux environment. These files can also be used and amended/adapted for deployment, if required.

Installation in Docker environment

You can run SignDoc Standard in a Docker Linux or Windows container.

 Additional information is available in the `docker` directory of the SignDoc Standard ZIP archive.

Content of the docker directory

The `docker` directory in the SignDoc Standard ZIP archive consists of tools and subdirectories with example configurations.

The following tools are available:

- **build_images.cmd** builds the required Windows Docker images.
- **build_images.sh** builds the required Linux Docker images.
- **start_server.ps1** starts SignDoc Standard as service inside Docker Windows container.
- **start_server.sh** starts SignDoc Standard inside Docker Linux container.

The subdirectories are:


- **01_base_image** is an example of base configuration for running SignDoc Standard. Also contains SignDoc Standard Docker files that are used in other samples.
- **02_high_availability** is an example of how to run SignDoc Standard behind a load balancing reverse proxy. It is possible to scale SignDoc Standard instances transparently and independently of other services without breaking the client sessions.
- **03_high_availability_ssl** is an example of how to run SignDoc Standard in HTTPS configuration. This is an extension of the `02_high_availability` example.
- **04_env_variables** is an example of the SignDoc Standard configuration using environment variables. There is no need to rebuild the Docker image to apply new SignDoc Standard service configuration properties. Windows Docker containers only.
- **mssql_database** contains everything you need to build a Docker image of Microsoft SQL Server with prepared SignDoc Standard database.
- **plugins** contains extension files. You can implement your own logic for starting SignDoc Standard inside the Docker container. In this case, all additional files must be put in this directory. For instance, `04_env_variables` example has additional configuration checks for Mail and SQL servers. The PowerShell scripts are located in this directory. Windows Docker containers only.

How to run

Several options are available to run SignDoc Standard in a Docker container. Follow the recommendation on the official Docker website for your preferred orchestration tools.

As a basic example, you can run SignDoc Standard with Docker CLI:

```
docker run -init-d -p 8080:8080 -name=signdoc_standard_basicsigndoc_standard
```

 Due to a bug in Docker BuildKit, it might be required to do `docker login` before building the SignDoc images.

Docker examples

SignDoc Standard can be easily run in a Docker environment. This directory contains several sample Docker configurations with Linux and Windows containers.

All configurations below have in common that the resulting Docker images are stateless and self-contained. That is, the complete application configuration is embedded in the image and there is no data storage inside the application container. This is a stable and easy approach to use SignDoc Standard in production environments:

- No dependencies outside the container that influence or change the behavior of SignDoc.
- Services can be scaled by starting multiple instances of the same container. The load balancer can treat all instances in the same fashion.

The main thing to consider is, to rebuild the Docker images whenever the application configuration is changed. This is especially important for the `SERVICE_EXTERNAL_HOST_URL` build argument what defaults to `http://localhost:6611`. This constrains the usage of SignDoc Standard effectively to the local computer. For a real-world installation, this build argument must be changed to an externally reachable URL.


Prerequisites

Linux:


- Docker 17.12.0-ce or newer.
- Docker-Compose 1.18.0 or newer.

Windows:

- Docker 19.03.2 or newer.
- Docker-Compose 1.24.1 or newer.

 We highly recommend using Windows Server 2019 or higher as the Docker host computer.

Windows Server 2022 limitations

 Only `01_base_image` and `04_env_variables` samples are supported on Windows Server 2022.

To build a SignDoc Standard image under Windows Server 2022

1. Copy `Dockerfile.Windows.2022` to `Dockerfile.Windows`.
2. Build images with `build_images.cmd`.

Windows Server 2019

To build a SignDoc Standard image under Windows Server 2019

1. Copy `Dockerfile.Windows.2019` to `Dockerfile.Windows`.
2. Build images with `build_images.cmd`.

Windows Server 2016 limitations

The following are known issues on Windows Server 2016:

- Docker-Compose doesn't work.
- The `localhost` is not accessible.

As a result, SignDoc Standard can be run with Docker CLI only. To build a SignDoc Standard image under Windows Server 2016

1. Replace `localhost` with your host computer name in `service_configuration.properties`.
2. Copy `Dockerfile.Windows.2016` to `Dockerfile.Windows`.
3. Build images with `build_images.cmd`.

Build images

Run the script `build_images.sh` on Linux or `build_images.cmd` on Windows. After they are successfully built, all sample Docker images are ready to use.

Sample images

The default configuration starts SignDoc Standard and makes the application accessible via `http://localhost:6611` or `http://localhost` or `https://localhost` depending on the example that was used.

To make SignDoc Standard accessible from other computers:

1. Set `SERVICE_EXTERNAL_HOST_URL` to a logical value.
2. Rebuild the images.
3. Make sure that the port configuration fits to the used `SERVICE_EXTERNAL_HOST_URL`. In the case of docker-compose, the port configuration is defined in `docker-compose.yml` or `docker-compose-windows.yml`.

Base image

- Dockerfile: `01_base_image/Dockerfile`
- Dockerfile.Windows: `01_base_image/Dockerfile.Windows`

The base image provides the basic building block. All other images extend from this image. The application configuration is embedded in the image while building by copying configuration files in the image and setting required environment variables.

Run with Docker CLI

```
docker run --init -d -p 8080:8080 --name=signdoc_standard_basic
signdoc_standard
```

Run with Docker Compose

Follow these steps:

1. Open a command line terminal in the directory with the `docker-compose.yml` or `docker-compose-windows.yml`.

2. Run `run_with_compose.sh` or `run_with_compose.cmd`.

Access the application

Open `http://localhost:6611` with a browser.

High-availability image

- Docker-Compose: `02_high_availability/docker-compose.yml`
- Docker-Compose-Windows: `02_high_availability/docker-compose-windows.yml`

This image runs SignDoc Standard behind a load balancing reverse proxy (ha-proxy) using a sticky session configuration. This makes it possible to scale the SignDoc Standard application transparently and independently of other services without breaking the clients' sessions.

If one of the SignDoc Standard instances fails or shuts down (for whatever reason), the health check monitoring removes the failing instance from the load balancing and the remaining SignDoc Standard instances processes the load.

Run with Docker Compose

Follow these steps:

1. Open a command line terminal in the directory with the `docker-compose.yml` or `docker-compose-windows.yml`.
2. Run `run_with_compose.sh` or `run_with_compose.cmd`.

Access the application

Open `http://localhost:6611` with a browser.

High-availability with HTTPS

- Docker-Compose: `03_high_availability_ssl/docker-compose.yml`
- Docker-Compose-Windows: `03_high_availability_ssl/docker-compose-windows.yml`

This image extends the high-availability image and adds an HTTPS configuration. The HTTPS is handled and enforced by the load balance, which means that plain HTTP requests are automatically redirected to the corresponding HTTPS request. The SignDoc Standard application does not need extra configuration, since it is handled by the load balancer.

To achieve this, the SignDoc Standard container contains an ENV variable with an SSL certificate. A self-signed sample certificate in the correct form can be created with these statements:

```
openssl req -x509 -newkey rsa:2048 -keyout /tmp/key.pem -out /tmp/ca.pem -days 1080 -nodes -subj '/CN=*/O=SSL Example/C=US'
cp /tmp/key.pem /tmp/cert.pem
cat /tmp/ca.pem >> /tmp/cert.pem
awk 1 ORS='\n' /tmp/cert.pem
```

The browser requires that you explicitly and manually trust the certificate before it connects to SignDoc Standard if you use a self-signed certificate.

A trustworthy certificate that is accepted automatically by the browser must be countersigned or issued by a trustworthy CA.

Run with Docker Compose

With Linux, follow these steps:

1. Open a command line terminal in the directory with the `docker-compose.yml`.
2. Run `run_with_compose.sh`.

With Windows, follow these steps:

1. Update `SERVICE_HTTP_PORT` and `SERVICE_EXTERNAL_HOST_URL` in `service_configuration.properties` (or simply replace `service_configuration.properties` with `service_configuration_ssl.properties`)
2. Rebuild the docker images.
3. Open a command line terminal in the directory with `docker-compose-windows.yml`.
4. Run `run_with_compose.cmd`.
5. Run `stop_with_compose.cmd` to stop.

Access the application

Open `https://localhost` with a browser.

Configure with environment variables (Windows only)

- Docker-Compose-Windows: `04_env_variables/docker-compose-windows.yml`

This image uses Docker `04_env_variables/signdoc.env` to configure SignDoc Standard. Also, the startup powershell script `04_env_variables/start_server.ps1` contains additional configuration checks for Mail and SQL servers. This prevents SignDoc Standard server from starting if there are configuration or network errors.

Run with Docker Compose

Follow these steps:

1. Replace `start_server.ps1` with `04_env_variables/start_server.ps1` from the `04_env_variables` folder.
2. Run the `build_images.cmd` script to build the image. This is required once. Then the image can be restarted without rebuilding.
3. Open a command line terminal in the directory with `docker-compose-windows.yml`.
4. Run `run_with_compose.cmd`.
5. Run `stop_with_compose.cmd` to stop.

Access the application

Open `http://localhost:6611` with a browser.

Apply new configuration settings and run with Docker CLI

Update `signdoc.env` with your configuration settings.

It is possible to use the following environment variables for the SignDoc Standard configuration:

- SERVICE_EXTERNAL_HOST_URL
- SERVICE_HTTP_PORT
- JDBC_URL
- JDBC_USERNAME
- JDBC_PASSWORD
- MAIL_SMTP_HOST
- MAIL_SMTP_PORT
- MAIL_SMTP_FROM
- MAIL_SMTP_USER (optional)
- MAIL_SMTP_PASSWORD (optional)
- MAIL_SMTP_STARTTLS_ENABLED (optional)
- MAIL_SMTP_STARTTLS_REQUIRED (optional)
- MAIL_SMTP_SSL_CHECKSERVERIDENTITY (optional)

For example, you can change `signdoc.env` similar to this example:

```
JDBC_URL=jdbc:sqlserver://signdoc.company.com:1433;databaseName=db_name
JDBC_USERNAME=db_user
JDBC_PASSWORD=db_user_password
MAIL_SMTP_HOST=signdoc.company.com
MAIL_SMTP_PORT=1025
MAIL_SMTP_FROM=ksdadmin@signdoc.company.com
SERVICE_HTTP_PORT=6622
SERVICE_EXTERNAL_HOST_URL=http://signdoc.company.com
```

Restart the image to apply the new SignDoc Standard configuration (you don't need to rebuild the image). Don't forget to publish a container's port to the host.

To restart the image, you can use Docker CLI:

```
docker run -p 6622:6622 --env-file ./signdoc.env signdoc_standard:3.3.0
```

New configuration settings are validated before the SignDoc Standard server is started. The results are displayed in the console:

```
Mandatory environment variables: PASS
```

```
SignDoc database configuration: PASS
```

```
SignDoc email configuration: PASS
```

i In case of invalid settings or network issues, the corresponding error with a description are returned. SignDoc Standard server will not start. Also, validation of SMTP settings is limited. If the SMTP server supports several configurations at the same time, you may encounter the case when SignDoc email configuration is passed, but SignDoc Standard server cannot send emails. In most cases, this is due to TLS settings. Follow the recommended settings of the SMTP provider. See the mail settings description below.

To access the application open `http://signdoc.company.com:6622` with a browser.

Mail settings description

- **MAIL_SMTP_HOST (required)**
The SMTP server to connect to.
- **MAIL_SMTP_PORT (optional)**
The SMTP server port to connect to.
- **MAIL_SMTP_FROM (required)**
Email address to use for SMTP MAIL command. This sets the envelope return address.
- **MAIL_SMTP_USER (optional)**
User name for authentication.
- **MAIL_SMTP_PASSWORD (optional)**
Password for authentication.
- **MAIL_SMTP_STARTTLS_ENABLED (optional)**
If true, enables the use of the STARTTLS command (if supported by the server) to switch the connection to a TLS-protected connection before issuing any login commands. Defaults to false.
- **MAIL_SMTP_STARTTLS_REQUIRED (optional)**
If true, requires the use of the STARTTLS command. If the server doesn't support the STARTTLS command, or the command fails, the connect method will fail. Defaults to false.
- **MAIL_SMTP_SSL_CHECKSERVERIDENTITY (optional)**
If set to true, checks the server identity as specified by RFC 2595. Defaults to false.

For example, to use Gmail's SMTP server, you need the following settings for your outgoing emails:

```
MAIL_SMTP_HOST=smtp.gmail.com
MAIL_SMTP_PORT=587
MAIL_SMTP_FROM=ksdadmin@localhost
MAIL_SMTP_USER=your_name@gmail.com
MAIL_SMTP_PASSWORD=your_password
MAIL_SMTP_STARTTLS_ENABLED=true
MAIL_SMTP_STARTTLS_REQUIRED=true
MAIL_SMTP_SSL_CHECKSERVERIDENTITY=false
```

Chapter 3

Advanced installation

After you have set up SignDoc Standard as described in the base installation, you can continue with the procedures in this chapter, if necessary.

Hardening a SignDoc installation

Hardening a SignDoc setup means to apply best practices and security measures to a SignDoc installation for production usage.

Reverse proxy

We recommend running SignDoc behind a reverse proxy, since this provides an additional abstraction layer between the application server and the users. The reverse proxy:

- Can act as a TLS/SSL endpoint for the system, which simplifies deployment and maintenance.
- Is usually capable of load-balancing requests to multiple SignDoc installations, which improves the high-availability of an installation.
- Can be configured to set specific HTTP header attributes to minimize commonly known attack vectors such as XSS.

HTTP headers

Additional HTTP response headers can be set or added using the configuration options `security.http.response.headers.set` or `security.http.response.headers.add` of the Administration Center.

Make sure to set these HTTP headers concerning security:

```
X-Frame-Options "SAMEORIGIN";  
X-Content-Type-Options "nosniff";  
X-XSS-Protection "1; mode=block";
```

Turn off server tokens

Application servers and reverse proxies often announce their identity and version via server tokens in the HTTP response. This information is superfluous and should be avoided.

Block excessively large uploads

The reverse proxy should reject upload requests that are too big. But it must also allow uploads that are justified. Some reverse proxies have a very small upload limit that must be increased. The limit must be at least 33% (recommended 50%) larger than the size of a single document to be uploaded.

Example: If the largest acceptable document size is 60 MB, the upload limit (maximum body size) 90 MB (+50%) would be a safe limit.

TLS/SSL

When the reverse proxy acts as a TLS endpoint, it must reject unsafe or outdated SLL protocols or cipher versions. Also, the HTTP protocol should be disabled.

Block access to URIs

Some resources are not needed for a typical production environment and can be safely blocked:

- /cirrus/swagger
- /cirrus/api-docs
- /cirrus/static/swagger-ui

Example configurations

The following files show an example configuration for a Nginx web server.

File: my_proxy.conf

```
server_tokens off;
client_max_body_size 100m;
```

File: default

```
add_header X-Frame-Options "SAMEORIGIN";
add_header X-Content-Type-Options "nosniff";
add_header X-XSS-Protection "1; mode=block";

location cirrus/swagger {
    return 404;
}
location cirrus/api-docs {
    return 404;
}
location cirrus/static/swagger-ui {
    return 404;
}
location /cirrus/swagger {
    return 404;
}
location /cirrus/api-docs {
    return 404;
}
location /cirrus/static/swagger-ui {
    return 404;
}
```

Content Security Policy (CSP)

The following CSP related HTTP headers can be set in a reverse proxy to enable a Content Security Policy for the Manage and Signing Client.

Manage Client and Signing Client combined (URI: <SignDoc Standard content, usually /cirrus>)

To tailor the CSP for specific clients, refer to this example.

```
Content-Security-Policy: default-src 'none'; style-src 'self' 'unsafe-inline'; script-src 'self' 'unsafe-inline' 'unsafe-eval'; img-src 'self' data: blob;; frame-src 'self'; font-src 'self'; media-src 'self'; object-src 'none'; form-action 'self'
```

Manage Client (URI: /cirrus/static/mc)


```
Content-Security-Policy: default-src 'none'; style-src 'self' 'unsafe-inline'; script-src 'self' 'unsafe-inline' 'unsafe-eval'; img-src 'self' data: blob;; frame-src 'self'; connect-src 'self'; font-src 'self'; media-src 'self'; object-src 'none'; form-action 'self'
```

Signing Client (URI: /cirrus/static/sc)

```
Content-Security-Policy: default-src 'none'; style-src 'self' 'unsafe-inline'; script-src 'self' 'unsafe-inline' 'unsafe-eval'; img-src 'self' data: blob;; frame-src 'self'; font-src 'self'; media-src 'self'; object-src 'none'; form-action 'self'
```

Authentication LDAP


This section describes the specification of the basic authentication via LDAP provided with SignDoc Standard.

 We highly recommend using the file `<INSTALLDIR>\service_configuration.properties` (instead of `cirrus.properties`) whenever it is required to configure SignDoc with a configuration file. Configurations set in this file are applied as a Java system property and therefore have highest precedence.

Prerequisites


The LDAP support in SignDoc Standard is not usable and not supported in a multi-tenant environment. SignDoc Standard maps LDAP user entries to SignDoc Standard users by the unique email address. A SignDoc Standard multi-tenant installation requires email addresses only to be unique within a single account.

The user's ID is defined by the setting `ldap.user.mail.attr`. It must represent the email address of the user.

 This is not a standard LDAP attribute and may have to be added by a system administrator.

Activating LDAP

LDAP support is activated by setting the property `authentication.provider` to the value `LDAP, CIRRUS`.


 Activate LDAP only after you have created the single account.

Auto creating a user

If a user logs in and a user with the mail address received from LDAP does not exist, a user is created automatically in SignDoc Standard. SignDoc Standard maps LDAP attributes to SignDoc


Standard user attributes. Each name of an LDAP attribute has a default value but can be customized by a SignDoc Standard property:

SignDoc Standard property	Default value	Mapped to this SignDoc Standard user attribute	Constraints
ldap.user.name.attr	cn	user name	
ldap.user.mail.attr	mail	OID	(mandatory setting) Must not already exist as a SignDoc Standard user. The email address must match this regular expression: <code>^[A-z0-9\\._%+\\-]+@[A-z0-9\\._-]+</code>
ldap.user.uid.attr	uid		(optional setting) Must not already exist and match the SignDoc Standard validation rule for OID (regex <code>^[a-zA-Z0-9_\\-]+</code>)

 If one of the above constraints are violated, SignDoc Standard will report an error and LDAP integration will not work reliably.

All values can be customized by the `<INSTALLDIR>\service_configuration.properties` file:

- **authentication.provider** (string): Activates LDAP support. Must be 'LDAP,CIRRUS'.
- **ldap.url** (string): The URL to connect to an LDAP server.

 We strongly recommend using the Idaps protocol because passwords are sent in plain text over the network. In this situation, a suitable certificate must be installed at the server.

Example

```
ldap://ad.kofax.com:389/dc=kofax,dc=de
```

- **ldap.manager.dn** (string): The manager DN. If your LDAP implementation does not allow anonymous access, a suitable user and password must be defined here.

Example

```
uid=admin,ou=system
```


- **ldap.manager.password** (string): The manager password.

Example

```
ldap.manager.password=secret
```

Omit manager dn and password for anonymous access.

- **ldap.userdn.patterns** (string): The value is a list of distinguished names (DN) separated by a colon.

 Because the field delimiter is the colon (':'), a DN containing colon(s) must be double-quoted. And a double-quoted DN must escape any double-quote sign with the escape character '\', should it be present in the DN.

Example

```
uid={0},ou=Users
```

The key '{0}' is substituted with the login name.

- **ldap.user.search.base** (string): The base DN for starting a search.

Example

```
dc=kofax,dc=de
```

- **ldap.user.search.filter** (string): A filter for the search (see RFC 2254)

Example

```
(cn=Babs Jensen)
```

- **ldap.user.name.attr** (string): The LDAP attribute that maps to a SignDoc Standard user name.
Default: cn

Example


```
ldap.user.name.attr=cn
```

- **ldap.user.mail.attr** (string):

The LDAP attribute which maps to a SignDoc Standard user email. Default: mail

Example

```
ldap.user.mail.attr=mail
```

 Additional "Brute Force Authentication Prevention" is not implemented if LDAP Authentication is configured.

Windows Authentication

SignDoc supports Windows Authentication for the SQL Server connection.

SignDoc uses the jdbc SQL Driver from Microsoft, so theoretically you can take advantage of each option the jdbc driver offers.

Please check [Building the connection URL](#) for more details.

For Windows Authentication you need to add `integratedSecurity=true` to the `jdbc.url` connection string.

Example:

```
jdbc.url=jdbc:sqlserver://  
<servername>:1433;databaseName=SignDoc;integratedSecurity=true
```

This forces Windows Authentication to be used for the SQL Server connection. The SignDoc service explicitly needs to be started under a Windows user context, which is used for Windows Authentication.

The Windows user can be configured in the `_conf_templates\SignDocStandard.xml` file, so the logon user automatically gets configured to the SignDoc service, when executing `service_up.cmd`.


Chapter 4

SignDoc Authentication Module (SAM)

The SignDoc Authentication Module is required to support Single Sign-on (SSO) scenarios with SignDoc Standard.

SAM provides an integration with external authentication providers. This version implements authentication using SAML2. Supported IDPs are Okta, Azure Active Directory (Azure AD), and Active Directory Federation Services (ADFS).

By clicking the green **Sign in with SSO** button, the users of the SignDoc Manage Client have the option to log into their account using their corporate SSO system.

 Single Sign-On is not possible if the SignDoc Manage Client is used in the Microsoft Teams tab. This is because Microsoft Teams runs SignDoc in an HTML Frame which most IDPs will reject. This is due to their X-Frame-Options settings usually not permitting IDP Authentication inside an HTML Frame.

Overview

SAM is part of SignDoc Standard and is available in the `directory` modules.

The configuration of SAM is done in the file `application.yml` (yaml syntax). Please be aware of the syntax rules of a yaml document.

The SAML2 configuration is done in SignDoc Standard.

Installation

To install SAM

1. Extract SAM to a suitable directory.
2. Complete the required configuration as described in the below configuration sections for SAM, SignDoc and SSO.

To start or restart the Windows service, run:


```
service_up.cmd
```

To stop or remove the Windows service, run:


```
service_remove.cmd
```

Docker image

A sample Docker file shows how to create an image that contains `application.yml` plus a PKCS12 certificate (to be provided) embedded in the image.

 This Docker image is only an example and has to be adapted if the configuration/certificate data should be managed outside the container.

SAM configuration with `application.yml`

The configuration done in `application.yml` is important from a security perspective since it creates a secure bond between SAM and SignDoc Standard.

servers: section

In this section general server settings such as `server.port` and `server.ssl` (for HTTPS support) can be set. Generally, this section can be configured using the standard Spring Boot configuration options and is not specific to SignDoc.

authmodule: section

This section is important since it configures the behavior of SAM.

```
authmodule.external_url:[URL with a FQDN]
```

We recommend to define with `external_url` the official external URL the application can be accessed. If `external_url` is not set, the application will try to dynamically construct this URL. The latter might not be reliable depending on the environment.

signdoc: section

This section is important since it configures the bonding between SAM and SignDoc Standard. Since SAM and SignDoc Standard are in general running on different systems, there must be a secure connection server side to exchange data. This is achieved by defining a unique shared secret (`shasec`) for a SignDoc Standard installation. This shared secret is declared in the `servers:` section for every server. SAM can bond with multiple SignDoc Standard servers at the same time. We recommend that every SignDoc Standard server defines its own unique secret.

shasec configuration

Generate a random string, such as a random UUID, to be used as the `shasec` secret.

Go to the SignDoc Administration Center and assign the `shasec` secret to the configuration setting `cirrus.security.ksd_appconf_shasec`.

The SAM configuration in `application.yml` is done by providing a list of `url` and `shasec` combinations per SignDoc Standard server:

i If one entry has `url=""`, it acts as a default for unknown servers, which means the `shasec` specified for the `url=""` entry is used for all URLs that are not explicitly listed.

```
servers:
- url: ""      # context url (CIRRUS_URL) id the SDS.
  shasec: ""   # shared secret
```

Example:

```
signdoc:
  servers:
  - url: "https://my_1st_signdoc_server/cirrus"
    shasec: "2ee6f243-4a0a-48ce-9144-ab5adc8fc8bd"
  - url: "https://my_2nd_signdoc_server/cirrus"
    shasec: "5c6e9601-96ae-431e-a628-ae6cbb6948be"
```

logging: section

In this section the logging output can be adjusted. This follows standard Spring Boot logging. Log output relevant to SAM can be adjusted by changing the log level of `de.softpro` loggers. Valid log levels: INFO, DEBUG, TRACE.

Example, enabling DEBUG output for SAM code:

```
logging:
  level:
    de.softpro: DEBUG
```

Redis support

It is possible to use Redis for server-side session management. This should be considered, when SAM is hosted behind a load balancer with multiple instances to improve high availability of the service.

Redis support configuration options:

```
spring.data.redis.host: "localhost"
spring.data.redis.port: 6379
spring.data.redis.password: "2beChanged!"
```

If required, you can find more configuration options here: [Spring Data Properties](#). Look for properties that start with `spring.data.redis`.

SignDoc Standard configuration

shasec

Make sure that the SignDoc Standard installation has set a unique value for `cirrus.security.ksd_appconf_shasec` in the Administration Center to enable a secure communication channel between SAM and SignDoc Standard. See also the section on `shasec` configuration above.

SSO configuration in Manage Client and Administration Center

The SSO configuration is account-specific. That is, every account can define its own SAM instance.

i Since SignDoc 3.3.0, a single SAM can handle all accounts of a SignDoc Standard installation. Only a single SAM instance is required.

In SignDoc Standard the SSO configuration is split into 2 parts:

- General SSO settings.
The general settings are described in the SSO section of the account configuration.
- SAML-related settings.
The SAML related settings, including descriptions, are available in the SAML section of the account configuration.

About "IDP initiated login" support

With the configuration option `cirrus.sso.saml.idp.initiated_login` it is possible to enable / disable support for an IDP initiated login.

If set to "On" (default), the user can not only start the login process using the green **Sign in with SSO** button in SignDoc, but also log in to the Manage Client directly from the IDP application. To save unnecessary requests on server side, we recommend supporting `sticky sessions` if the SAM is hosted behind a reverse proxy or load balancer.

The setting must be set to "Off" for all SignDoc accounts that share the same IDP application for authentication with each other. In this case, an IDP initiated login is not possible for these accounts. It is required for this use case to support `sticky sessions` if SAM is hosted behind a reverse proxy or load balancer.

SSO configuration options

The following configuration options in SignDoc Standard can modify the SSO behavior of SignDoc Standard:

- `cirrus.sso.auth.module.url`
- `cirrus.sso.autologin`
- `cirrus.sso.create.user`
- `cirrus.sso.create.user.account`
- `cirrus.sso.sanitize.userid`
- `cirrus.sso.saml.idp.initiated_login`
- `cirrus.sso.saml.idp.metadata.url`
- `cirrus.sso.saml.idp.metadata.xml`
- `cirrus.sso.saml.idp.attribute.email`
- `cirrus.sso.saml.idp.attribute.login`
- `cirrus.sso.saml.idp.attribute.name`

- `cirrus.sso.saml.idp.attribute.first_name`
- `cirrus.sso.saml.idp.attribute.last_name`
- `cirrus.sso.saml.idp.attribute.roles`
- `cirrus.sso.saml.idp.attribute.teams`
- `cirrus.sso.saml.display_result`
- `cirrus.sso.saml.display_config`

For more details, see the [Kofax SignDoc Standard Administrator's Guide](#).

Step by step example

This example includes instructions to set up a new SSO authentication for a SignDoc Standard `sales` account.

Given:

- SignDoc Standard is installed.
 - Account `sales` is available in SignDoc.
 - `CIRRUS_URL`: `https://signdoc_server/cirrus`
- SAM 3.3 is installed.
 - `SAM_URL`: `https://signdoc_authentication_module:6612`
- Random `shasec`: `12a0b68a-fb4f-4e7a-ad6b-555f2376c5b8`

Procedure:

1. Set the unique `shasec` value in the SignDoc Standard Administration Center and save the setting.
2. Set the `cirrus.sso.create.user.account` to `sales` in the SignDoc Standard Administration Center and save the setting. This will use `sales` as default for SSO.
3. Set the `cirrus.sso.auth.module.url` in the `sales` SignDoc Standard account settings and save the settings.

`cirrus.sso.auth.module.url => https://signdoc_authentication_module:6612`

4. Add a new URL/shasec mapping to `application.yml`.

```
signdoc:
  servers:
    - url: "https://signdoc_server/cirrus"
      shasec: "12a0b68a-fb4f-4e7a-ad6b-555f2376c5b8"
```

5. Set the `external_url` in `application.yml`.

```
authmodule:
  external_url: "https://signdoc_authentication_module:6612"
```

6. Restart SAM using `service_up`.
7. Set the `cirrus.sso.saml.display_result` and `cirrus.sso.saml.display_config` in the `sales` SignDoc Standard account administration to "On" and save the settings.
8. Open `https://signdoc_server/cirrus` in the browser and click the green **Sign in with SSO** button.

You should now see a page that displays SAML-related information to be used in the IDP. Use these two values to configure the SAML Service at the IDP for this SAM instance:

- Identifier (Entity ID)
 - Reply URL (Assertion Consumer Service URL)
9. Copy the Metadata URL or Metadata XML of the SAML Service as provided by the IDP and assign it to the account configuration option `cirrus.sso.saml.idp.metadata.url` or `cirrus.sso.saml.idp.metadata.xml` of the `sales` account and save the settings.
 10. Set the `cirrus.sso.saml.display_config` in the `sales` SignDoc Standard account administration to "Off" and save the setting.
 11. Open `https://signdoc_server/cirrus` in the browser and click again the green **Sign in with SSO** button. You should now be redirected to the IDP. After successful authentication, you should see a page that lists all SAML Authentication Assertion attributes. Verify that these attributes are mapped by the settings of the SAML configuration section in the `sales` SignDoc Standard account administration. Be sure to confirm that the attributes defining the user's name are correctly mapped. SignDoc Standard provides some sensible defaults for Okta and Microsoft Active Directory.
 12. If the attributes make sense, set the `cirrus.sso.saml.display_result` in the `sales` SignDoc Standard account administration to "Off" and save the setting.
 13. Open `https://signdoc_server/cirrus` in the browser and click again the green **Sign in with SSO** button. This time, after clicking this button, you should be logged in to the `sales` account automatically. If your user account did not exist before in SignDoc Standard, it was automatically created.

References

For more information on YAML and YAML specifications, see the [YAML website](#).

Find detailed information on Spring boot properties and Spring data properties documentation on the [Spring website](#).

- [Spring Boot Properties](#)
- [Spring Data Properties](#)

Chapter 5

Upgrade SignDoc Standard

SignDoc can be upgraded from any previous version. Unless specified in the version-specific sections below, the following generic upgrade procedure can be applied:

1. Stop SignDoc using `service_remove.cmd`. See "Tools" in [Content of the SignDoc Standard ZIP archive](#) section.
2. Make a backup of the database or create a snapshot of the database that can be restored, in case the upgrade fails.
3. Install the new SignDoc version as described in [Quick start](#).
4. Apply the existing old SignDoc configuration to the new SignDoc installation. For example, apply any existing configuration from `service_configuration.properties` (and possibly other configuration files) to the new installation. The configuration typically affects the database connection, SMTP server, `SERVICE_EXTERNAL_HOST_URL`, and more.
5. Make sure database migrations are enabled (this is the default). See "Control database migrations" in the [Advanced configuration](#) section.
6. Start the new SignDoc version using `service_up.cmd`. See "Tools" in [Content of the SignDoc Standard ZIP archive](#) section.

i After the system is upgraded, it is no longer possible or supported to connect the old SignDoc installation to the upgraded/migrated database.

Upgrade from SignDoc Standard 2.2.1

If you are upgrading an existing SignDoc 2.2.1 installation (version \leq 2.2.1.2.0.63) you are required to make configuration changes.

File: `service_configuration.properties`

- The `jdbc.password` configuration setting must not contain any of these 3 characters: `<` `>` `&`
- The `jdbc.url` configuration setting MUST NO LONGER be enclosed in quotes as it was true for SignDoc up to 2.2.1.2.0.63. Until now, the `jdbc.url` value had to be enclosed in single quotes, if there were spaces in `jdbc.url`. If this should be the case for the installation to upgrade, the single quotes must now be removed.

Example

Invalid settings:

```
jdbc.url='jdbc:h2:${SIGNDOC_HOME}/db/  
signdoc_database;MVCC=TRUE;DB_CLOSE_DELAY=-1;INIT=SET COLLATION ENGLISH  
STRENGTH SECONDARY'
```

Valid settings:

```
jdbc.url=jdbc:h2:${SIGNDOC_HOME}/db/  
signdoc_database;MVCC=TRUE;DB_CLOSE_DELAY=-1;INIT=SET COLLATION ENGLISH  
STRENGTH SECONDARY
```

Upgrade from SignDoc Standard 1.3.1 or earlier versions

To upgrade an existing SignDoc Standard 1.3.1 or earlier version, the following steps need to be performed:

1. Stop and disable automatic restart for all existing SignDoc Standard instances older than SignDoc 2.1.0.
Don't stop the SignDoc Standard database.
If applicable: Remove any global or system PATH setting that contains the existing SignDoc Standard instance.
2. Install SignDoc 3.3.0 as described in [Quick start](#).
3. Make sure database migrations are enabled (this is the default). See [Control database migrations](#).
4. Configure SignDoc 3.3.0 for production as described in [Production setup](#). We recommend that you apply a fresh configuration.
 - If **KTA integration** is used, the existing configuration section of `cirrus.properties` can be copied to the new installation. See [KTA integration](#).
 - If **LDAP authentication** is used, the existing configuration section of `cirrus.properties` can be copied to the new installation. See [LDAP integration](#).
5. Start the new SignDoc version using `service_up.cmd`. See "Tools" in [Content of the SignDoc Standard ZIP archive](#) section.
6. Open `http://<your_server>:<port>/cirrus/client` and log in.

Upgrade troubleshooting

It should be sufficient to make sure that there are no PATH entries that point to an old SignDoc Standard installation. If an installation fails, verify the following:

- [General prerequisites](#) are met.
- The installation is performed on a supported Windows Server operating system specified in the [Kofax SignDoc Technical Specifications](#) document.
- There should be no Windows PATH (system or user) entry (environment variable) pointing to an old SignDoc Standard installation.
- There should be no CIRRUS_HOME environment variable set (system or user).

Chapter 6

Database migration

This chapter describes the database migration mechanism used by SignDoc Standard.

Any product that uses a schema-based database faces the challenge of managing database changes while migrating from one version to another. This includes changes to the database schema, such as adding a new column, moving data from one location to another, and more. Not only the database has to be adapted to the new schema, the existing data has to be migrated to fit it.

Database migrations standardize the way this is done, keeping track of the versions that have been applied to the data.

Database migration with Flyway

SignDoc Standard uses Flyway to standardize database migration scripts. For more information on Flyway, see the documentation on the Flyway website. Flyway uses migrations that are named to a specific schema, containing the version number and description in the file name. It also keeps track of the version the database currently has and all applied changes by creating a database table named `schema_version` and recording all migrations it has done.


Since the migration scripts (or migration Java classes) are part of the product, you can always tell what state the database is in and what changes still need to be applied.

Flyway migrations are integrated in SignDoc Standard. When the application starts, it checks the database version and executes any outstanding migrations in the order of their version number, thus bringing the database up to date.

Load balancing considerations

If multiple SignDoc nodes are hosted behind a load balancer (or reverse proxy), consider the following before upgrading to a new SignDoc version:

- Shut down all SignDoc nodes (instances) prior to starting the latest version.
- The first startup of the new SignDoc version should be done with a single node (instance) to process all migration tasks. After the first node is completed (the application or REST API is accessible), start the other nodes (instances).

 We strongly recommend that you perform a database backup before attempting to migrate the database. Due to the nature of some migration steps and the fact that multiple migration steps are applied during a version update, a database rollback is not possible.

Chapter 7

Troubleshooting

This chapter describes the issues you may encounter and their resolution.

General

If you need to contact Kofax Support, it is useful to compile the following log and configuration files:

Log files

- All files in the directory `<INSTALLDIR>\signdoc_home\logs\`
- All files in the directory `<INSTALLDIR>\service\logs\`
- Windows event log
- Firewall logs
- Browser console log

Configuration files

- All files in the directory `<INSTALLDIR>\signdoc_home\conf\`

 Make sure to delete sensitive information.

Email settings

When an error occurs sending an email, check and fix the following:

- You specified the wrong server. The server you specified exists, but it is not an SMTP server.
- You specified the wrong port number. Ask the person who runs the SMTP server what the correct port number is.
- The server is down. This is usually temporary. If it persists, contact the server administrator.
- Your firewall is blocking the port.
- Your ISP is blocking the port. This usually affects port 25, and you can often work around it by using port 587, but details depend on your ISP and on the SMTP server's configuration.
- You specified TLS, but the server does not support it.

Apache Tomcat

When issues related to Apache Tomcat occur, ensure the following:

Security

- For a production environment, restrict the communication between cirrus and sdweb to local interfaces.

Performance

- The connector default size in bytes for POST requests is limited to 2 MB. Increase the size to handle larger documents by adding the attribute `maxPostSize="52428800"` to the Tomcat connector definition.

Appendix A

Glossary

A

Assertion

The response of the IPD to the SP containing information about the user.

C

CIRRUS_URL

The URL of the SignDoc Standard installation pointing to the `/cirrus` context.

Example: `https://my_signdoc_server/cirrus`

FQDN

Fully Qualified Domain Name. Example: `www.kofax.com`

I

IDP

SAML2 Identity Provider. A software service that can authenticate users.

Example: Okta, Azure Active Directory (Azure AD), Active Directory Federation Services (ADFS)

S

SAM

SignDoc Authentication Module.

SAML / SAML2

A secure protocol to identify users over the network.

shasec

A shared secret between SAM and a SignDoc Standard installation.

SP

Service Provider. This is the SignDoc Authentication Module itself.

SSO

Single Sign-on. An environment that allows a user to authenticate against a central instance (IDP). By doing this the user gets implicit authorization services supporting Single Sign-on.